République Tunisienne

Ministère de l'Enseignement Supérieur
et de la recherche scientifique

Université de Gafsa
Institut Supérieur des Sciences
Appliquées et de la Technologie de Gafsa

Cycle de Formation en Mastère

Professionnelle dans la
Discipline Expert Cyber Sécurité

Mémoire de *MASTERE*
Expert Cyber Sécurité
N° d'ordre : 02−ECS

MEMOIRE

Présenté à

L'Institut Supérieur des Sciences Appliquées et de Technologie de Gafsa

(Département Informatique et télécommunication)
En vue de l'obtention Diplôme en Expert Cyber Sécurité

MASTERE

Dans la discipline Expert Cyber Sécurité

Par

Taissir BRIKI

CREATION D'UN MODELE DETECTION D'INTRUSION BASE SUR L'APPRENTISSAGE AUTOMATIQUE

Soutenu le 06/06/2023

Soutenu devant le jury composé de :

Mme.	**Fatma HRIZI**	*Président*
M.	**Ali KOTTI**	*Rapporteur*
Mme.	**Haifa CHORFI**	*Encadreur*
M.	**Ahmed KHLIFI**	*Co-Encadreur*

A.U : 2022 – 2023

Dédicaces

A mes très chers parents (faleh & moufida)

Tous les mots du monde ne sauraient exprimer l'immense amour qui je vous porte, ni la profonde gratitude que je vous témoigne pour tous les efforts et les sacrifices que vous n'avez jamais cessé de consentir pour mon instruction et mon bien-être.

A mes très Cher frères.

Qui n'ont pas cessé de me conseiller, encourager et soutenir tout au long de mes études

Que dieu les protège et leur offre la chance et le bonheur.

A tous mes amis de promotion deuxième année master ECS toute personne qui occupe une place dans mon cœur

Taissir

Remerciements

Je voudrais premièrement remercier Dieu pour toutes les bénédictions dont il a comblé ma vie. Ce mémoire n'aurait pas été possible sans l'intervention d'un grand nombre de personnes.

*Madame « **CHORFIHaïfa** »*

J'aimerais, exprimer toute ma gratitude pour la qualité et la complémentarité de son encadrement

*Je remercie Monsieur « **KHLIFI Ahmed** » d'avoir accepté de diriger ce travail de recherche.*

Je tiens également à exprimer ma gratitude à ma famille, en particulier à ma mère, mes frères qui m'ont toujours soutenu et poussé à poursuivre mes études. Grâce à leur soutien, ce travail actuel devient possible.

Mes vifs remerciements s'adressent également aux membres de jury pour avoir accepté de juger mon travail et tous mes enseignants de L'ISSAT pour leurs générosités et la grande patience dont ils ont su faire preuve malgré leurs charges académiques et professionnelles

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

Sommaire

Liste des abréviations

ACID : Atomicité, Cohérence, Isolation et Durabilité

ANN : Artificiel News Network

BDD : Base de données

CGI: Common Gateway Interface

CNN: Cable News Network

CPU: Central Processing Unit

DNS: Domain Name System

FTP: File Transfer Protocol

GPU: Graphics Processing Unit

HIDS: Host Intrusion Detection System

HTTP: Hypertext transfert Protocol

IA : Intelligence Artificiel

ICMP : Internet Control Message Protocol

IDS: Intrusion Detection System

IP: Internet Protocol

IPS: Intrusion Prevention System

K-NN: K- Nearest Neighbors

NIDS: Network Intrusion Detection System

NSM: Network and System Management

RAM: Random Access Memory

RNN: Recurrent Neural Network

SMB: Server Message Block

TCP: signifie Transmission Control Protocol

URL: Uniform Ressource Locator

WSN: Wireless Sensor Networks

Liste des Figure

Liste des tableaux

Introduction générale

Les réseaux et les systèmes informatiques sont devenus des outils indispensables au fonctionnement des entreprises. Ils sont aujourd'hui déployés dans tous les secteurs professionnels : les universités, les banques, les assurances ou encore le domaine militaire.

L'informatique gérée par ces systèmes fait l'objet de convoitises. Elle peut être exposée à des attaques qui exploitent des éléments vulnérables du système d'information. La détection des actions malveillantes est rapidement devenue une nécessité. Les mesures de prévention se sont révélées insuffisantes et ont amené la création de systèmes de détection d'intrusions (IDS : Intrusion Détection systèmes).

Une intrusion est définie comme étant toute tentative pouvant nuire à l'intégralité, la confidentialité ou la disponibilité dans le réseau ainsi que toute tentative visant à contourner les dispositifs de sécurité mis en place sur le réseau ou une machine. Ces tentatives d'intrusions peuvent être bénignes comme extrêmement dangereuses et préjudiciable pour l'entreprise.

Le domaine de la détection d'intrusion est encore jeune mais en plein développement. Nous dénombrons à l'heure actuelle environ une centaine de systèmes de détections d'intrusions (ou IDS pour Intrusion Detection System), que ce soit des produits commerciaux ou du domaine public. Ces systèmes de surveillance du réseau sont devenus pratiquement indispensables dû à l'incessant accroissement en nombre et en dangerosité des attaques réseaux depuis quelques années.

Dans notre travail nous s'intéressent sur les différentes techniques utilisées dans la détection d'intrusion, généralement ces dernières sont des techniques de datamining ou de machine Learning et technique statique

Notre mémoire est organisé comme suit :

- Le premier chapitre est consacré à la présentation des différents aspects de la sécurité informatique et les systèmes de détection d'intrusion.
- Le deuxième chapitre est axé à la présentation des techniques de la détection d'intrusion. Nous avons présenté : Machine Learning, Datamining et les techniques statiques.
- Le troisième chapitre consiste à présenter trois méthodes de la technique machine Learning et établir une comparât entre ces méthodes.

Chapitre 1 : Etat de l'art

1. Introduction

Les systèmes de détection d'intrusion sont des systèmes de sécurité conçus pour surveiller le flux de données via des réseaux de capteurs sans fil afin de détecter les intrus et de faire face non seulement aux attaques connues, mais également aux attaques inconnues.

2. L'intrusion

Avec le développement continu des nouvelles technologies liées à l'informatique, il ya un nombre croissant de problèmes de piratage et d'intrusion en cours. Ce piratage et les intrusions peuvent être trouvés dans un large éventail d'ordinateurs de bureau, les réseaux et le Internet. Dans cette partie, nous parlerons de l'intrusion en général

2.1. Définition

En sécurité informatique, la détection d'intrusion est l'acte de détecter les actions qui essaient de compromettre la confidentialité, l'intégrité ou la disponibilité d'une ressource. La détection d'intrusion peut être effectuée manuellement ou automatiquement. Dans le processus de détection d'intrusion manuelle, un analyste humain procède à l'examen de fichiers de logs à la recherche de tout signe suspect pouvant indiquer une intrusion

2.2. Fonctionnement

Lorsqu'une intrusion est découverte par un système de détection d'intrusion, les actions typiques qu'il peut entreprendre sont par exemple d'enregistrer l'information pertinente dans un fichier ou une base de données, de générer une alerte par e-mail ou un message sur un pager ou un téléphone mobile. Déterminer quelle est réellement l'intrusion détectée et entreprendre certaines actions pour y mettre fin ou l'empêcher de se reproduire, ne font généralement pas partie du domaine de la détection d'intrusion. Cependant, quelques formes de réaction automatique peuvent être implémentées par l'interaction de systèmes de détection d'intrusion et de systèmes de contrôle d'accès tels que les pares-feux. Les techniques de détection d'intrusion. Deux techniques de détection d'intrusion sont généralement mises en œuvre par les systèmes de détection d'intrusion courants :

- **La détection d'abus (misuse detection)** : dans la détection d'abus (aussi appelée détection de mauvaise utilisation), systèmes de détection d'intrusion analyse

l'information recueillie et la compare (pattern matching, approche par scénarii) avec une base de données

- **La détection d'anomalie (anomaly detection) :** la détection d'anomalie de comportement est une technique assez ancienne (elle est utilisée également pour détecter des comportements suspects en téléphonie, comme le phreaking). L'idée principale est de modéliser durant une période d'apprentissage le comportement « normal » d'un système/programme/utilisateur en définissant une ligne de conduite (dite baseline ou profil3), et de considérer ensuite (en phase de détection) comme suspect tout comportement inhabituel (les déviations significatives par rapport au modèle de comportement « normal »).

2.3. Différents types des systèmes de détection d'intrusion

Les attaques utilisées par les pirates sont très variées, puisque certaines utilisent des failles réseaux et d'autres des failles de programmation. C'est la raison pour laquelle la détection d'intrusion doit se faire à plusieurs niveaux.

Donc, on distingue deux systèmes de détection d'intrusion que nous détaillerons ci-dessous les caractéristiques principales.

2.3.1. Systèmes de détection d'intrusion réseau (NIDS)

Les IDS réseaux analysent en temps réel le trafic qu'ils aspirent à l'aide d'une sonde (carte réseau en mode promiscuous Ensuite, les paquets sont décortiqués puis analysés.

Il est fréquent de trouver plusieurs systèmes de détection d'intrusion sur les différentes parties du réseau. On trouve souvent une architecture composée d'une sonde placée à l'extérieure du réseau afin d'étudier les tentatives d'attaques et d'une sonde en interne pour analyser les requêtes ayant traversé le pare-feu. [10]

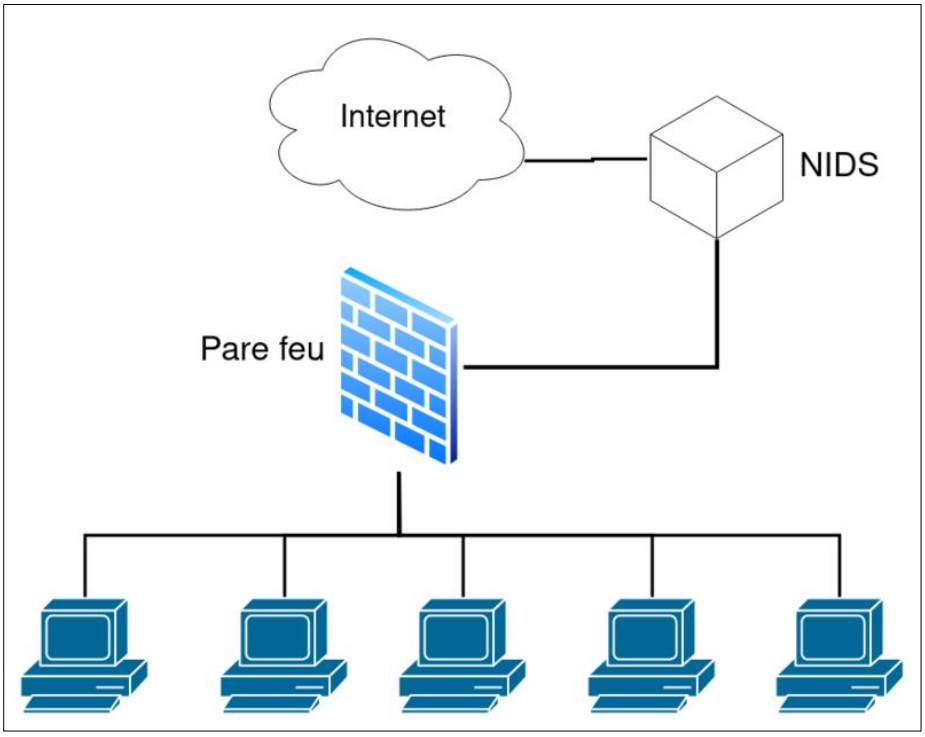

Figure 1. 1: Systèmes de détection d'intrusion réseau (NIDS)[10]

2.3.2. Systèmes de détection d'intrusion de type hôte (HIDS)

Les systèmes de détection d'intrusion analysent le fonctionnement de l'état des machines sur lesquelles ils sont installés afin de détecter les attaques en se basant sur des démons (tels que syslogdpar exemple). L'intégrité des systèmes est alors vérifiée périodiquement et des alertes peuvent être levées. [10]

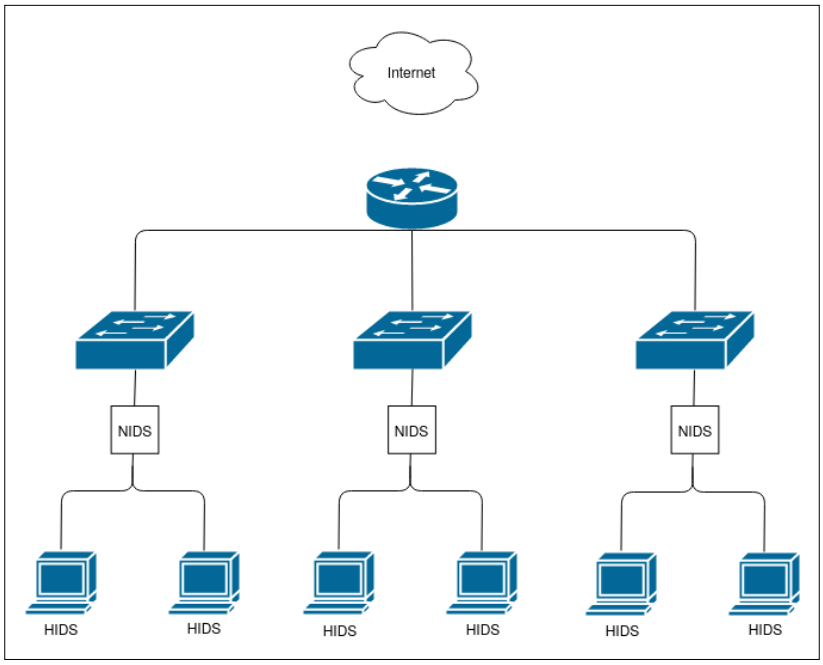

Figure 1. 2: Systèmes de détection d'intrusion de type hôte (HIDS)[10]

2.3.3. Systèmes de détection d'intrusion Hybrides

Les IDS hybrides rassemblent les caractéristiques des NIDS et HIDS. Ils permettent, en un seul outil, de surveiller le réseau et les terminaux. Les sondes sont placées en des points stratégiques, et agissent comme NIDS et/ou HIDS suivant leurs emplacements. Toutes ces sondes remontent alors les alertes à une machine qui va centraliser le tout, et lier les informations d'origine multiple. Ainsi, on comprend que les IDS hybrides sont basés sur une architecture distribuée, ou chaque composant unifie son format d'envoi. Cela permet de communiquer et d'extraire des alertes plus pertinentes. [10]

2.4. Etude comparatif

Le HIDS et le NIDS sont des systèmes de détection d'intrusion. Bien qu'ils visent la sécurité, les deux systèmes présentent des différences sur de nombreux points comme représente le tableau ci-dessous :

Tableau 1. 1:la comparaison entre NIDS, HIDS et hybrides

	NIDS	HIDS	Systèmes hybrides
Avantages	Contrôler un grand nombre d'hôtes avec un petit coût de déploiement. Identifier les attaques de/à multiples hôtes. Assurer la sécurité contre les attaques puisqu'il est invisible	Contrôler les activités locales des utilisateurs avec précision. Capable de déterminer si une tentative d'attaque est couronnée de succès. Capable de fonctionner dans des environnements cryptés	Moins de faux positifs. Meilleure corrélation (la corrélation permet de générer de nouvelles alertes à partir de celles existantes). Possibilité de réaction sur les analyseurs.
Inconvénients	Incapable de fonctionner dans des environnements cryptés Ne permet pas d'assurer si une tentative d'attaque est couronnée de succès.	Ladifficulté de déploiement et de gestion, surtout lorsque le nombre d'hôte qui ont besoin de protection est large. Incapable de détecter des attaques contre de multiples cibles dans le réseau.	Ils sont plus chers qu'une installation classique.

3. Système de détection d'intrusion (IDS)

Un système de détection d'intrusion est un mécanisme destine à repérer des activités anormales ou suspectes sur la cible analysée (réseau, hôte). Debar simplifie le système de détection d'intrusion dans un détecteur qui analyse les informations en provenance du système surveille.

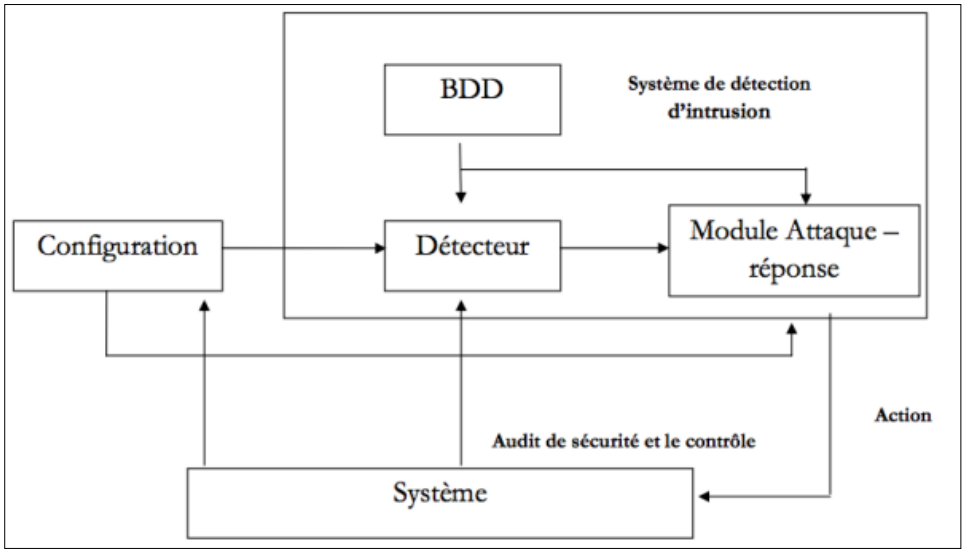

Figure 1. 3:Modelé simplifie d'un système de détection d'intrusions

Dans les systèmes d'information, les intrusions peuvent être définies comme toutes les activités qui violent la politique de sécurité du système[1]. Un pirate informatique ou un attaquant qui essaie de trouver un moyen d'obtenir un accès non autorisé à des informations, de causer des dommages ou de se livrer à d'autres activités malveillantes.

La détection d'intrusion est le processus de surveillance et d'analyse des événements qui se produisent dans un réseau ou un système informatique pour détecter les signes de menaces imminentes susceptibles de violer les politiques de sécurité du système informatique ou les pratiques de sécurité standard [2]. Un système de détection d'intrusion est un ensemble de composants matériels et logiciels conçus pour automatiser le processus de détection d'intrusion.

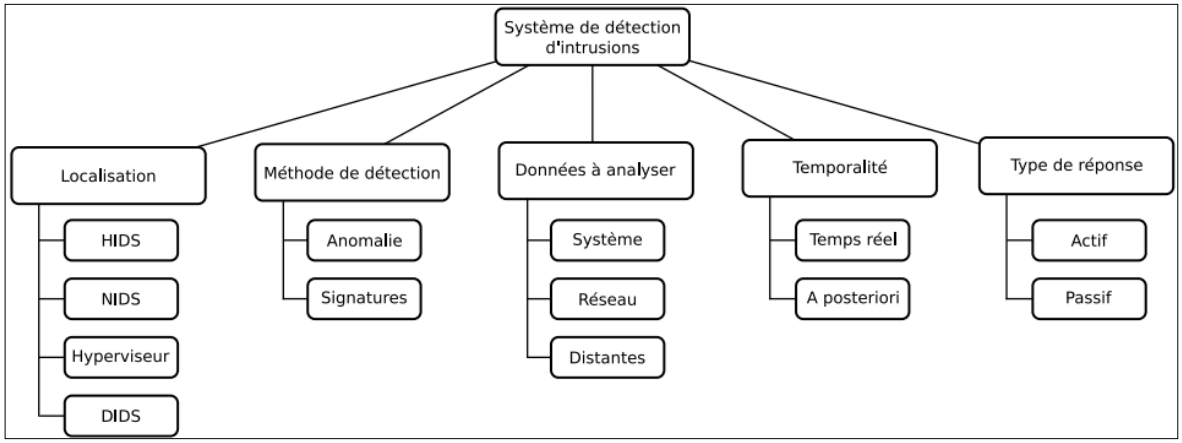

Figure 1. 4: Classification des IDS[3]

3.1. Evaluation d'un système de détection d'intrusion

Philip et définit trois critères pour évaluer l'efficacité des systèmes de détection d'intrusion [3] :

- **L'exactitude (accuracy)** on parle de l'exactitude quand le système de détection d'intrusion déclare comme malicieux une activité légale. Ce critère correspond au faux positif.
- **La performance (performance)** la performance de système de détection d'intrusion est le taux de traitement des évènements. Si ce taux est faible, la détection en temps réel est donc impossible.
- **La complétude (completeness)** on parle de la complétude quand le système de détection d'intrusion rate la détection d'une attaque. Ce critère est le plus difficile, parce qu'il est impossible d'avoir une connaissance globale sur les attaques. Ce critère correspond au vrai négatif.

Debar et al dans [4] a rajouté également les deux critères suivants :

- **La tolérance aux fautes (Fault tolerance)** le système de détection d'intrusion doit lui-même résisté aux attaques, particulièrement au déni de service. Ceci est important, parce que plusieurs systèmes de détection d'intrusion s'exécutent sur des matériels ou logiciels connus comme vulnérables aux attaques.
- **La réaction à temps (Timeliness)** le système de détection d'intrusion doit s'exécuter et propager les résultats de l'analyse le plus tôt possible, pour permettre à l'officier de sécurité de réagir avant que des graves dommages n'aient lieu. Ceci implique plus qu'un calcul de performance, parce qu'il ne s'agit pas seulement de temps de traitement des évènements, mais aussi le temps nécessaire pour la propagation et la réaction à cet évènement.

3.2. Les composants d'un système de détection d'intrusion

Un système de détection d'intrusion se compose généralement des composants illustrés ci-dessous [6] :

3.2.1. Les agents locaux

Ils surveillent les activités locales d'un nœud capteur [6]. C'est-à-dire : ils vérifient l'information traitée par le nœud, les lectures de capteur ou des paquets transmis par le nœud. Ils sont imbriqués dans chaque nœud capteur et activés lorsque le nœud doit traiter de l'information.

3.2.2. Les agents globaux

Ils surveillent les communications de leurs voisins, traitant l'information qui peut être extraite. Ils sont également situés dans chaque nœud et exécutent leurs tests périodiquement. En outre, il est possible d'ajuster leurs niveaux d'activation dus à la redondance du réseau. Noter que, pour les réseaux ayant une architecture en one-hop clusters, il peut être possible d'inclure l'agent global de nœud seulement dans les clusters head.

3.2.3. Les agents de station de base

Analysent la circulation de l'information obtenue à partir des nœuds capteur. Puisque la station de base est riche en ressources, elle peut être en activité à tout moment, exécutant des algorithmes complexes pour découvrir toutes les variations qui peuvent mener à localiser une intrusion. Chacun des agents définis précédemment devrait se composer des composants suivants [7] :

3.2.4. L'acquisition de données

Ce composant rassemble les données à partir des autres nœuds capteur. Ces données doivent être traitées et analysées ensuite stockées par le composant statistique [6]. La source de ces données dépend de types d'agents à partir desquels elles sont récupérées. Ainsi, pour les agents locaux les données sont obtenues depuis les paquets détectés et traités ou envoyés par le nœud. Pour les agents globaux les données sont obtenues depuis les paquets circulants sur son voisinage qui ne sont pas forcément destinés à lui.

3.2.5. Un composant statique

Son rôle est de stoker les données traitées et analysées par le composant d'acquisition. Le contenu principal de ce composant doit être partagé par les agents.

3.2.6. Un composant de détection

Il est responsable d'obtenir les résultats des composants définis précédemment puis analyser ces résultats pour signaler en cas de signes d'intrusions [8]. Dans les cas où le composant de détection considérerait qu'un nœud se comporte d'une manière anormale, il doit stocker cette information dans la base de données d'alerte afin d'économiser l'énergie, les tests liés au composant statistique doivent être exécutés périodiquement.

3.2.7. La base de données d'alerte

Elle est utilisée pour stocker les informations de sécurité générées par les agents. Le format et la taille de cette base dépend des protocoles utilisés. Cependant, elle doit contenir les informations suivantes : le temps de création, classification et source d'alerte [9].

3.2.8. Le composant de collaboration

Il peut être activé quand la communication avec d'autres parties du système ou du voisinage est nécessaire.

3.3. Les techniques de détection d'intrusion

Afin de détecter un intrus, nous devons employer un modèle de détection d'intrusion. Un IDS doit pouvoir distinguer le comportement normal et anormal, afin de découvrir les tentatives malveillantes à temps.

La détection d'intrusion pour les réseaux de capteurs peut être classifiée dans trois larges catégories : détection par scénarios, détection comportementale et la détection par spécification.

3.3.1. La détection par scénarios

Cette technique consiste à créer une base de signature d'attaques. En se basant sur celle-ci, une comparaison entre les comportements observés et les scénarios d'attaques prédéfinis dans cette dernière sera faite [18]. Cette technique essaie de détecter l'évidence de l'activité intrusive indépendamment de n'importe quelle connaissance concernant le comportement normal du système.

L'avantage de cette technique réside dans l'efficacité de détecter les attaques connues dans la base de signature. Par contre son inconvénient majeur est qu'on ne peut pas détecter des attaques originales. La base de données de signature d'attaques doit continuellement être mise à jour de façon à ce que :

- Les règles de mise à jour doivent être effectuées d'une façon fiable et sécurisée, ellesne sont pas faciles à réaliser dans WSN.
- Les données de la base de signatures seront stockées sur les nœuds cependant la capacité mémoire de ces derniers est réduit.

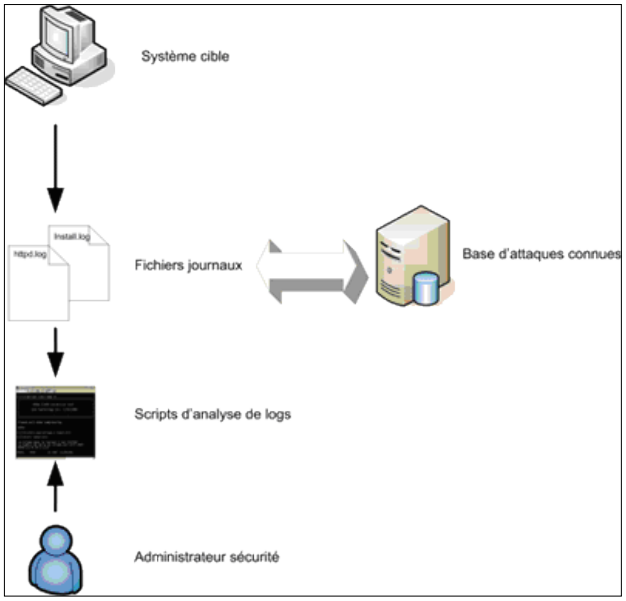

Figure 1. 5: L'approche par scénario

3.3.2. La détection par comportement

Dans cette technique un modèle de référence de comportements normaux du système est créé. Le comportement observé du système cible est comparé aux comportements normaux et espérés. Si le comportement du système est significativement différent du comportement normal ou attendu, on dit que le système cible présente des anomalies et fait l'objet d'une intrusion.

L'avantage majeur de cette technique est de pouvoir détecter de nouvelles attaques. Cependant, elle génère souvent de nombreuses fausses alertes car une déviation du comportement normal ne correspond pas toujours à l'occurrence d'une attaque, et le profil normal doit être périodiquement mis à jour et les déviations du profil normal calculé.

3.3.3. La détection par spécifications

Cette technique est également basée sur les déviations de comportement normal afin de détecter des attaques, mais le comportement normal est spécifié manuellement comme un ensemble de contrainte du système. De cette façon, les comportements légitimes invisibles ne causeront pas un taux élevé de fausses alertes, comme dans l'approche de détection d'anomalie. En outre, puisqu'il est basé sur des déviations des comportements légitimes, elle peut encore détecter des attaques inconnues.

3.4. Exemples de systèmes de détection d'intrusion Classique

Dans cette partie nous avons présenté des exemples des systèmes de détection d'intrusion classique

3.4.1. Snort

Snort est un système de détection d'intrusion (IDS) et de prévention d'intrusion (IPS) gratuit et open-source1 créé en 1998 par Martin Roesch.

Développé à l'origine par la société Sourcefire, il est aujourd'hui maintenu par Cisco System à la suite du rachat de Sourcefire en 2013.

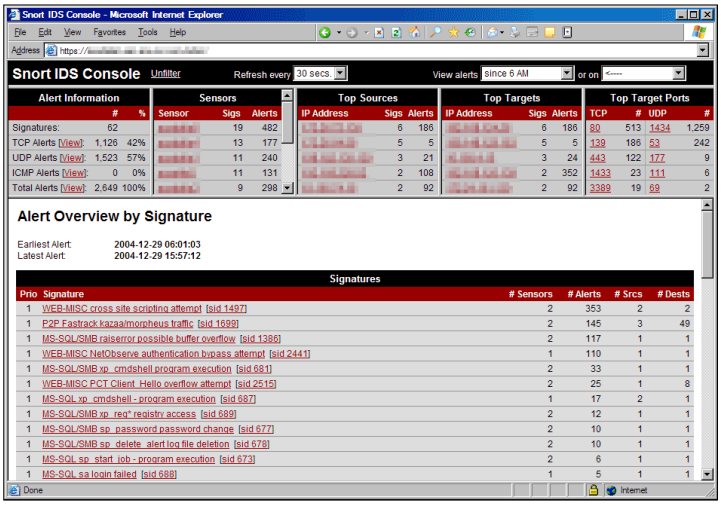

Figure 1. 6: système de détection d'intrusion Snort

- **Fonctionnement**

Le système de détection et de prévention des intrusions (IDS/IPS) Snort a la capacité d'effectuer une analyse du trafic en temps réel et un enregistrement des paquets sur les réseaux IP. Snort effectue l'analyse des protocoles, la recherche et la mise en correspondance des contenus.

Le programme peut également être utilisé pour détecter des sondes ou des attaques, y compris, mais sans s'y limiter, les prises d'empreinte de la pile TCP/IP, les attaques d'URL sémantiques, les dépassements de tampon, les attaques sur SMB et les balayages de ports4.

Snort peut également être utilisé avec d'autres projets open sources tels que SnortSnarf, ACID, sguil et BASE (qui utilise ACID) afin de fournir une représentation visuelle des données concernant les éventuelles intrusions.

3.4.2. Suricata

Suricata est un logiciel open source de détection d'intrusion (IDS), de prévention d'intrusion (IPS), et de supervision de sécurité réseau (NSM). Il est développé par la fondation OISF (Open Information Security Foundation).

Suricata permet l'inspection des Paquets en Profondeur (DPI). De nombreux cas d'utilisations déontologiques peuvent être mis en place permettant notamment la remontée d'informations qualitatives et quantitatives.

Scirius est une interface web sous licence GPLv écrite avec Django destinée à l'édition des règles Suricata4. La version mono-serveur est open source et libre tandis que la version multiserveur Scirius Enterprise est commercialisée, elles sont toutes deux éditées par la société Stamus Network.

Figure 1. 7: logiciel de détection d'intrusion Suricata

➢ **Fonctionnement**

Suricata analyse le trafic sur une ou plusieurs interfaces réseaux en fonction de règles activées. Il génère, par défaut, un fichier JSON. Celui-ci peut être ensuite utilisé par le logiciel de type Extract-transform-load comme par exemple logstash souvent utilisé avec Elasticsearch.

3.4.3. AIDE (Advanced Intrusion Detection Environment)

Advanced Intrusion Detection Environment (AIDE) était initialement développé en tant que logiciel gratuit similaire à Tripwire sous Licence Publique Générale GNU (GPL).

Les principaux développeurs sont Rami Lehti et Pablo Virolainen, tous deux associés à l'Université technologique de Tampere ainsi que Richard van den Berg, un consultant en sécurité néerlandais indépendant. Le projet est utilisé par de nombreux

systèmes type Unix en tant qu'outil peu coûteux de détection de rootkits et de contrôle de référence.

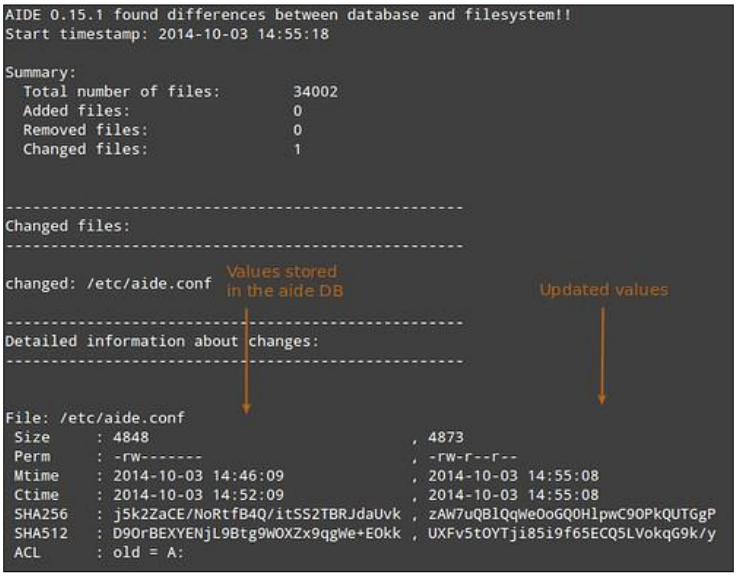

Figure 1. 8: Advanced Intrusion Detection Environment

- **Fonctionnalité**

AIDE prend un « instantané » de l'état du système, enregistre les fragmentations, les moments liés à des modifications et toute autre donnée concernant les fichiers définis par l'administrateur. Cet « instantané » est utilisé pour générer une base de données qui est enregistrée et peut-être être stockée sur un périphérique externe pour plus de sécurité.

Lorsque l'administrateur souhaite exécuter un test d'intégrité, l'administrateur place la base de données précédemment générée en un lieu accessible et commande AIDE afin de comparer la base de données avec l'état réel du système. Toute modification qui se serait produite sur l'ordinateur entre la création de l'instantané et le test sera détectée par AIDE et sera signalée à l'administrateur. AIDE peut être configuré pour s'exécuter de façon planifiée et signaler quotidiennement les changements grâce aux technologies d'ordonnancement comme le cron, qui est le comportement par défaut du package AIDE de Debian.

Ceci est principalement utilisé pour des raisons de sécurité étant donné que toute modification malveillante qui aurait pu se produire au sein du système serait signalée par AIDE.

3.5. Enjeux de la sécurité au sein de l'entreprise

La sécurité de l'entreprise consiste en l'ensemble des stratégies et procédures utilisées pour défendre une organisation contre les acteurs malveillants.

3.5.1. Enterprises utilisant snort

Les entreprises utilisant snort se trouvent le plus souvent aux États-Unis et dans l'industrie des technologies et services de l'information.

Some of the companies that use snort include:

Tableau 1. 2: Entreprises utilisant snort

Entreprise	Site Internet	Pays
California State University-Stanislaus	csustan.edu	United States
NetSuite Inc	netsuite.com	United States
Red Hat Inc	redhat.com	United States
Blackfriars Group	blackfriarsgroup.com	United Kingdom

❖ **Problèmes rencontrés par l'entreprise lors de l'utilisation du système » snort »**

Primo, le risque engendré par un trafic très important qui pourrait entraîner une perte de fiabilité et second, étant situé hors du domaine de protection du firewall, le NIDS est alors exposé à d'éventuelles attaques pouvant le rendre inefficace

3.5.2. Enterprises utilisant Suricata

Nous avons des données sur plusieurs entreprises qui utilisent Snort comme système d'infiltration. Les entreprises qui utilisent snort se trouvent le plus souvent aux États-Unis et dans l'industrie des technologies et des services de l'information. Dans le tableau ci-dessous (tableau 1.3), nous avons fourni exemple des entreprises qui utilise Snort

Tableau 1. 3: Entreprises utilisant Suricata

Entreprise	Site Internet	Pays
Proofpoint	proofpoint.com	Sunnyvale
Inflow NS	inflow-ns.com	San Antonio
AERMOR	aermor.com	Virginia Beach

❖ **Problèmes rencontrés par l'entreprise lors de l'utilisation du système » Suricat »**

Suricat est un peu plus complexe à installer et la communauté est plus petite que ce que Snort a amassé, mais cela pourrait changer. Suricata est développé par l'open information Security Foundation (OISF).

4. Etude de l'existant

Dans nos jours, où quand l'informatique contrôle tout le monde, il existe des nombreux systèmes de détection d'intrusion dans les réseaux. Celle génériques et d'autres spécifiques, notamment

Tableau 1. 4: systèmes de détection d'intrusion

Nom de l'outil	Plate-forme	Type d'IDS	Nos notes	Fonctionnalités
Copain	Unix, Linux, Mac-OS	NIDS	4/5	Journalisation et analyse du trafic, Fournit une visibilité sur les paquets, le moteur d'événements, Scripts de politique, Possibilité de surveiller le trafic SNMP, Possibilité de suivre l'activité FTP, DNS et HTTP.
OSSEC	Unix, Linux, Windows, Mac-OS	HIDS	4/5	Libre d'utiliser la sécurité HIDS open source, Possibilité de détecter toute modification du registre sous Windows, Possibilité de surveiller toutes les tentatives d'accès au compte root sur Mac-OS, Les fichiers journaux couverts incluent les données de messagerie, FTP et de serveur Web.
Renifler	Unix, Linux, Windows	NIDS	5/5	Renifleur de paquets, Enregistreur de paquets, Threat intelligence, blocage de signature, Mises à jour en temps réel des signatures de sécurité,Reporting approfondi, Capacité à détecter une variété

				d'événements, y compris les empreintes digitales du système d'exploitation, les sondes SMB, les attaques CGI, les attaques par débordement de tampon et les analyses de ports furtifs.
Suricata	Unix, Linux, Windows, Mac-OS	NIDS	4/5	Collecte des données au niveau de la couche application, Possibilité de surveiller l'activité du protocole à des niveaux inférieurs tels que TCP, IP, UDP, ICMP et TLS, suivi en temps réel pour les applications réseau telles que SMB, HTTP et FTP, L'intégration avec des outils tiers tels que Anaval, Squil, BASE et Snorby, module de script intégré, utilise à la fois des méthodes basées sur les signatures et les anomalies, Architecture de traitement intelligente.
Oignon de sécurité	Linux, Mac-OS	HIDS, NIDS	4/5	Distribution Linux complète avec un accent sur la gestion des journaux, Surveillance de la sécurité d'entreprise et détection des intrusions, s'exécute sur Ubuntu, intègre des éléments de plusieurs outils d'analyse et frontaux, notamment Network Miner, Snorby, Xplico, Sguil, ELSA et Kibana, Comprend également des fonctions HIDS, un renifleur de paquets effectue une analyse de réseau Comprend de jolis graphiques et tableaux.

4.1. Critique de l'existant

Par nature, les systèmes de détection d'intrusion Classique comme représente le tableau 1.4 doivent mettre leur carte réseau en mode "promiscuous" ce qui va leur

permettre de recevoir l'intégralité des trames circulant sur le réseau. Ainsi, l'IDS ne générera généralement aucun trafic et se contentera d'aspirer tous les paquets.

4.2. Défis de la détection d'intrusions

Dans ce qui suit, on présente les principaux problèmes auxquels font face les détecteurs d'intrusions.

4.2.1. Prise en charge de l'aspect temporel

L'un des principaux problèmes dont souffrent les systèmes de détection, est effectivement J'aspect temporel dans les spécifications logiques des attaques ou des comportements, selon l'approche d'analyse choisie. La plupart du temps, les règles de détection d'intrusions définissent des contraintes sur des contenus indépendamment du temps. [4]

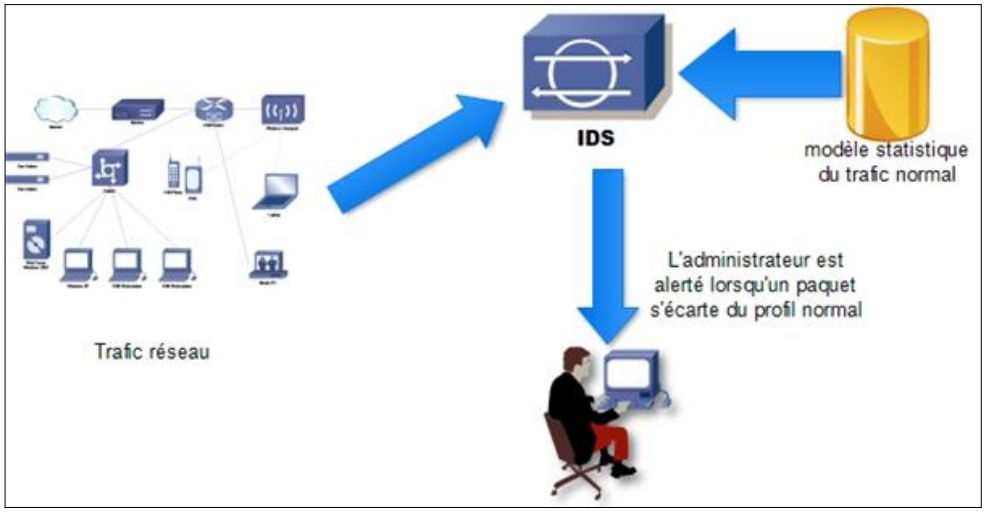

Figure 1. 9: Système de détection

4.2.2. Multiplication des alertes

La multiplication des alertes est j'un des principaux problèmes des détecteurs d'intrusions. En effet, si un administrateur du système informatique se trouve noyé dans des journaux d'alertes et qu'en plus il constate que les alertes signalées ne sont pas toutes pertinentes et utiles, il finira par ne plus prendre au sérieux l'outil de détection d'intrusions. [5]

La multiplication des alertes peut être la conséquence d'une ou de plusieurs caractéristiques des attaques, de l'architecture de la solution de détection ou encore de la méthode d'analyse. Benjamin Morin a défini trois principales sources des multiplications : la récurrence, la redondance et la division.

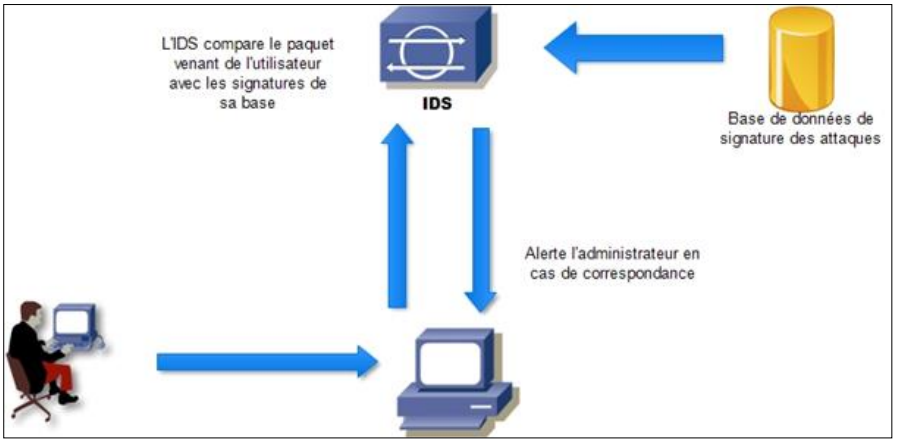

Figure 1. 10: système de détection multiplication des alertes

5. Conclusion

Au cours de ce chapitre nous avons discuté les exigences et les problèmes à résoudre de point de vue des besoins de l'utilisateur. Nous avons présenté le système de detection d'intrusion et les techniques utilisées. Cette phase nous permettant de préparer la technologie de travail qui est notre but dans le chapitre suivant.

Chapitre 2 :

Les réseaux de neurones

1. Introduction

Dans ce chapitre Nous allons parler sur l'intelligence artificielle, Ces applications dans la vie quotidienne, par la suite nous allons présenter quelques approches pour la prédiction d'intrusion. Enfin Nous allons introduit un concept d'apprentissage automatique.

2. L`apprentissage profond (deeplearning)

L'apprentissage en profondeur fait partie d'une famille de méthodes d'apprentissage automatique basées sur des modèles de données d'apprentissage, Dans cette partie nous présenterons l'apprentissage en profondeur et leur fonctionnement.

2.1. Définition

L'apprentissage profond est un groupe de techniques d'apprentissage automatique qui ont permis des avancées significatives en intelligence artificielle ces dernières années.

Dans l'apprentissage automatique, un logiciel analyse un ensemble des données pour établir des règles permettant de tirer des conclusions sur des nouvelles données.

L'apprentissage en profondeur est basé sur ce que l'on appelle, par analogie, des "réseaux de neurones artificiels", constitués des milliers d'unités ("neurones") effectuant chacune des petites opérations simples. Les résultats de la première couche de « neurones » sont utilisés comme entrées dans les calculs de la deuxième couche, et ainsi de suite.

Par exemple, pour la reconnaissance visuelle, les premières couches d'unités définissent des lignes, des courbes, des angles... les couches supérieures définissent des formes, des groupes de formes, des objets, des contextes... Les avancées en deeplearning sont rendues possibles notamment grâce à la montée en puissance des ordinateurs et le développement de bases de données massives (« big data »).

2.1.1. Fonctionnement du deep Learning

L'apprentissage profond est basé sur un réseau de neurones artificiels inspirés du cerveau humain. Ce réseau est constitué de dizaines voire de centaines de « couches » de neurones, dont chacune reçoit et interprète les informations de la couche précédente. Par exemple, le système apprendra à reconnaître les lettres avant de traiter les mots dans le texte, ou à déterminer s'il y a un visage dans une image avant de connaître son identité.

2.1.2. Domaines d'application

Le deep Learning est utilisé dans de nombreux domaines :

- ✓ Reconnaissance d'image,
- ✓ Traduction automatique,
- ✓ Voiture autonome,
- ✓ Diagnostic médical,
- ✓ Recommandations personnalisées,
- ✓ Modération automatique des réseaux sociaux,
- ✓ Prédiction financière et trading automatisé,
- ✓ Identification de pièces défectueuses,
- ✓ Détection de malwares ou de fraudes,
- ✓ Chatbots (agents conversationnels),
- ✓ Exploration spatiale,
- ✓ Robots intelligents.

3. Etude technologique

Une étude du projet commence toujours par la collecte d'informations qui pourraient faciliter l'imagination de la solution technologique. L'utilité de créer un agent conversationnel, c'est qu'il va essayer de comprendre l'intention et d'y répondre. C'est une discussion entre humain et machine, le créateur toujours aimerait rendre la machine aussi vivante qu'un être humain, c'est le concept de l'intelligence Artificielle. Et Pour parler d'une détection d'intrusion intelligent basé sur le Deep Learning il est nécessaire de définir les grands axes où se déroule notre projet.

3.1. Intelligence artificielle

L'intelligence artificielle (IA) est un processus d'imitation de l'intelligence humaine qui repose sur la création et l'application d'algorithmes exécutés dans un environnement informatique dynamique. Son but est de permettre à des ordinateurs de penser et d'agir comme des êtres humains. Tout simplement on peut dire que l'intelligence artificielle c'est une machine intelligente qui simule la pensée ainsi le comportement de l'être humain.

Pour y parvenir, trois composants sont nécessaires :

- Des systèmes informatiques
- Des données avec des systèmes de gestion

- Des algorithmes d'IA avancés (code)

➢ **Domaine d'application**

Les domaines d'application de l'intelligence artificielle sont nombreux. Il est présent dans les caméras des smartphones. Le tableau ci-dessous représente les différents domaines d'application :

Tableau 2. 1 : Les domaines d'application de l'intelligence artificielle

Domaines d'application	**Technologies**
Les services bancaires	Les technologies financières : Fintech
Les transports	La révolution des véhicules autonomes.
La distribution	Plateformes intelligentes (Amazon).
La médecine	Intervention chirurgicale à distance. La médecine prédictive. La lutte contre le Covid-19 (désinfection, accompagnement des malades, dépistage, récolte de données, la recherche de médicaments).
Les villes	La « smartization » des villes.
Les usines	Usines du futur 4.0 Robots.
Les télécommunications	Les réseaux deviennent apprenants avec une capacité d'amélioration en continu.
Les Aéroports	Fournir des données fiables sur les mouvements des avions. Améliorer l'expérience client et optimiser les opérations.

3.2. Machine Learning

Machine Learning : Appelé aussi apprentissage automatique. C'est une méthode qu'utilise l'Intelligence Artificielle en se fondant sur des approches statistiques pour donner aux machines la capacité d'apprendre à partir de données provenant du big data, c'est à dire de s'améliorer en ayant la capacité à résoudre des tâches sont être explicitement programmées.

L'apprentissage automatique, également appelé apprentissage machine ou apprentissage artificiel et en anglais machine Learning, est une forme d'intelligence artificielle (IA) qui permet à un système d'apprendre à partir des données et non à l'aide d'une programmation explicite. Cependant, l'apprentissage automatique n'est pas un processus simple. Au fur et à mesure que les algorithmes ingèrent les données de formation, il devient possible de créer des modèles plus précis basés sur ces données. Un modèle de machine Learning est le résultat généré lorsque vous entraînez votre algorithme

d'apprentissage automatique avec des données. Après la formation, lorsque vous fournissez des données en entrée à un modèle, vous recevez un résultat en sortie. Par exemple, un algorithme prédictif crée un modèle prédictif. Ensuite, lorsque vous fournissez des données au modèle prédictif, vous recevez une prévision qui est déterminée par les données qui ont servi à former le modèle.

- **Etude de quelques approches pour la prédiction d'intrusion**

Dans un contexte général, Un grand nombre de classificateurs peut être utilisé pour identifier les individus entre eux. Plusieurs réseaux à apprentissage profond pré-entraînés peuvent être employés afin d'obtenir des performances remarquables sans pour autant nécessiter un long processus d'apprentissage ou de conception de système spécialisé à une situation particulière. Quelques exemples : DeepFace, FaceNet et TBE-CNN et HaarNet.

Il existe trois types importants de réseaux de neurones qui constituent la base de la plupart des modèles pré-entraînés en apprentissage en profondeur :

- **Classification naïve bayésienne**

La classification bayésienne naïve est un type de classification probabiliste bayésienne simple basée sur le théorème bayésien avec une forte indépendance des hypothèses. Il implémente un classificateur Bayes naïf, ou un classificateur Bayes naïf, qui appartient à la famille des classificateurs linéaires.

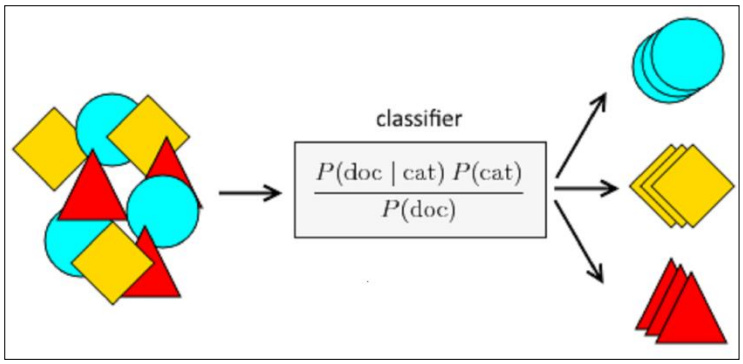

Figure 2. 1: Classification naïve bayésienne[11]

- **K Nearst Neighbors (K-NN)**

L'algorithme K-NN (K-nearest Neighbors) est une méthode d'apprentissage supervisé. Il peut être utilisé aussi bien pour la régression que pour la classification. Son fonctionnement peut être assimilé à l'analogie suivante.

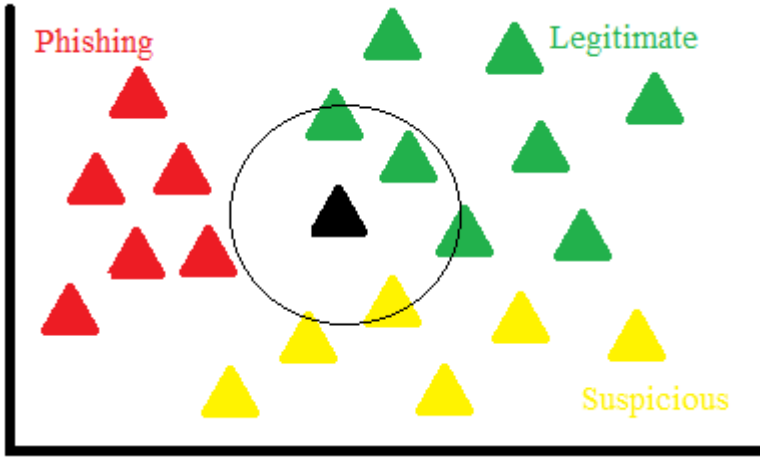

Figure 2. 2: K Nearst Neighbors (K-NN)[12]

L'algorithme K-NN va se baser sur le jeu de données en entier. En effet, pour une observation, qui ne fait pas parti du jeu de données, qu'on souhaite prédire, l'algorithme va chercher les K instances du jeu de données les plus proches de notre observation. Ensuite pour ces voisins, l'algorithme se basera sur leurs variables de sortie (output variable) pour calculer la valeur de la variable de l'observation qu'on souhaite prédire.

Par ailleurs :

- Si K-NN est utilisé pour la régression, c'est la moyenne (ou la médiane) des variables des plus proches observations qui servira pour la prédiction
- Si K-NN est utilisé pour la classification, c'est le mode des variables des plus proches observations qui servira pour la prédiction
- **Arbre de décision**

Un arbre de décision est un outil d'aide à la décision représentant un ensemble de choix sous la forme graphique d'un arbre. Les différentes décisions possibles sont situées aux extrémités des branches, et sont atteintes en fonction de décisions prises à chaque étape [6].

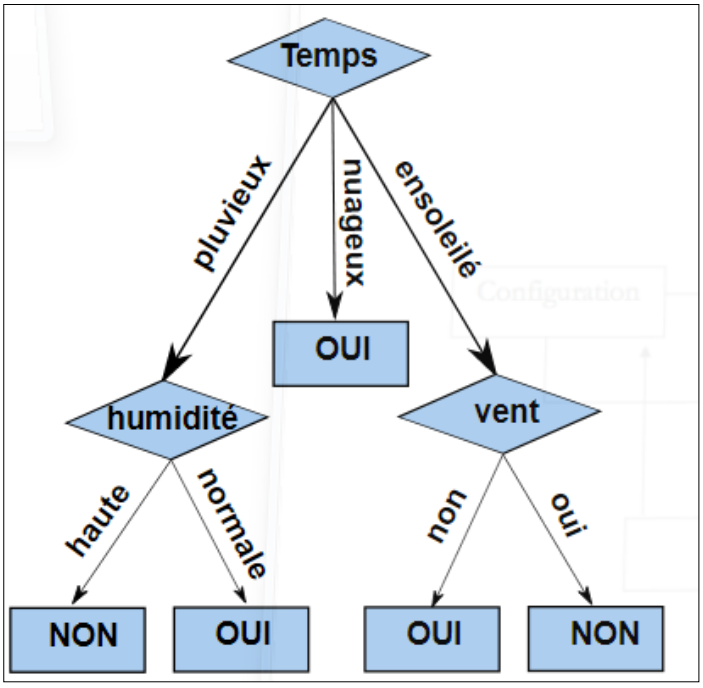

Figure 2. 3: Arbre de décision[13]

- **Artificial Neural Network (ANN)**

Les ANN ont été développés comme une généralisation des modèles mathématiques de la biologie de la cognition humaine. Cependant, le super-ordinateur le plus puissant d'aujourd'hui ne peut rivaliser avec un cerveau humain en termes de connectivité et de complexité. Par conséquent, un ANN est considéré comme un mode de système neuronal biologique extrêmement simplifié

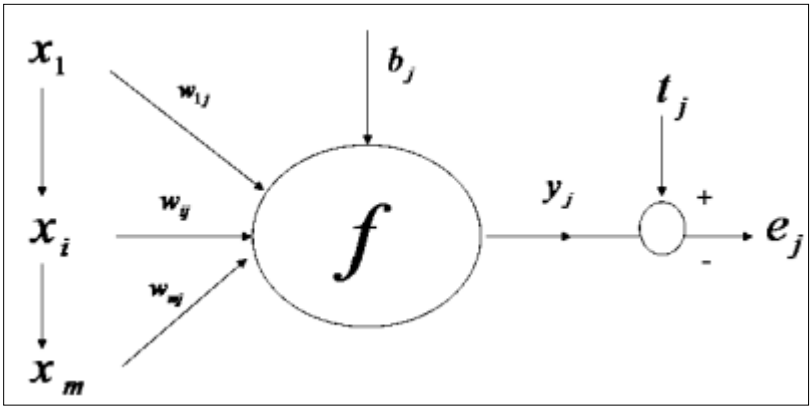

Figure 1. 11: Modèle mathématique d'un neurone ANN [14]

- **Fonctionnement d'ANN**

Tout d'abord, les informations sont introduites dans la couche d'entrée. Ce qui le transfère ensuite aux couches cachées, et l'interconnexion entre ces deux couches attribue des poids à chaque entrée de manière aléatoire au point initial. Ensuite, le biais est ajouté à

chaque neurone d'entrée et après cela, la somme des poids qui est une combinaison de poids et de biais passe par la fonction d'activation. La fonction d'activation a la responsabilité du nœud à déclencher pour l'extraction des caractéristiques et enfin la sortie est calculée. Par conséquent, tout ce processus est connu sous le nom de propagation vers l'avant. Après avoir obtenu le modèle de sortie pour le comparer avec la sortie d'origine et l'erreur est connue et enfin, les poids sont mis à jour dans la propagation vers l'arrière pour réduire l'erreur et ce processus se poursuit pendant un certain nombre d'époques (itération). Enfin, les poids du modèle sont mis à jour et la prédiction est effectuée.

- **Approche CNN**

Un CNN (**Figure 2.4**) demande moins de prétraitement que d'autres méthodes. Cette technique est très utilisée pour la reconnaissance et la classification d'images. Comparativement aux RNN, un CNN est plus rapide car, lorsque le RNN calcule de façon séquentielle les mots d'une phrase, le CNN le fait simultanément. Il est donc possible d'obtenir des résultats similaires en abaissant les contraintes calculatoires, ce qui en fait un modèle intéressant pour le TAL. Il semble l'être particulièrement pour les tâches de classification [7].

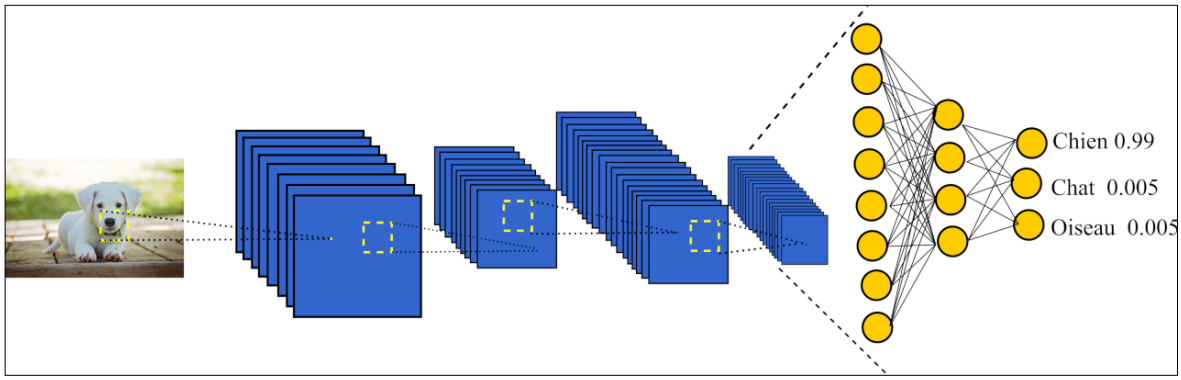

Figure 2. 4 : Architecture du réseau convolutif [7]

3.3. Comparaison des diffèrent approche

La différence entre les différentes méthodes d'apprentissage représente dans la table ci-dessous :

Tableau 2. 2 : Comparaison des différentes méthodes d'approche

	METHODOLOGIE	**UTILISATION**
KPPV	Non paramétrique Distance euclidienne	Estimation Le pourcentage de prédiction
ANN	Support Vector Machine Classifieur linéaire	Discrimination La reconnaissance des chiffres manuscrits Régression.
RNN	Le réseau de neurones Algorithmes mathématique	Identification des modèles et d'images Reconnaissance vocale Traduction automatique…
CNN	Le réseau de neurones Apprentissage automatique	Reconnaissance et le traitement des images. Corrélation locale des données Détections de visages très complexes……

3.4. Deep Learning

Le DeepLearning ou apprentissage profond est un type d'intelligence artificielle dérivé de la machine Learning (apprentissage automatique) où la machine est capable d'apprendre par elle-même, contrairement à la programmation où elle se contente d'exécuter à la lettre des règles prédéterminées.

L'apprentissage profond s'appuie sur un réseau de neurones artificiels s'inspirant du cerveau humain. Ce réseau est composé de dizaines voire de centaines de « couches » de neurones, chacune recevant et interprétant les informations de la couche précédente. Le système apprendra par exemple à reconnaître les lettres avant de s'attaquer aux mots dans un texte, ou détermine s'il y a un visage sur une photo avant de découvrir de personne il s'agit.

Ces trois domaines peuvent être vus comme étant imbriqués mais ils ne sont pas au même niveau comme il indique cette figure :

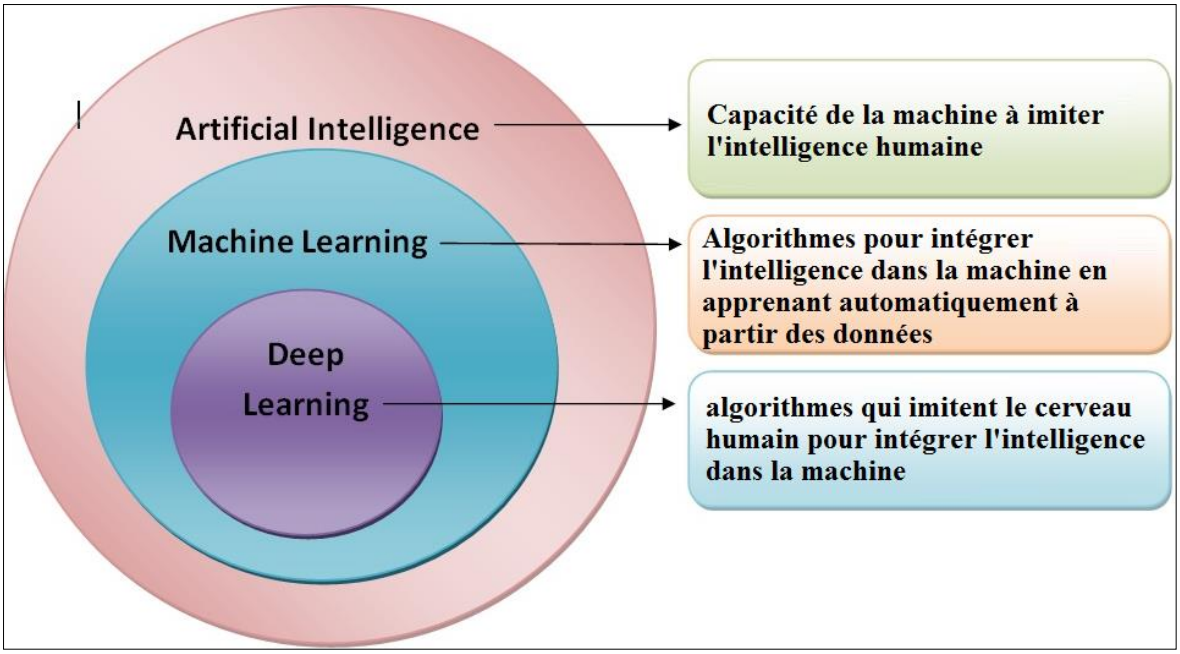

Figure 2. 5: les différences entre IA, ML et DL

4. Présentation des réseaux de neurones convolutifs (CNN)

Pour bien comprendre les réseaux de neurones convolutifs, il est important de connaître les bases des réseaux de neurones, dans cette partie nous allons présentera l'architecture de réseaux de neurones convolutifs et les différentes couches.

4.1. L'architecture des réseaux de neurones convolutifs (CNN)

Les réseaux convolutifs (CNN/ConvNets) sont particulièrement utiles pour identifier des objets, des visages et des scènes dans les images afin de les reconnaître. Ils apprennent directement à partir de données d'images et utilisent des modèles pour classer l'image.

Les réseaux convolutifs ont généralement plusieurs couches cachées qui, en général, sont organisées en blocs (CONV→ ReLU → POOL). Mais certains réseaux convolutifs ont des centaines de couches cachées. Les calques masqués plus proches du calque d'entrée détectent des caractéristiques très simples telles que les bords et les dégradés de couleurs dans les images. Ensuite, les couches supérieures combinent ces caractéristiques simples dans des motifs plus complexes. Enfin, les couches proches de la sortie combinent ces motifs plus complexes.

Dans cette section, nous allons expliquer les étapes de fonctionnement de réseau convolutif (CNN/ConvNets) suivant son architecture, ainsi que présenter les étapes de l'analyse faite par cet algorithme.

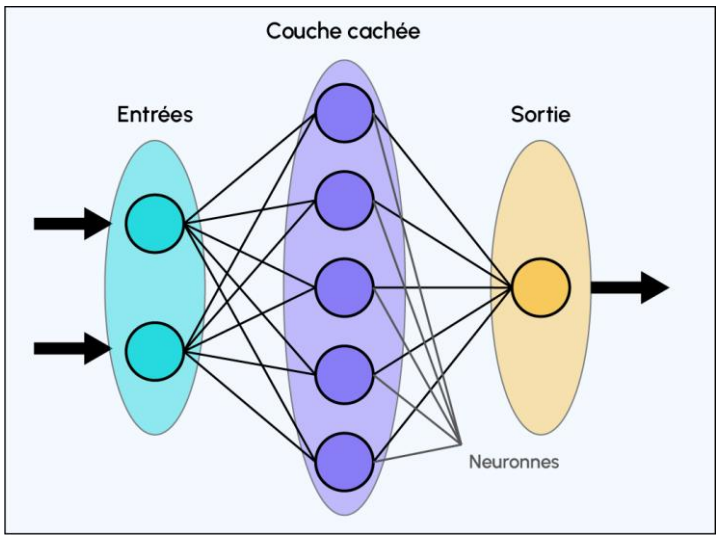

Figure 2. 6: CNN classique [15]

Ici nous pouvons voir chaque pile de couches et comment chaque neurone ne traite qu'un patch de son entrée. La taille de l'image continue à diminuer jusqu'à atteindre les couches entièrement connectées

4.2. Principe général de CNN

Les CNN sont utiles pour les applications basées sur la reconnaissance d'objets et le computer vision, comme les véhicules autonomes ou la reconnaissance faciale.

L'utilisation des CNN pour le Deep Learning s'est répandue pour trois raisons importantes :

- ✓ Ils éliminent la nécessité d'effectuer une extraction manuelle des caractéristiques, car ils les apprennent directement.
- ✓ Ils produisent d'excellents résultats de reconnaissance.
- ✓ Ils peuvent être ré entraînés pour effectuer de nouvelles tâches de reconnaissance, ce qui vous permet de vous appuyer sur des réseaux préexistants.

Les éléments de base de l'architecture de CNN sont :

2.1.1. La couche de convolution

La couche de convolution est parfois appelée couche d'extraction de Caractéristiques, car les caractéristiques de l'image sont extraites dans cette couche. La

convolution est une opération mathématique comme l'addition et la multiplication, il est très utile de simplifier des équations plus complexes, cette opération est largement utilisée dans le traitement du signal numérique. Lorsque l'on applique la convolution au traitement d'image, on réalise la convolution (combiner) l'image d'entrée avec une sous-région de cette image (filtre). Le filtre est aussi connu sous le nom du noyau de convolution, il consiste en des poids de cette sous-région

Supposons que nous avons à l'entrée la matrice suivante :

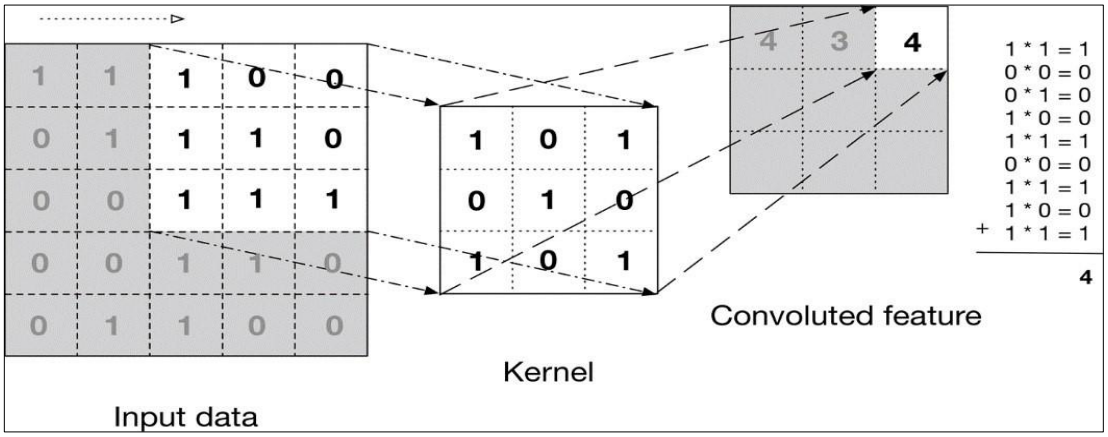

Figure 2. 7: l'opération de convolution[16]

Ainsi, on applique un filtre qui sert à faire ressortir certaines caractéristiques d'une image donnée (couleur, contour, luminosité, netteté, etc.…). Ce filtre va être déplacé par pas successifs sur l'ensemble d'image.

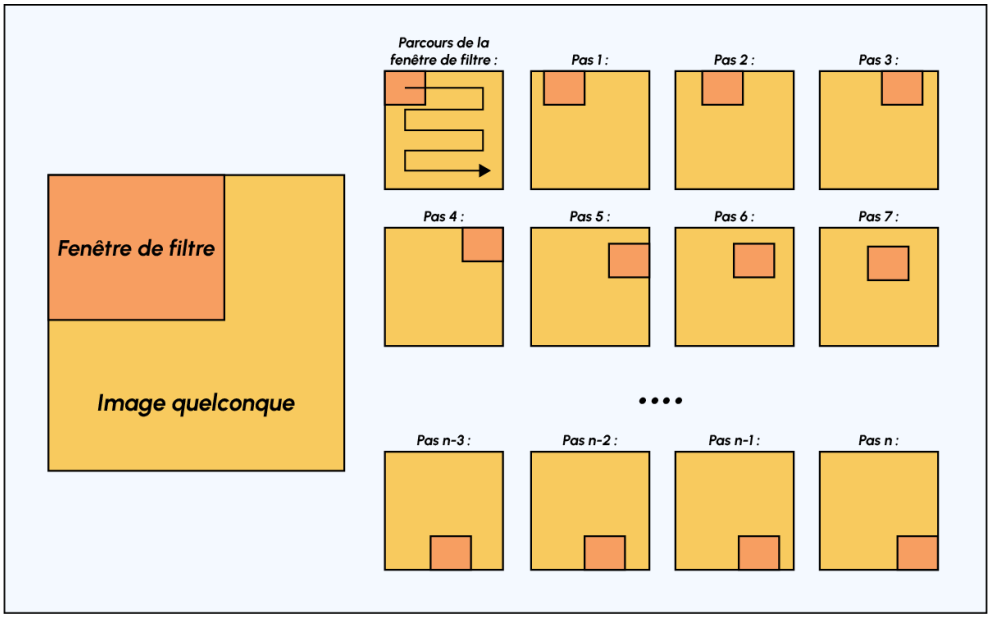

Figure 2. 8: Parcours de la fenêtre de filtre sur l'image[17]

Pour chaque position du filtre, les valeurs des deux matrices en superposition (filtre et image à traiter) sont multipliées. Chaque valeur ainsi inférée est projetée dans une nouvelle matrice. Cette matrice représente une nouvelle image qui fait ressortir les caractéristiques (featur emap) recherchées au travers du filtre.

2.1.2. Couche d'activation

La couche de convolution génère une matrice de taille beaucoup plus petite que l'image d'origine. Cette matrice est exécutée à travers une couche d'activation, qui introduit une non-linéarité pour permettre au réseau de se former via une rétro propagation. Les fonctions d'activation couramment utilisées sont la fonction Relu (***Rectified Linear Units)***. Les fonctions d'activation modernes normalisent la sortie à une plage donnée, pour garantir une convergence stable du modèle.

2.1.3. Couche de pooling

Ce type de couche est souvent placé entre deux couches de convolution. L'opération de pooling consiste à réduire la taille des images, tout en préservant leurs caractéristiques importantes. Pour cela, on découpe l'image en cellules régulière, puis on garde au sein de chaque cellule la valeur maximale.

En pratique, on utilise souvent des cellules carrées de petite taille pour ne pas perdre trop d'informations. Les choix les plus communs sont des cellules adjacentes de taille 2 ×2 pixels qui ne se chevauchent pas, ou des cellules de taille 3 × 3 pixels, distantes les unes des autres d'un pas de 2 pixels (qui se chevauchent donc).

La couche de pooling permet de réduire le nombre de paramètres et de calculs dans le réseau. On améliore ainsi l'efficacité du réseau et on évite le sur-apprentissage.

Il existe différents types de couches de pooling : max-pooling, average-pooling, etc. Le plus courant et le plus utilisé est le max-pooling.

Le modèle de réseau neuronal convolutif (CNN) contient souvent une couche de max-pooling dans une application de détection d'objets et de classification d'images. Le max-pooling réduit la dimension de l'image en extrayant la valeur la plus élevée dans une région identifiée par le filtre de regroupement maximal. Le but de l'utilisation de l'opération de max-pooling est de réduire le nombre de paramètres dans le modèle et de conserver les caractéristiques essentielles d'une image. Moins de paramètres diminuent la complexité du modèle et son temps de calcul.

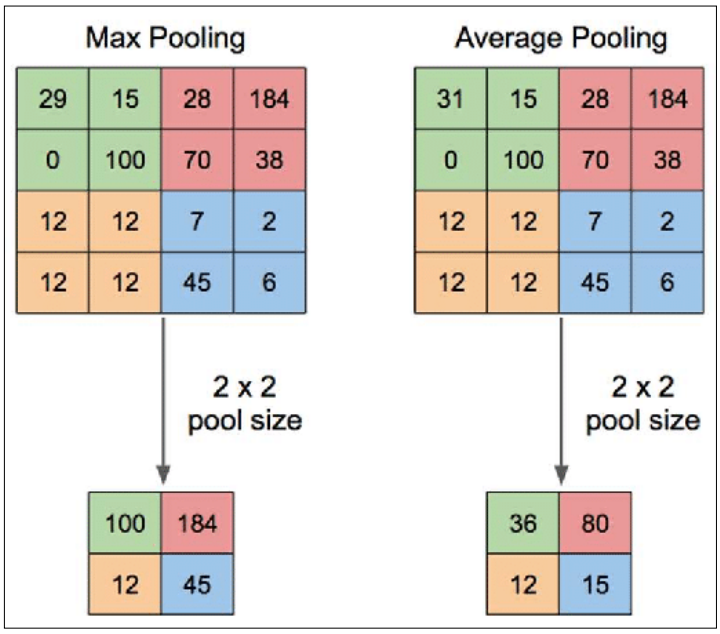

Figure 2. 9: Opération de Max-pooling avec filtre 2x2.[18]

2.1.4. Couche fully-connected

Constitue toujours la dernière couche d'un réseau de neurones, convolutif ou non. Son entrée est un vecteur unidimensionnel représentant la sortie des couches précédentes. Sa sortie est une liste de probabilités pour différentes étiquettes possibles attachées à l'image (par exemple voiture, personne…). L'étiquette qui reçoit la probabilité la plus élevée est la décision de détection.

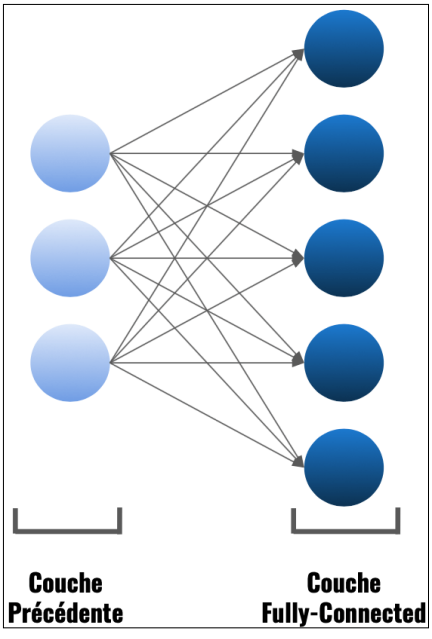

Figure 2. 10: Couche Fully-Connected

5. Systèmes de détection d'intrusion basés sur IA

Les systèmes de détection d'intrusion reposent sur un ensemble de mécanismes et d'algorithmes permettant de détecter, de manière optimale, des intrusions ou menaces dans un réseau informatique. L'apprentissage automatique peut être utilisé pour améliorer de manière significative la fiabilité de détection d'intrusion.

5.1. Obtention des données

La première étape est l'obtention des données. Lors de la phase d'apprentissage, ces informations permettent de connaître les habitudes des utilisateurs ou les différents types d'attaques de manière générale. Lors de la phase d'exécution, ils permettent de détecter une attaque. Néanmoins, la question est de savoir s'il existe une mesure de la quantité de données nécessaire pour avoir un modèle correct, c'est-à-dire qui respecte un certain taux d'erreur accepté. Cette question est encore ouverte à ce jour. Toutefois, le choix des informations à prendre en compte dépendra du type d'apprentissage.

5.2. Supervisé ou non supervisé

Les apprentissages supervisés utilisent généralement des data sets représentant des attaques connues. Les apprentissages non supervisés se basent uniquement sur les comportements des utilisateurs pour détecter une variation par rapport aux comportements normaux qui représentent une attaque.

Les algorithmes supervisés réalisent d'excellents résultats pour des intrusions connues, ils sont meilleurs que les algorithmes non supervisés. Inversement, pour des agressions inconnues, les algorithmes supervisés voient une réduction drastique de leur efficacité contrairement aux algorithmes non supervisés. Ceci peut s'expliquer par le fait que, puisque les algorithmes non supervisés ne font que partitionner des données, les attaques leur sont toujours inconnues. En effet, étant donné que ces derniers ne font que regrouper des données proches, ils ne savent pas quand elles représentent une attaque. Bien que les algorithmes supervisés et non supervisés obtiennent des résultats semblables pour des attaques inconnues, les modèles non supervisés sont préférés pour leur robustesse. En effet, ceux-ci ne changent pas drastiquement leur taux de réussite selon que l'intrusion est connue ou non.

5.3. Hypothèse

Dans le cas des algorithmes supervisés, une hypothèse difficile à respecter lors de l'apprentissage est qu'il n'y a pas d'attaque contenue dans les informations modélisant le comportement normal des programmes qui peuvent pourtant avoir beaucoup d'irrégularités. En effet, dans le cas contraire, on insérerait des propriétés d'une attaque comme comportement normal. Par conséquent, cette dernière ne sera pas détectée. Inversement, un algorithme non supervisé peut lever cette hypothèse en nettoyant les données pour enlever les informations d'attaque. Pour cela, on recherche un ensemble d'enchaînement, appelé motif, de système call fortement présent dans le système et on les range suivant leur dangerosité d'attaquer le système [9].

5.4. Implémentation

Une méthode généralement utilisée, qui est basée sur l'approche comportementale, est la prise d'empreintes des utilisateurs, c'est-à-dire de leur comportement, et de regarder quand elle ne lui correspond pas sur le système. Ainsi, on peut détecter un comportement anormal et donc une attaque éventuelle. Inversement, on peut prendre l'empreinte de certains pirates connus pour les détecter. Cette dernière peut être apprise par la machine Learning. Toutefois, une autre méthode, basée sur l'approche par scénario, est l'utilisation de données représentant des attaques. Pour que cela soit réaliste, il faut que l'IDS soit suffisamment rapide, efficace et flexible aux petits changements normaux des utilisateurs, sans toutefois permettre de dévier vers une situation d'attaque. Dans le cas contraire, soit elle ne sera pas détectée soit le nombre de faux positifs pourrait exponentiellement augmenter.

De manière globale, on réalise un tel IDS en trois étapes :

- On modélise tous les comportements normaux de chaque utilisateur ou les signatures des attaques,
- On le donne au machine Learning pour qu'il l'apprenne et ensuite on regarde si le comportement dévie de l'habituel ou s'approche d'une situation offensive. Dans certains cas, il est intéressant de savoir le type de l'attaque et non seulement si elle a eu lieu [10].
- Une dernière implémentation est la gestion d'un grand nombre d'alertes venant des IDS par du machine Learning. Ainsi, cette méthode est une sorte de filtre de ces dernières permettant de se focaliser sur les alarmes les plus importantes. En effet, un IDS peut

générer un nombre volumineux de fausses alertes, ce qui rend la tâche des administrateurs système impossible. Ceci est donc un complément aux IDS et non un remplacement de ceux-ci. Comme vu précédemment, il existe des NIDS et des HIDS.

6. Solution proposée et défit

Pour faire face aux problèmes qui existent, nous sommes besoin d'un 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 une technique de détection basées sur des approches d'apprentissage profond pour détecter les intrusions dans les connexions réseau.

A ce niveau nous proposons une solution améliorée et fiable qui prend en compte tous les exigences actuelles de la technologie et qui permettra de répondre aux futures besoins des administrateurs.

7. Conclusion

Dans ce chapitre, nous avons présenté l'architecture des réseaux de neurones convolutifs leur principe et les Différents types de filtres à convolutions, par la suite nous avons expliqué Apprentissage des réseaux de neurones convolutifs et dans le chapitre suivant intitulé « Les réseaux de neurones convolutifs » on va présenter qu'est-ce qu'un réseau de neurones convolutifs.

Chapitre 3 :

Détection d'intrusion

1. Introduction

Nous avons vu dans le chapitre précédant quelques généralités sur les techniques de la détection d'intrusion.

Dans ce chapitre nous allons présenter un système complet, alors je veux commencer de la partie software. Ainsi, nous passons à la configuration et installation les diffèrent bibliothèques. Ensuite nous allons présenter la méthode SVM qui utilisées pour la détection d'intrusion. A la fin, on présentera des tests de classification d'intrusion.

2. Environnement de travail

Pour réaliser notre travail nous avons besoin d'un éditeur de développement python, nous avons choisir la plateforme web google Colab

2.1. Google Colab

Google Colab ouColaboratory est un service cloud, offert par Google (gratuit), basé sur Jupyter Notebook et destiné à la formation et à la recherche dans l'apprentissage automatique. Cette plateforme permet d'entraîner des modèles de Machine Learning directement dans le cloud. Sans donc avoir besoin d'installer quoi que ce soit sur notre ordinateur à l'exception d'un navigateur.

Figure3. 1: Logo google Colab

2.2. Python

Python est un langage de programmation open source créé par le programmeur Guido van Rossum en 1991. ... Il s'agit d'un langage de programmation interprété, qui ne nécessite donc pas d'être compilé pour fonctionner. Un programme " interpréteur " permet d'exécuter le code Python sur n'importe quel ordinateur.

Figure3. 2: Logo python

3. Outils et bibliothèques utilisés

Pour réaliser notre travail nous avons besoin d'installer les d'efférents bibliothèques suivantes :

3.1. L'environnement matériel

Le Deep Learning est un domaine avec des exigences en calculs intenses et la disponibilité des ressources (surtout en GPU) dédiés à cette tache vont fondamentalement influencer sur l'expérience de l'utilisateur car sans ses ressources, il faudra trop de temps pour apprendre de ses erreurs ce qui peut être décourageant.

Les expérimentations ont tous été effectuées sur une machine qui offre des performances acceptables dont voici les caractéristiques :

Tableau 3. 1:caractéristique de machine

CPU	**Intel Core i5-4440 (3.1 GHz)**
GPU	**MSI GEFORCE GTX 770 TWIN FROZR GAMING OC 2GB**
RAM	**8GB**

3.2. Numpy

Numpy est une bibliothèque numérique apportant le support efficace de larges tableaux multidimensionnels, et de routines mathématiques de haut niveau (fonctions spéciales, algèbre linéaire, statistiques, etc.).

Figure3. 3: Bibliothèque Numpy

3.3. Seaborn

Seaborn est une bibliothèque de visualisation Python basée sur matplotlib. Il fournit une interface de haut niveau pour dessiner des graphiques statistiques attrayants.

Figure3. 4:Bibliothèque Seaborn

3.4. Matplotlib

Matplotlib est une bibliothèque du langage de programmation Python destinée à tracer et visualiser des données sous formes de graphiques. Elle peut être combinée avec les bibliothèques python de calcul scientifique NumPy et SciPy. Matplotlib est distribuée librement et gratuitement sous une licence de style BSD.

Figure3. 5: Logo Matplotlib

3.5. Pandas

Pandas est une bibliothèque écrite pour le langage de programmation Python pour le traitement et l'analyse de données. En particulier, il fournit des structures de données et des opérations pour manipuler des matrices scalaires et des séries temporelles. Pandas est un logiciel libre sous licence BSD.

Figure3. 6: logo de bibliothèque pandas

4. Modèle proposé

Notre sujet traite la classification des connexions TCP/IP pour la détection d'intrusions pour cela et Comme pour tous modèles de classification, l'élaboration de notre modèle respecte les phases principales suivantes :

Le prétraitement, l'apprentissage et la phase de test. Comme l'indique le diagramme ci-dessous :

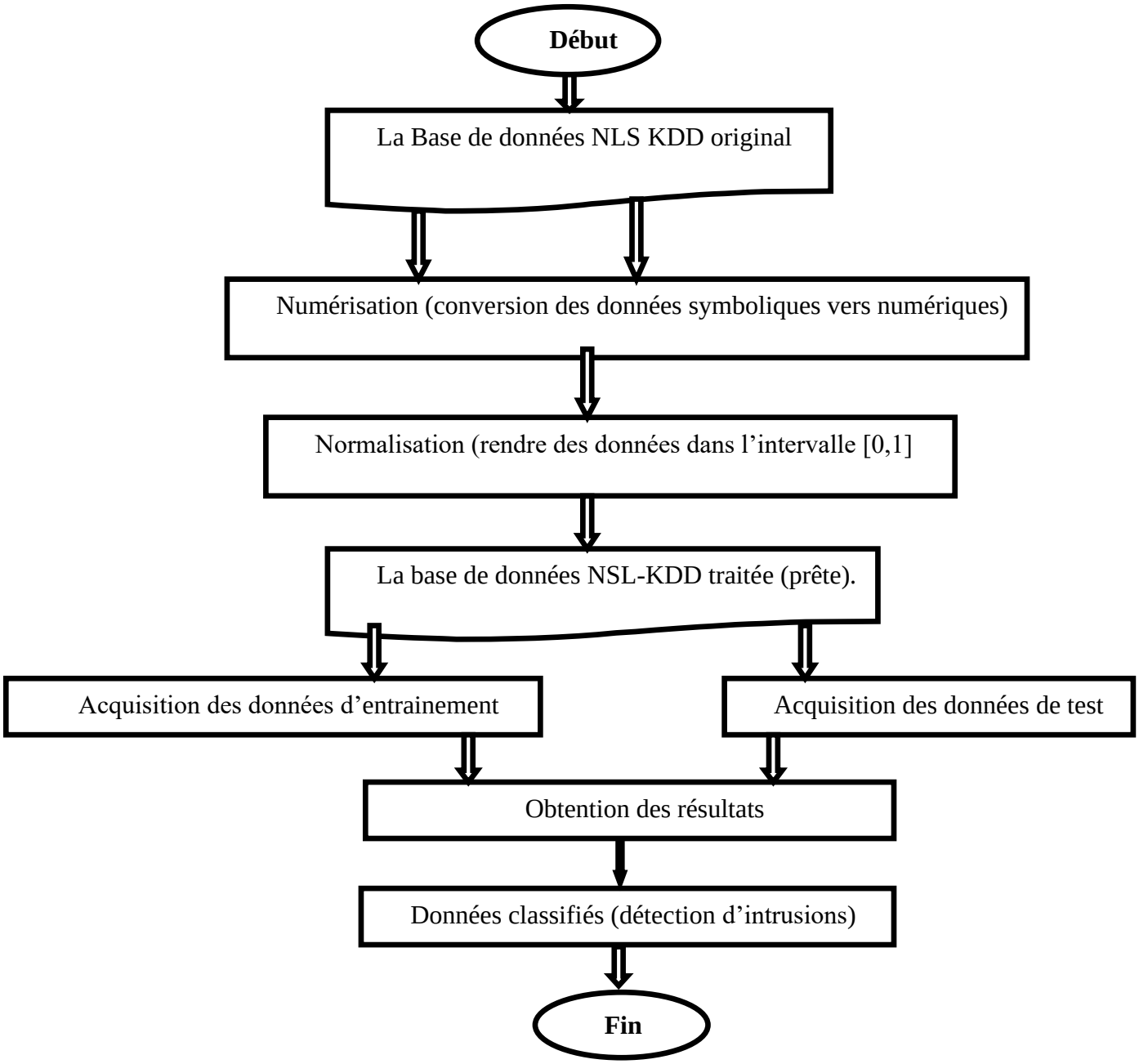

Figure3. 7: Organigramme de fonctionnement de modèle de détection d'intrusion

➢ **La Base de données NLS KDD original**

L'ensemble de données NSL-KDD présente les avantages suivants par rapport à l'ensemble de données KDD d'origine :

- ❖ Il n'inclut pas les enregistrements redondants dans l'ensemble du train, de sorte que les classificateurs ne seront pas biaisés vers des enregistrements plus fréquents.
- ❖ Il n'y a pas de doublons dans les ensembles de tests proposés ; par conséquent, les performances des apprenants ne sont pas biaisées par les méthodes qui ont de meilleurs taux de détection sur les dossiers fréquents.
- ❖ Le nombre d'enregistrements sélectionnés de chaque groupe de niveau de difficulté est inversement proportionnel au pourcentage d'enregistrements dans l'ensemble de données KDD d'origine. Par conséquent, les taux de classification des méthodes distinctes d'apprentissage automatique varient dans un plus large éventail, ce qui rend plus efficace d'avoir une évaluation.

```
[11] conn_data_train.head()
```

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...
0	0	tcp	ftp_data	SF	491	0	0	0	0	0	...
1	0	udp	other	SF	146	0	0	0	0	0	...
2	0	tcp	private	S0	0	0	0	0	0	0	...
3	0	tcp	http	SF	232	8153	0	0	0	0	...
4	0	tcp	http	SF	199	420	0	0	0	0	...

5 rows × 43 columns

Figure3. 8: La base de données NSL-KDD

Importation de données

```
[8] conn_data_train = pd.read_csv('/content/dataset_taissir/Train.txt', header = None)
    conn_data_test = pd.read_csv('/content/dataset_taissir/Test.txt', header = None)
```

```
conn_data_train.head()
```

	0	1	2	3	4	5	6	7	8	9	...	33	34	35	36	37	38	39	40	41	42
0	0	tcp	ftp_data	SF	491	0	0	0	0	0	...	0.17	0.03	0.17	0.00	0.00	0.00	0.05	0.00	normal	20
1	0	udp	other	SF	146	0	0	0	0	0	...	0.00	0.60	0.88	0.00	0.00	0.00	0.00	0.00	normal	15
2	0	tcp	private	S0	0	0	0	0	0	0	...	0.10	0.05	0.00	0.00	1.00	1.00	0.00	0.00	neptune	19
3	0	tcp	http	SF	232	8153	0	0	0	0	...	1.00	0.00	0.03	0.04	0.03	0.01	0.00	0.01	normal	21
4	0	tcp	http	SF	199	420	0	0	0	0	...	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	normal	21

5 rows × 43 columns

Figure3. 9: Numérisation (conversion des données symboliques vers numériques)

➢ **Normalisation (rendre des données dans l'intervalle [0,1]**

Pour la classification binomiale, nous convertissons la variable d'attaque en 0-1

- Normale
- Valeurs autres que normales (attaque)

```
[15] conn_data_train.attack.head()

0    0
1    0
2    1
3    0
4    0
Name: attack, dtype: int64
```

Figure3. 10 : Normalisation (rendre des données dans l'intervalle [0,1]

➢ **La base de données NSL-KDD traitée (prête).**

```
cont_var_summary = cont_var_train.apply(lambda x : continous_var_summary(x)).T.round(1)
cont_var_summary
```

attack	125973.0	0.0	5.863000e+04	0.5	0.0	0.5	2.000000e-01	0.0	0.0	0.0	0.0	0.0	0.0
count	125973.0	0.0	1.059528e+07	84.1	14.0	114.5	1.311220e+04	0.0	1.0	1.0	1.0	2.0	14.0
diff_srv_rate	125973.0	0.0	7.942900e+03	0.1	0.0	0.2	0.000000e+00	0.0	0.0	0.0	0.0	0.0	0.0
dst_bytes	125973.0	0.0	2.491634e+09	19779.1	0.0	4021269.2	1.617061e+13	0.0	0.0	0.0	0.0	0.0	0.0
dst_host_count	125973.0	0.0	2.294585e+07	182.1	255.0	99.2	9.841900e+03	0.0	1.0	3.0	11.0	82.0	255.0
dst_host_diff_srv_rate	125973.0	0.0	1.044960e+04	0.1	0.0	0.2	0.000000e+00	0.0	0.0	0.0	0.0	0.0	0.0
dst_host_rerror_rate	125973.0	0.0	1.496960e+04	0.1	0.0	0.3	1.000000e-01	0.0	0.0	0.0	0.0	0.0	0.0
dst_host_same_src_port_rate	125973.0	0.0	1.869170e+04	0.1	0.0	0.3	1.000000e-01	0.0	0.0	0.0	0.0	0.0	0.0
dst_host_same_srv_rate	125973.0	0.0	6.566240e+04	0.5	0.5	0.4	2.000000e-01	0.0	0.0	0.0	0.0	0.0	0.5
dst_host_serror_rate	125973.0	0.0	3.583330e+04	0.3	0.0	0.4	2.000000e-01	0.0	0.0	0.0	0.0	0.0	0.0
dst_host_srv_count	125973.0	0.0	1.456916e+07	115.7	63.0	110.7	1.225510e+04	0.0	1.0	1.0	2.0	10.0	63.0
dst_host_srv_diff_host_rate	125973.0	0.0	4.099500e+03	0.0	0.0	0.1	0.000000e+00	0.0	0.0	0.0	0.0	0.0	0.0

➢ **Résultat**

Figure3. 11: La base de données NSL-KDD traitée (prête).

➢ **Diviser les données en train et tester**

```
[52] train_x, test_x, train_y, test_y = train_test_split(conn_data_train_new[feature_columns],
                                                         conn_data_train_new['attack'], test_size = 0.3, random_state = 222)
     print('No. of observation train: ' , train_x.shape[0], ' | ', 'No. of observation test: ' ,test_x.shape[0])

No. of observation train:  88181  |  No. of observation test:  37792
```

Figure3. 12: Diviser les données en train et tester

Ma figure ci-dessous Vérifier le score de précision sur le train et le test :

```
#1. Vérifier le score de précision sur le train et le test

print('Accuracy Score for train dataset : ' , metrics.accuracy_score(rf_train_predict.actual, rf_train_predict.predicted))
print('Accuracy Score for validation dataset : ' , metrics.accuracy_score(rf_test_predict.actual, rf_test_predict.predicte

print('Accuracy Score for test dataset : ' , metrics.accuracy_score(conn_data_test_new.attack, test_predicted))
```

```
Accuracy Score for train dataset :  1.0
Accuracy Score for validation dataset :  0.9998147756138865
Accuracy Score for test dataset :  0.844082682753726
```

```
#2. Vérifier le score roc_auc sur le train et le test

print('ROC-AUC Score for train dataset : ' , metrics.roc_auc_score(rf_train_predict.actual, rf_train_predict.predicted))
print('ROC-AUC Score for validation dataset : ' , metrics.roc_auc_score(rf_test_predict.actual, rf_test_predict.predicted

print('ROC-AUC Score for test dataset : ' , metrics.roc_auc_score(conn_data_test_new.attack, test_predicted))
```

```
ROC-AUC Score for train dataset :  1.0
ROC-AUC Score for validation dataset :  0.9998157240689962
ROC-AUC Score for test dataset :  0.8596288058804219
```

Figure3. 13: le score de précision sur le train et le test

- **Données classifiés (détection d'intrusions)**

```
#4. Créer un rapport de classement
print(metrics.classification_report(conn_data_test_new.attack, test_predicted))
```

```
              precision    recall  f1-score   support

           0       0.74      0.97      0.84      9711
           1       0.97      0.75      0.85     12833

    accuracy                           0.84     22544
   macro avg       0.86      0.86      0.84     22544
weighted avg       0.87      0.84      0.84     22544
```

Figure3. 14: Données classifiés

Nous avons utilisé la méthode (SVM), c'est une technique réalisée par Vapnik basée sur la méthode d'apprentissage automatique et sur la théorie de l'apprentissage statistique [19], et l'un des algorithmes supervisés pour résoudre des problèmes de discrimination et de régression. Le SVM est l'un des classifieur linéaires utilisés pour séparer les données par un hyperplan.

Le modèle proposé se construire un système de détection d'intrusion réseau pour détecter les anomalies et les attaques sur le réseau la figure ci-dessous représente l'architecture du model :

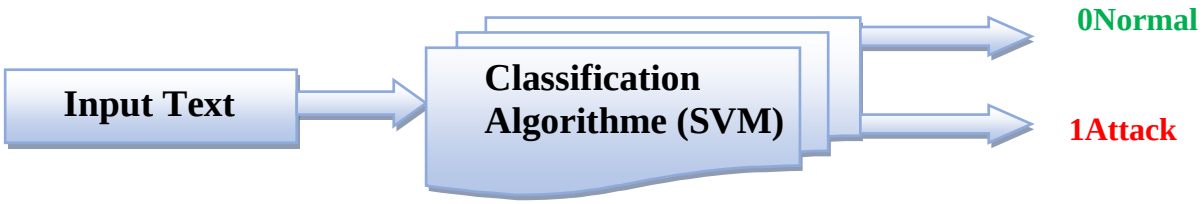

Figure3. 15: Architecture de modèle proposé

5. Base d'apprentissage NSL-KDD

La base de données NSL-KDD (Fig. 3.8) comporte 125973 trames, chaque trame est soit une attaque ou une trame normale. Cette base de données contient 22 types d'attaque, on peut répartir ces attaques en 4 catégories. [20]

Tableau 3. 2: représentation de la base de données NSL-KDD.

Catégorie	Type d'attaque	Nombre de trame	Taux de trame %
Normal	Normal	67343	53.46
Dos	neptune, smurf, teardrop, pod, back, land	45927	36.46
Prob	portsweep, ipsweep, nmap, satan	11656	9.25
R2L	guess_passwd, ftp_write, imap, phf, warezclient, multihop, warezmaster, spy	995	0.79
U2R	buffer_overflow, loadmodule, perl, rootkit	52	0.04

Classification des attaques : SVM est ensuite classée selon les données observées, qu'il s'agisse d'une activité normale ou d'une attaque. S'il s'agit d'une attaque, elle sera divisée en quatre catégories supplémentaires qui peuvent être R2l, Probe, U2R et DOS. La classification des attaques est donnée dans la figure 3.7.

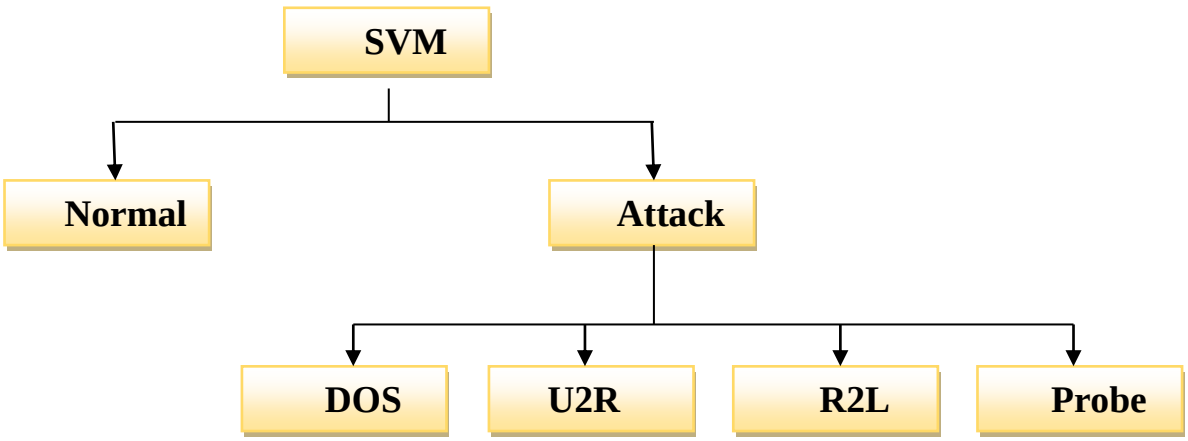

Figure3. 16: Classification de quatre types d'attaques

Les Données de NSL-KDD sont de trois types: numérique, Nominale et binaire. Les attributs 2, 3 et 4 sont nominales, 7, 12, 14, 15, 21 et 22 sont binaires, et le reste des attributs sont de type numérique comme le montre le tableau (1) de l'‟annexe, avant de passer au travail expérimental, l'ensemble de données NSL-KDD est d'abord passé par une opération de prétraitement des données et la conversion du type des attributs.

- **Dataset utilisé**

NSL-KDD est un ensemble de données suggéré pour résoudre certains des problèmes inhérents à l'ensemble de données KDD'99. Bien que cette nouvelle version de l'ensemble de données KDD souffre encore de certains des problèmes discutés par McHugh et peut ne pas être un parfait représentant des réseaux réels existants, en raison de l'absence d'ensembles de données publiques pour les IDS basés sur le réseau, nous croyons qu'il peut encore être appliqué comme un ensemble de données de référence efficace pour aider les chercheurs à comparer les différentes méthodes de détection des intrusions. De plus, le nombre de dossiers dans le train et les ensembles d'essais NSL-KDD est raisonnable. Cet avantage rend abordable d'exécuter les expériences sur l'ensemble complet sans avoir besoin de sélectionner au hasard une petite partie. Par conséquent, les résultats de l'évaluation des différents travaux de recherche seront cohérents et comparables.[20]

6. Résultat

Dans cette partie nous avons présenté les résultats de chaque partie de notre modèle proposée :

6.1. Score de précision pour l'ensemble de données

Tableau 3. 3: Taux de précision

Type d'attaque	Taux de précision
Dos	79.25%
Normal	79.55%
Prop	79.55%
R2L	79.78
U2R	79.87%

Dans notre modèle nous atteint un taux de précision qui égal à

- Accuracy score on testing data 0.9995
- Accuracy score on testing data 0.8301
- Accuracy score for train dataset 0.886

- Accuracy score for test dataset 0.738

6.2. Matrice de Confusion

Les ensembles de données étaient séparés en poids égaux afin de concaténer les fichiers, nous obtenions une partie commune qui produisait un ensemble de données, puis une influence de la matrice de confusion entre les 4 catégories d'attaque.

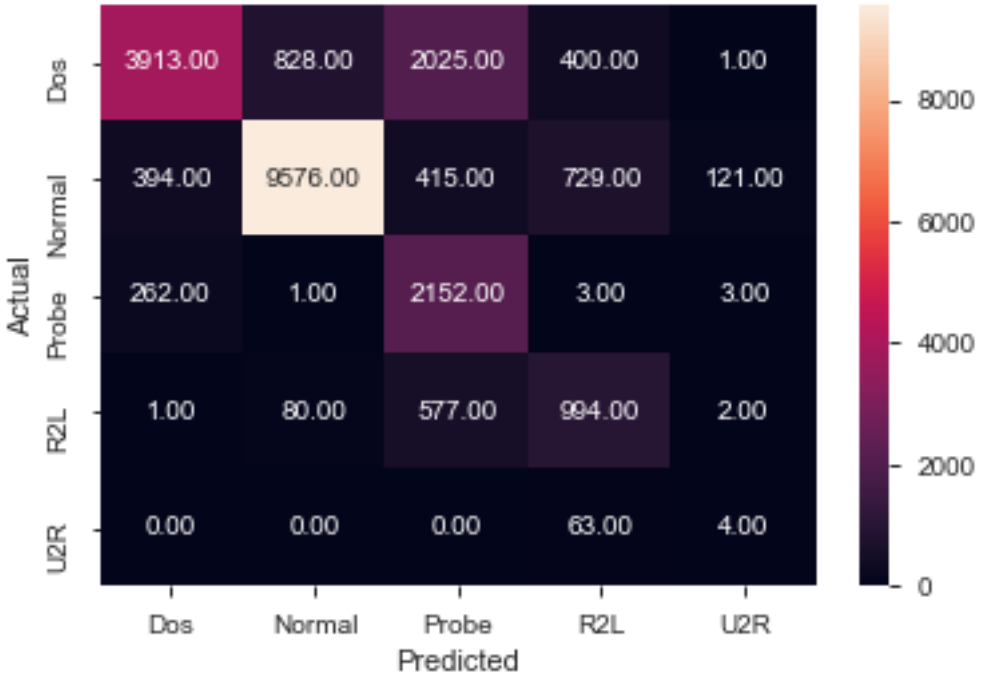

Figure3. 17:Matrice de confusion

7. Conclusion

Ce chapitre a été consacré en premier lieu à la présenté les outils utilisés pour la configuration de notre travail, nous a permis d'acquérir une bonne base de connaissance sur python. Ensuite, nous avons décrit les différentes étapes pour réaliser notre model utilisées. Enfin, nous avons établi la détection d'intrusion.

Conclusion Générale

Parmi les différents outils de sécurité informatique, on trouve le system de détection d'intrusion. Cet outil est devenu très indispensable pour tout réseau informatique, il nous permet de connaitre toutes activités anormales qui peuvent présenter un danger pour notre réseau.

Les systèmes de détection d'intrusion sont généralement classifiés en deux catégories, les systèmes de détection d'intrusion par signatures et les systèmes de détection d'intrusion par anomalies.

Les systèmes de détection d'intrusion par signatures : ce type a montré beaucoup de limites avec la rapidité et l'augmentation du trafic réseau mais il ne peut pas détecter les nouvelles formes des attaques.

Les systèmes de détection d'intrusion basée sur les anomalies ont été proposé afin de traiter les problèmes du premier type, ou les techniques d'apprentissage automatique ont été utilisés. Malgré la puissance et l'efficacité des techniques de l'apprentissage automatique, les systèmes de détection d'intrusion de ce dernier souffrent de certaines limites comme la nécessité de faire une mise à jour régulière, la nécessité de préparer les données d'entrainement, la difficulté de détecter les nouvelles formes d'attaques etc.

Dans notre étude, nous avons proposé une approche basée sur l'apprentissage automatique, et cette proposition est basée sur plusieurs niveaux de détection d'intrusion utilisant de nombreux algorithmes d'apprentissage visant à améliorer les performances et la précision des identifiants.

Le travail consiste à sélectionner où bien classer la catégorie d'attaque avant de classer son type a l'aide du Modèle de sélection, qui a été implémenté par le SVM, qui permettent de résoudre des problèmes tant de classification que de régression ou de détection d'anomalie. Concernant la deuxième étape, elle contient quatre modèles nommés Classificateur de Type, chaque catégorie (DOS, Probe, U2R et R2L) a son Classificateur de types spécifique qui est offert par sa spécialisation avec une précision élevée, ces classificateurs utilisent Random Forest.

Références/bibliographiques

[1]. 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.

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

[3]. H. Debar, M. Dacier, and A. Wespi. Towards a taxonomy of intrusion-detection systems. Computer Networks, 1999. A. Phillip, Porras and Alfonso Valdes. Live traffic analysis of tcp /ip gateways. Proc. ISOC Symposium on Network and Distributed System Security (NDSS98). (San Diego, CA, March 98), Internet Society.

[4]. J. Olivain. Plate-forme de détection d'intrusions : Analyse et corrélation temporelle d'événements en temps réel. Rapport de stage, Laboratoire Spécification et Vérification, ENS Cachan, France, novembre, décembre 2003.H. Debar, M. Dacier & A. Wespi. "A revised taxonomy for intrusion detection systems. Annales des télécommunications''. July–August 2000.

[5]. B. Morin. Corrélation d'alertes issues d'outils de détection d'intrusions avec prise en compte d'informations sur le système surveillé. Thèse de doctorat, Institut National des Sciences Appliquées de Rennes, fèvrier 2004.

[6]. https://fr.wikipedia.org/

[7]. M. Parchami, S. Bashbaghi, E. Granger, and S. Iftekar Sayed, "Using Deep Autoencoders to Learn Robust Domain-Invariant Representations for Still-to-Video Face Recognition," presented at the Advanced Video and Signal based Surveillance, Lecce, Italy, 2017.

[8]. P. Laskov & P. Düssel & C. Schäfer & K. Rieck (2005), _Learning intrusion detection: Supervised or unsupervised? _, Fraunhofer-FIRST.IDA, Berlin,Germany.

[9]. Marcus A. Maloof (2005), _Machine Learning and Data Mining for Computer Security_ , Springer London Ltd, ISBN-10 184628029X ; ISBN-13 978-1846280290.

[10]. Sailesh Kumar, « Survey of Current Network Intrusion Detection Techniques », CSE571S: Network Security, 2007, p. 102-107

[11]. Abdelli belkacem (Année universitaire 2020-2021)Classification Automatique des documents textuels

[12]. voisins par Faïcel Chamroukhi, Classification supervisée : Les K-plus proches

[13]. https://projeduc.github.io/: Introduction à l'apprentissage automatique

[14]. Mohamed Yessin AMMAR, « Mise en couvre de réseau de neurones pour la modélisation de cinétique réactionnelles en vue de la transposition batch/continu » , Thèse de doctorat , École Nationale d'Ingénieurs de Sfax , Tunisie , 17 juillet 2007.

[15]. CLASSIFICATION DE LA SÉVÉRITÉ DES BOGUES PAR L'UTILISATION DE MÉTRIQUES TIRÉES DE L'HISTORIQUE GIT

[16]. Kim, Yoon . 2014. "Convolutional Neural Networks for Sentence Classification./I ln Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (Emnlp), 1746-51. Doha, Qatar: Association for Computational Linguistics.

[17]. https://www.codingame.com/playgrounds/56931/les-bases-de-python-pour-le-lycee/3--manipulations-dimages-suite

[18]. Réseaux à convolution avancés et architectures convolutives modernes Par Pierre-Marc Jodoin

[19]. Mohamadally Hasan Fomani Boris BD Web, ISTY3 Versailles St Quentin, France.

[20]. https://www.unb.ca/cic/datasets/nsl.htm

Annexe

Annexe 1 : Les attributs de NSL KDD

Les détails des attributs sont répertoriés dans les tableaux suivants

N°	Nom de l'attribut	Type	Description
1	duration	Numérique	La durée de connexion
2	protocol_type	Nominal	Protocole utilisé dans la connexion (tcp, udp, icmp)
3	service	Nominal	Service réseau de destination,(http,telnet, ftp_data, etc.)
4	flag	Nominal	Statut de la connexion –Normal ou Erreur (SF, REJ, S0, S1, etc.)
5	src_bytes	Numérique	Nombre d'octets de donnée transférés de la source à la destination (491, etc.)
6	dst_bytes	Numérique	Nombre d'octets de données transférés de destination à la source (0, etc.)
7	land	Binaire	**Si** l'adresse IP de source et destination et le nombre de port sont les mêmes **alors**, *land*=1 **sinon** *land*=0
8	wrong_fragment	Numérique	Nombre total de fragments erronésdans cette connexion
9	urgent	Numérique	Nombre de paquets urgents
10	Hot	Numérique	Nombre d'indicateurs « Hot »
11	num_failed_logins	Numérique	Nombre de tentatives de connexion échouées
12	logged_in	Binaire	**Si** connecté avec succès **alors** *logged_in*=1 **sinon** *logged_in*=0
13	num_compromised	Numérique	Nombre de conditions compromises
14	root_shell	Binaire	1 si le root shell est obtenu, 0autrement
15	su_attempted	Binaire	1 **si** la commande "su root" a été tentée ou utilisée, **sinon** 0
16	num_root	Numérique	Nombre d'accès " root " ou nombre d'opérations effectuées comme racine dans la connexion
17	num_file_creations	Numérique	Nombre d'opérations de création de fichiers
18	num_shells	Numérique	Nombre d'invites du shell

19	num_access_files	Numérique	Nombre d'opérations sur les fichiers de contrôle d'accès
20	num_outbound_cmds	Numérique	Nombre de commandes sortantes dans une session FTP
21	is_host_login	Binaire	1 **si** la connexion appartient à la liste du « hot » (root ou admin);**sinon** 0
22	is_guest_login	Binaire	1 **si** le login est un login «guest »;**sinon** 0
23	count	Numérique	Nombre de connexions vers le même hôte de destination que la connexion en cours dans les deux dernières secondes
24	srv_count	Numérique	Nombre de connexions vers le même service (N° Port) que la connexion en cours dans les deux dernières secondes
25	serror_rate	Numérique	Le pourcentage de connexions qui ont activé le *flag* s0, s1, s2 ou s3,parmi les connexions agrégées dans *count*
26	srv_serror_rate	Numérique	Le pourcentage de connexions qui ont activé le *flag* s0, s1, s2 ou s3,parmi les connexions agrégées dans *srv_count*
27	rerror_rate	Numérique	Le pourcentage de connexions qui ont activé le *flag* REJ, parmi les connexions agrégées dans *count*
28	srv_rerror_rate	Numérique	Le pourcentage de connexions qui ont activé le *flag* REJ, parmi les connexions agrégées dans srv_*count*
29	same_srv_rate	Numérique	Le pourcentage de connexions qui sont au même service, parmi les connexions agrégées dans *count*
30	diff_srv_rate	Numérique	Le pourcentage de connexions qui sont aux différents services, parmi les connexions agrégées dans *count*
31	srv_diff_host_rate	Numérique	Le pourcentage de connexions qui sont à différentes machines de destination, parmi les connexions agrégées dans srv_*count*
32	dst_host_count	Numérique	Nombre de connexions ayant la même adresse IP de l'hôte de destination
33	dst_host_srv_count	Numérique	Nombre de connexions ayant le même numéro de port
34	dst_host_same_srv_rate	Numérique	Le pourcentage de connexions qui sont au même service, parmi les connexions agrégées dans *dst_host_count*
35	dst_host_diff_srv_rate	Numérique	Le pourcentage de connexions qui sont aux différents services, parmi les connexions agrégées dans *dst_host_count*
36	dst_host_same_src_port_rate	Numérique	Le pourcentage de connexions qui sont au même port de source, parmi les connexions agrégées dans*dst_host_srv_count*

Résumé

Ce mémoire intitule "création d'un modèle de détection d'intrusion base sur l'apprentissage automatique" entre dans la phase de réalisation de mémoire de fin d'étude de ISSAT Gafsa.

Son but principal et de créer un modèle intelligent base sur l'apprentissage profond « deeplearning »

Ce modèle doit détecter les intrusions pour réaliser le modèle que nous avons utilisé avant le réseau de neurone convolutif « CNN » utilisé sont KDD99.

Mots clés : détection, apprentissage automatique, **Intelligence artificielle**, **KDD**

Abstract

This thesis entitled "creation of an intrusion detection model based on machine learning" enters the phase of realization of ISSAT Gafsa end-of-study dissertation.

Its main goal is to create an intelligent model based on deep learning.

This model must detect intrusions to realize the model we used before the convolutional neural network "CNN" used are KDD99.

Keywords: detection, machine learning, artificial intelligence, KDD

الخلاصة

هذه الرسالة العلمية بعنوان "إنشاء نموذج للكشف عن التسلل بناءً على تعلم الآلة" تدخل مرحلة إنجاز مذكرة نهاية الدراسة في معهد العلي للعلوم التطبيقية وتكنولوجيا بقفصة.

هدفها الرئيسي هو إنشاء نموذج ذكي يستند إلى التعلم العميق.

يجب أن يكون هذا النموذج قادرًا على اكتشاف المتسللات لتنفيذ النموذج الذي استخدمناه سابقًا الشبكة وهي العصبية التابعة للتصفية المرئية والتي استخدمت مجموعة البيانات.

الكلمة المفاتيح :الكشف، التعلم الآلي، الذكاء الاصطناعي