République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Aïn-Témouchent Belhadj BOUCHAIB

Faculté des sciences et de la technologie
Département des Mathématiques et de l'Informatique

Mémoire

En vue de l'obtention du Diplôme de Master en Informatique

Option :

Réseaux et Ingénierie des Données (RID)

Présenté par :

Mr. DEYOKO Mamadou Kariba

SYSTÈME DE DÉTECTION D'INTRUSIONS INFORMATIQUES PAR RÉSEAUX DE NEURONES.

Encadrant :

Dr. BELGRANA Fatima Zohra
Maître de Conférences "A" à l'U.B.B.A.T.

Devant le jury composé de :

Président :	*Mlle. Bouhalouan Djamila (M.A.A)*	U.B.B.A.T.
Examinateur :	*Mr. Messaoudi Mohamed Amin (M.A.A)*	U.B.B.A.T.

Année universitaire 2020-2021

Remerciements

Je remercie avant toute chose le tout miséricordieux de m'avoir donné la vie, la santé mentale et physique pour mener à terme ce projet, à mes parents pour leur soutien et leurs sacrifices ainsi mes frères et sœurs, à mes tontons et tantes, à mes cousins et amis.

Je remercie aussi le Dr BELGRANA Fatima Zohra, Maitre de Conférences "A" au C.U.B.B.A.T, pour sa collaboration, son aide et ses conseils tout au long de ce projet.
Je remercie Mlle Bouhalouan Djamila d'avoir accepté d'examiner ce travail. Je remercie Mr Messaoudi Mohamed Amin d'avoir accepté d'examiner ce travail.
Enfin je remercie toute personne ayant contribué de près ou loin à la réalisation de ce travail.

Table des Matières

Table des Figures

Liste des tableaux

Introduction générale

Les ordinateurs sont considérés comme l'une des avancées technologiques les plus importantes du XXe siècle. Au cours des dernières années, l'utilisation des ordinateurs à la maison et dans les entreprises a considérablement augmentée. Cependant, les problèmes de sécurité et de confidentialité existent depuis longtemps, avant même que l'ordinateur ne devienne un élément vital des opérations des organisations. Aujourd'hui, la sécurité est un problème majeur et de plus en plus important pour tous les réseaux et ordinateurs dans l'environnement d'entreprise d'aujourd'hui. Internet (comme beaucoup d'autres choses) est à double tranchant. C'est l'entrée de beaucoup de choses bénéfiques. Malheureusement, cela ouvre également la voie à de nombreuses choses nuisibles pour se connecter à votre appareil. Les pirates et les intrus ont fait de nombreuses tentatives réussies pour faire tomber les réseaux et les systèmes d'entreprises de premier plan. De nombreuses méthodes ont été développées pour sécuriser l'infrastructure du système et la communication sur Internet, telles que l'utilisation de pare-feu, la détection d'intrusion et le système de cryptage[Aboud, 2009].

La détection d'intrusion est le processus de surveillance des événements se produisant dans un système informatique ou un réseau et de leur analyse pour détecter des signes d'intrusion. Elle vise à protéger la confidentialité, l'intégrité et la disponibilité des systèmes d'information critiques en réseau[Seleznyov, 2002]. Le Système de Détection d'Intrusion (SDI) est un système qui rassemble et analyse des informations provenant de diverses zones d'un ordinateur ou d'un réseau pour identifier les attaques menées contre ces composants. Le SDI utilise un certain nombre de méthodes génériques pour surveiller les exploitations de vulnérabilités.

Notre objectif alors est de réaliser un SDI tout en améliorant la qualité de détection, nous nous sommes situés dans approche basée signature en utilisant la base de données NSL-KDD[Unb 2019]. Nous avons opté pour un Réseau de Neurones Artificiel (RNA) et plus particulièrement la carte de Kohonen qui représente un véritable outil de classification automatique.

Notre mémoire est organisé en 3 chapitres, le premier chapitre porte sur la Sécurité Réseau (SR) et les systèmes de détection d’intrusions, où nous introduisons dans un premier temps quelques notions de base de la SR telles que la politique de sécurité informatique, les types de menaces et techniques d’attaque, ainsi que les dispositifs de sécurité. Dans un second temps,

nous présentons une classification des SDIs en illustrant les avantages et les inconvénients de chaque catégorie.

Le second chapitre est dédié à un état de l'art sur les SDIs en présentant d'abord les différentes méthodes issues de l'apprentissage automatique et plus particulièrement les RNs puis nous verrons quelques travaux des SDIs via cette approche.

Le troisième chapitre quant à lui, est dédié à la conception de notre SDI, nous décrivons d'abord la base de données utilisée, puis notre approche proposée en illustrant l'algorithme de la carte de Kohonen adopté. Nous clôturons le chapitre avec la présentation des résultats que nous avons obtenus ainsi que notre application.

Chapitre 1 : Sécurité informatique et système de détection d'intrusion

1.1 Introduction

L'internet de nos jours, face à l'abondance des gens connectés et du nombre de liens existants est tant confronté à un problème de gestion qu'un problème de sécurité. En effet, toute sorte de personne utilise internet, aussi des bienveillants que des malveillants. Pour assurer la bonne cohésion de ce trafic influent, il nous est essentiel de comprendre certains aspects surtout du point de vue des malveillants, un coté souvent négligé par le tier de la population.

Avec le développement d'Internet, chacun a accès au réseau où de plus en plus d'informations circulent. De plus en plus, les entreprises communiquent et diffusent via ce media, que ce soit dans leurs liens avec leurs fournisseurs ou leurs partenaires ou en interne, dans les relations entre les employés eux-mêmes.

Nous sommes face non seulement à une augmentation de la quantité, mais aussi et surtout de l'importance des données.

Cet environnement est sujets à des attaques à cause des failles laisse par le système qui est tant renforce par des outils telles que : les mécanismes de chiffrement, les pares-feux, les anti-virus et ce pour la détection d'intrusion.

Ce chapitre sera dédié pour discuter les aspects de la sécurité informatique et des systèmes de détection d'intrusion.

1.2 Sécurité informatique

1.2.1 Définition

La sécurité informatique est l'ensemble des moyens mis en œuvre pour réduire la vulnérabilité d'un système contre les menaces accidentelles ou intentionnelles. L'objectif de la sécurité informatique est d'assurer que les ressources matérielles et/ou logicielles d'un parc informatique sont uniquement utilisées dans le cadre prévu et par des personnes autorisées.[Yende 2018]

1.2.2 Politique de sécurité informatique

Une politique de sécurité guide un système informatique selon des buts et des objectifs en fournissant un cadre pour la sélection et la mise en œuvre des contre-mesures contre les menaces, sans elle ce dernier est susceptible d'avoir un désordre de contre-mesures.[Ahmed Chaouki LOKBANI 2017]

Une bonne politique est toujours adaptée aux menaces, car elle devrait préciser qui est responsable de quoi (mise en œuvre, exécution, vérification, examen), la nature de cette politique de sécurité du réseau et pourquoi elle est de cette nature. La réponse à ces questions est très importante parce qu'une politique claire, concise, cohérente et constante est plus susceptible d'être suivie.

La politique de sécurité est de savoir comment vous déterminez quelles contre-mesures il faut utiliser. Par exemple : Avez-vous besoin d'un pare-feu? Comment devez-vous configurer votre pare-feu ? Avez-vous besoin d'un jeton d'accès, ou un mot de passe est suffisant ? Les utilisateurs sont autorisés à accéder à la vidéo en streaming à partir de leurs navigateurs Web ? S'il n'y a pas de politique, il n'y aura pas de base pour répondre systématiquement à ces questions. Malheureusement, la plupart des organisations n'ont pas une politique de sécurité réseau. Ou bien ils le font, mais personne ne la suit.

1.2.3 Exigence et objectifs

L'objectif de la sécurité est d'assurer les cinq principes clés suivants:[Bloch et al. 2013]

- **La confidentialité :** La confidentialité est définie par l'organisation internationale de normalisation (ISO) comme « le fait de s'assurer que l'information n'est seulement accessible qu'à ceux dont l'accès est autorisé », elle consiste à préserver la révélation non autorisée d'information sensible. La révélation pourrait être intentionnelle comme les attaques qui visent de casser le chiffrement des données et lire les informations, ou involontaire dû au manque de vigilance ou de l'incompétence des individus qui manient les informations.
- **La disponibilité :** La disponibilité assure la pérennité du service opportun aux utilisateurs autorisés qui ont un accès non interrompu aux informations dans le système et le réseau.
- **L'intégrité :** L'intégrité est la propriété d'une information de ne pas être altérée. Donc le système informatique doit :
  - Empêcher une modification par une personne non autorisée ou une modification incorrecte par une personne autorisée.

- Faire en sorte qu'aucun utilisateur ne puisse empêcher une modification légitime de l'information.

En plus, il faut se prémunir contre les fautes affectant l'intégrité des données, en intégrants dans le système des mécanismes permettant de détecter les modifications des informations d'une part et de contrôler l'accès à ces dernières d'autre part (en gérant les droits d'accès des programmes et utilisateurs). De plus, une validation en amont peut également être réalisée pour prévenir les fautes accidentelles.

- **L'authentification :** L'authentification est tout simplement le contrôle d'accès, ce service signifie que celle-ci les personnes autorisées peuvent accéder aux informations, des simples moyens comme la gestion des mots de passe permet de garantir les services d'authentification.
- **La non répudiation :** Cette propriété garantie qu'un sujet ayant réalisé une action dans le système ne puisse nier l'avoir réalisée. Assurer la non répudiation d'une transmission signifie assurer que les extrémités d'une transmission (émetteur et récepteur) sont bien les seules personnes autorisées à envoyer ou réceptionner les informations sans aucune remise en cause, qui est d'ailleurs généralement garanti par le moyen d'un fichier électronique appelé certificat numérique qui assure l'identité de l'émetteur et le récepteur. Les certificats eux-mêmes sont protégés par le moyen des signatures des utilisateurs, dans certains cas la signature d'un utilisateur est son identité.

1.2.4 Menace

Une menace informatique représente le type d'actions susceptibles de nuire dans l'absolu à un système informatique. En termes de sécurité informatique les menaces peuvent être le résultat de diverses actions. Ce sont des adversaires déterminés capables de monter une attaque exploitant une vulnérabilité. [Ben Brahim and Amiche 2017]

a. Menace accidentelle

Ils sont liés à des erreurs involontaires des utilisateurs du poste, perte accidentelle de données, dégradation ou destruction involontaire de matériel, copies illicites de logiciels….

b. Menace intentionnelle

Une menace intentionnelle est une action exécutée par une entité pour violer la sécurité de l'information et l'utilisation non autorisée des ressources. Les menaces intentionnelles peuvent être passives ou active.

Une menace passive constitue à écouter le trafic du réseau (ou de la machine) cible, donc l'interface de la machine attaquante (du pirate) est en mode écoute. L'objectif est de découvrir et capturer des trames du réseau cible pour y rechercher des informations particulières : clés de cryptages, mots de passe ou données. Elle se réalise grâce à des outils tels que : les sniffer, les scanners, etc.

Par contre dans la menace passive, ici l'attaquant n'est plus en mode écoute. Elles consistent à modifier des données ou des messages, à s'introduire dans des équipements réseau ou à perturber le bon fonctionnement de ce réseau, et aussi à interroger le réseau (ou la machine) cible, contourner le dispositif de sécurité existant par diverses méthodes tels que: Déni de Service et Virus.

1.2.5 Attaque

Une attaque informatique est une action volontaire et malveillante menée au moyen d'un réseau informatique visant à causer un dommage aux informations et aux personnes qui les traitent (particuliers, entreprises, hôpitaux , institutions...)[Deluzarche 2019]

Les hackers utilisent plusieurs types d'attaques qui peuvent regroupées en 2 familles :

1.2.5.1 Attaque directe

C'est la plus simple des attaques. L'hacker attaque directement sa victime à partir de son ordinateur. La plupart des "script kiddies" utilisent cette technique. En effet, les programmes de hack qu'ils utilisent ne sont que faiblement paramétrables, et un grand nombre de ces logiciels envoient directement les paquets à la victime.

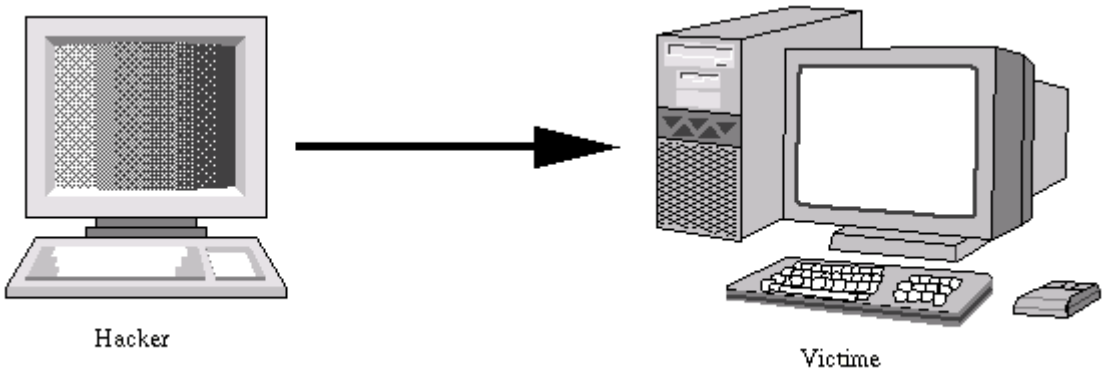

Figure 1-1 Attaque Directe[Bourouh and Kanoun 2017]

1.2.5.2 Attaque indirecte

a. Par rebond

Cette attaque est très prisée des hackers. En effet, le rebond a deux avantages :

- Masquer l'identité (l'adresse IP) de l'hacker.
- Éventuellement, utiliser les ressources de l'ordinateur intermédiaire car il est plus puissant (CPU, bande passante...) pour attaquer.

Le principe en lui-même, est simple : Les paquets d'attaque sont envoyés à l'ordinateur intermédiaire, qui répercute l'attaque vers la victime. D'où le terme de rebond.

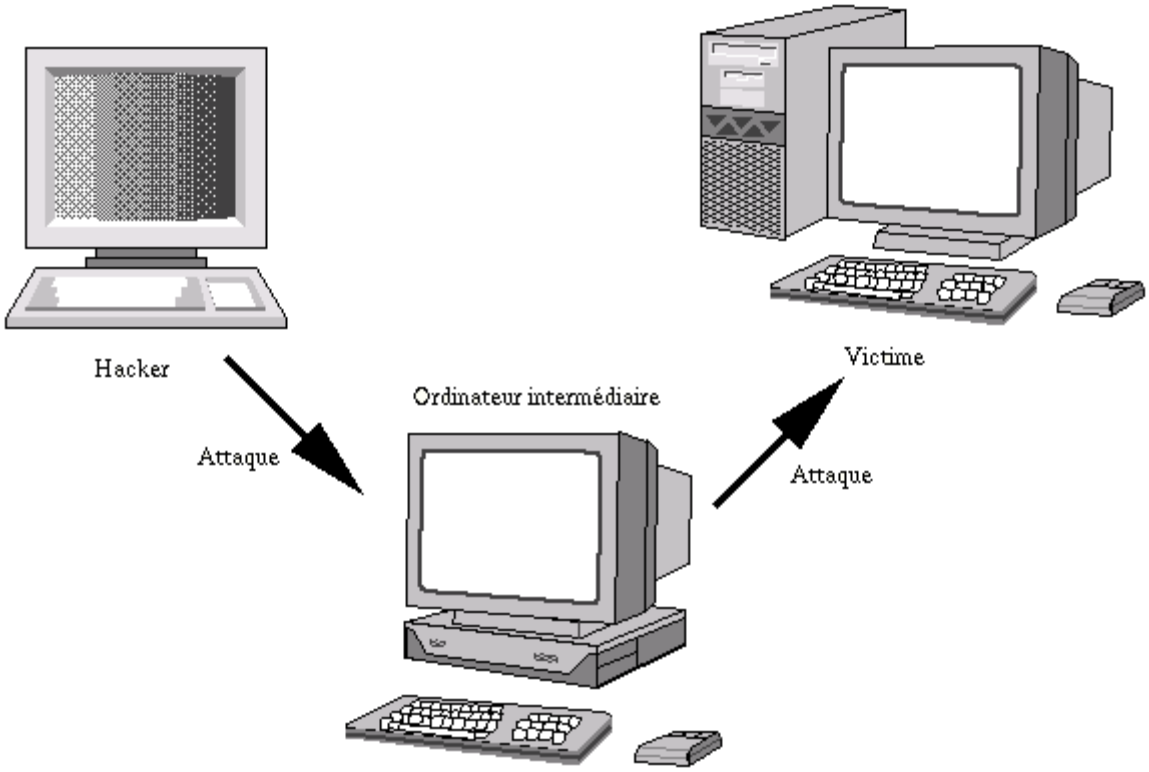

Figure 1-2 Attaque Indirecte par rebond[Bourouh and Kanoun 2017]

b. Par réponse

Cette attaque est un dérivé de l'attaque par rebond. Elle offre les mêmes avantages, du point de vue de l'hacker. Mais au lieu d'envoyer une attaque à l'ordinateur intermédiaire pour qu'il la répercute, l'attaquant va lui envoyer une requête. Et c'est cette réponse à la requête qui va être envoyée à l'ordinateur victime.

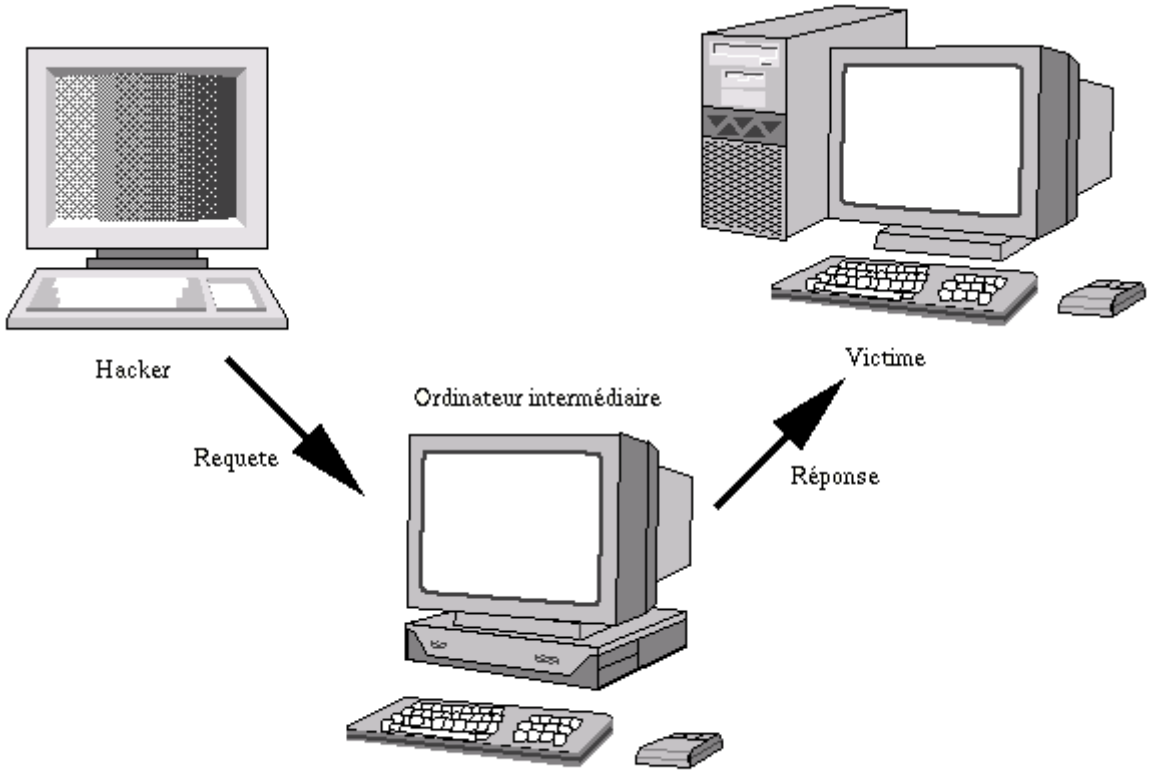

Figure 1-3 Attaque Indirecte par réponse[Bourouh and Kanoun 2017]

1.2.6 Quelques attaques informatiques

Parmi les techniques d'attaques on en distingue plusieurs

- **Virus** : un virus informatique est un logiciel ou une partie de code malveillant qui se réplique automatiquement et s'infiltre subrepticement dans votre appareil sans votre autorisation.[Avast 2019]
- **Vers** : Un ver informatique est un type de logiciel malveillant qui circule dans le monde entier par l'intermédiaire de connexions réseau à la recherche de ses cibles. Les vers sont dangereux car ils exploitent les vulnérabilités informatiques connues (par exemple, un problème du système de sécurité d'un ordinateur) pour s'insérer dans une machine. Une fois qu'ils y sont parvenus, il est extrêmement difficile de les arrêter car ils se déplacent partout pour rechercher leur cible.
- **Cheval de Troie :** Un cheval de Troie sur un ordinateur est un programme exécutable qui est présenté comme ayant une action précise, généralement bénéfique pour l'ordinateur (comme un logiciel gratuit ou sous licence), mais lorsque ce programme

est lancé, il va causer des actions plus ou moins graves sur votre ordinateur, comme supprimer des mots de passe, voler des mots de passe, envoyer des informations confidentielles au créateur du programme, formater votre disque dur...

- **Bombe Logique :** Sont appelés bombes logiques les dispositifs programmés dont le déclenchement s'effectue à un moment déterminé en exploitant la date du système, le lancement d'une commande, ou n'importe quel appel au système.
- **Logiciel espion (Spyware) :** Le logiciel espion ou spyware en anglais, est un logiciel malveillant qui s'installe de manière insidieuse dans votre ordinateur.
- **Déni de Service (DoS) :** Le déni de service (DoS, en anglais Denial of Service) apparaît lorsqu'un pirate désactive ou altère un réseau, des systèmes ou des services dans le but de refuser le service prévu aux utilisateurs normaux. Les attaques par déni de service mettent le système en panne ou le ralentissent au point de le rendre inutilisable. Le déni de service peut consister simplement à supprimer ou altérer des informations. Dans la plupart des cas, l'attaque se résume à exécuter un programme pirate ou un script. C'est pour cette raison que les attaques par déni de service sont les plus redoutées.[Ben Brahim and Amiche 2017]
- **Man in the Middle :** Le Man in the Middle (en français L'attaque de l'homme du milieu) est une attaque qui a pour but d'intercepter les communications entre deux parties, sans que ni l'une ni l'autre ne puisse se douter que le canal de communication entre elles a été compromis. Le canal le plus courant est une connexion à Internet de l'internaute lambda. L'attaquant doit d'abord être capable d'observer et d'intercepter les messages d'une victime à l'autre.
- **Sniffing :** C'est une technique qui consiste à analyser le trafic réseau. Lorsque deux ordinateurs communiquent entre eux, il y a un échange d'informations (trafic). Mais, il est toujours possible qu'une personne malveillante récupère ce trafic. Elle peut alors l'analyser et y trouver des informations sensibles.
- **Probing :** C'est une attaque qui tente d'obtenir des informations d'un réseau. Le but ici est d'agir comme un voleur et de voler des informations importantes, qu'il s'agisse d'informations personnelles sur des clients ou d'informations bancaires.
- **Usurpation d'adresse IP (spoofing IP) :**(en anglais : *IP spoofing* ou *IP address spoofing*) est une technique de piratage informatique utilisée en informatique qui consiste à envoyer des paquets IP en utilisant une adresse IP source qui n'a pas été attribuée à l'ordinateur qui les émet. Le but peut être de masquer sa propre identité lors

d'une attaque d'un serveur, ou d'usurper en quelque sorte l'identité d'un autre équipement du réseau pour bénéficier des services auxquels il a accès.

- **ARP Poisoning :**(« empoisonnement ») est une technique utilisée en informatique pour attaquer tout réseau local utilisant le protocole de résolution d'adresse ARP, les cas les plus répandus étant les réseaux Ethernet et Wi-Fi. Cette technique permet à l'attaquant de détourner des flux de communications transitant entre une machine cible et une passerelle : routeur, box, etc. L'attaquant peut ensuite écouter, modifier ou encore bloquer les paquets réseaux.[Ben Brahim and Amiche 2017]
- **User to Root :** L'objectif de cette classe d'attaques est d'obtenir la main de l'administrateur système (Root) à partir d'un simple compte utilisateur par l'exploitation des vulnérabilités, Les exploits les plus connus sont les débordements réguliers des Buffers (buffer overflows) dus aux erreurs de programmation, Les principales attaques de ce type sont : Eject, FF config, Format, Load module, Perl, Ps, Xterm.
- **Remote to User :** Dans cette classe d'attaque, l'attaquant essaye d'exploiter les vulnérabilités d'une machine distante afin d'avoir un accès illégal à cette dernière, Pour réussir cette attaque, l'attaquant exploite les bugs des applications installées dans la machine cible, les mauvaises configurations de celles-ci et du système qui les héberge, etc.

1.2.7 Dispositifs de sécurité

Il existe plusieurs dispositifs de sécurité nous citons

a. **Les anti-virus :** Un logiciel antivirus aide à protéger votre ordinateur contre les malwares et les attaques de cybercriminels

Un logiciel antivirus offre une protection contre ces types de cybermenaces en effectuant des tâches essentielles :

- Identification de fichiers spécifiques pour la détection de logiciels malveillants ;
- Planification d'analyses automatiques ;
- Analyse d'un seul fichier ou de l'ensemble de votre ordinateur au choix ;
- Suppression de codes et logiciels malveillants ;
- Vérification de la sécurité de votre ordinateur et autres appareils ;

On préconise souvent d'utiliser des antivirus avec licence et les mettre à jour, car celle-ci permet de corriger les failles détectées soit par les utilisateurs ou les concepteurs de ces systèmes informatiques. [Norton 2021]

b. Les mécanismes de chiffrement : Algorithme généralement basé sur des clefs et transformant les données. Sa sécurité est dépendante du niveau de sécurité des clefs.

c. Les pares-feux (Firewalls) : Un pare-feu est un appareil de protection du réseau qui surveille le trafic entrant et sortant et décide d'autoriser ou de bloquer une partie de ce trafic en fonction d'un ensemble de règles de sécurité prédéfinies. Un pare-feu peut être un équipement physique, un logiciel ou une combinaison des deux. (Cisco, 2021)

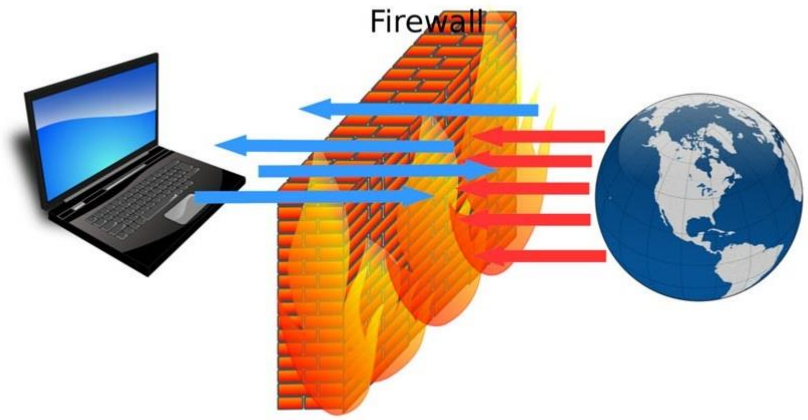

Figure 1-4Pare-feu

d. Les VPN : Un VPN (*Virtual Private Network*, en français : réseau privé virtuel) est vu comme une extension des réseaux locaux et préserve la sécurité logique que l'on peut avoir à l'intérieur d'un réseau local. Il correspond en fait à une interconnexion de réseaux locaux via une technique de « tunnel ».

La technique consiste à utiliser Internet comme support de transmission en utilisant un protocole de « tunnellisation » (en anglais *tunneling*), c'est-à-dire encapsulant les données à transmettre de façon chiffrée. On parle alors de VPN pour désigner le réseau

ainsi artificiellement créé. Ce réseau est dit virtuel car il relie deux réseaux « physiques » (réseaux locaux) par une liaison non fiable (Internet), et privé car seuls les ordinateurs des réseaux locaux de part et d'autre du VPN peuvent accéder aux données en clair.

e. **Les DMZ :** Une DMZ ou zone démilitarisée est une partie du réseau local dont l'objectif est d'être accessible depuis l'extérieur du réseau local, avec ou sans authentification préalable. En effet, pour des raisons à la fois techniques et stratégiques.

   Une zone démilitarisée est définie aussi comme un sous-réseau séparé du réseau local et isolé de celui-ci et d'Internet (ou d'un autre réseau) par un pare-feu. Ce sous-réseau contient les machines étant susceptibles d'être accédées depuis Internet. Le pare-feu contrôle toutes les communications et les autorisera à passer ou bloquera donc les accès au réseau local pour garantir sa sécurité. Et les services susceptibles d'être accédés depuis Internet seront situés en DMZ.[Ben Brahim and Amiche 2017]

f. **Les SDIs :** un autre mécanisme de protection, qui représente notre domaine de travail et qui va être détaillé dans la section suivante.

1.3 Système de détection d'intrusion

1.3.1 Définition

Le SDI est un composant important pour la sécurité du système. Il complète la sécurité d'autres technologies en fournissant des informations pour la gestion. Il ne détecte pas seulement les attaques découvertes par d'autres éléments de sécurité, mais tente également de fournir une notification de nouvelles attaques auxquelles les autres ingrédients ne peuvent pas s'attendre. Cela se fait en surveillant et en analysant en permanence les événements qui se produisent dans le système informatique ou le réseau de l'intérieur ou de l'extérieur.

Il y a trois étapes principales dans le processus de détection d'intrusion, qui sont : la surveillance et l'analyse du trafic et du système, identifier les activités anormales, et évaluer la gravité et déclencher des alarmes.

Ainsi, le SDI peut être considéré comme un système de défense qui surveille le trafic sur le réseau ou analyse l'activité du système. De cette façon, il peut détecter diverses activités qui

pourraient mettre en danger la sécurité du système. En outre, comme le SDI stocke généralement les cas de base des données dans des dossiers, cela fournit des informations précieuses et pourrait être utilisé comme preuve dans une action en justice contre l'agresseur. Par conséquent, les SDIs sont des éléments importants pour compléter l'infrastructure de l'information sécurité et conduisent le complément logique au pare-feu [Kazienko and Dorosz 2003].

1.3.2 Classification des SDIs

Il existe plusieurs types de SDI disponibles aujourd'hui, caractérisés par différentes approches de surveillance et d'analyse. Chaque approche sera expliquée et chacune d'elle présente des avantages et des inconvénients distincts. De plus, toutes les approches peuvent être décrites en termes de modèle pour les SDIs. Parmi les critères de classification les plus courants on a :

- Les sources d'information
- Le type d'analyse
- Le type de réponse
- Le temps de détection

1.3.2.1 Les Sources d'informations

Les sources d'information sont l'une des premières questions sur lesquelles se concentrer lors de la conception d'un système de détection d'intrusion. Ces sources peuvent être classées de plusieurs manières. En ce qui concerne la détection des intrusions, elles sont classées en fonction de l'emplacement en raison de certains SDIs analysant les packages du réseau, capturés à partir de la dorsale du réseau ou des segments LAN, tandis que d'autres SDIs analysent les événements générés par les systèmes d'exploitation ou les logiciels d'application à la recherche de signes d'intrusion.

a. Les SDIs bases sur le réseau (SDIR)

La majorité des systèmes de détection d'intrusion commerciaux sont basés sur le réseau. Ces IDS détectent les attaques en capturant et en analysant les paquets réseau. En écoutant un segment de réseau ou un commutateur, un IDS basé sur le réseau peut surveiller le trafic réseau affectant plusieurs hôtes connectés au segment de réseau, protégeant ainsi ces hôtes.[Bace and Mell 2001]

- **Avantages**
  - Un IDS bien situé peut surveiller un grand réseau tant qu'il a une capacité suffisante pour analyser le trafic dans sa totalité.
  - Les NIDS ont un faible impact sur le réseau, restant généralement passifs et n'interférant pas avec les opérations normales du dernier.
  - Les NIDS peuvent être configurés pour être invisibles sur le réseau afin pour augmenter la sécurité contre les attaques.
  - NIDS peut être très sûr contre l'attaque et peut être invisible pour beaucoup d'attaquants.
- **Inconvénients**
  - Il est difficile à traiter tous les paquets circulant sur un grand réseau. De plus il ne peut pas reconnaître des attaques pendant le temps de haut trafic.
  - Les IDS basés sur le réseau ne peuvent pas analyser les informations cryptées. Ce problème augmente à mesure que de plus en plus d'organisations (et d'attaquants) utilisent des réseaux privés virtuels (VPN).
  - Quelques NIDS provoquent des paquets en fragments. Ces paquets mal formés font devenir l'IDS instable.
  - La plupart des IDS basés sur le réseau ne peuvent pas dire si une attaque a réussi ou non ; ils peuvent seulement discerner qu'une attaque a été lancée. Cela signifie qu'après qu'un IDS basé sur le réseau détecte une attaque, les administrateurs doivent enquêter manuellement sur chaque hôte attaqué pour déterminer s'il a effectivement été pénétré.

b. Les SDIs basés sur l'hôte (SDIH)

Les systèmes de détection d'intrusion basés sur l'hôte visent à collecter des informations sur l'activité d'un système unique particulier, ou hôte [Bace 1999].

Ces agents basés sur l'hôte, qui sont parfois appelés capteurs, seraient généralement installés sur une machine considérée comme susceptible d'attaques possibles. Le terme « hôte » désigne un individu ordinateur, donc un capteur séparé serait nécessaire pour chaque machine. Les capteurs fonctionnent en collectant des données sur les événements qui se déroulent sur le

système surveillé. Ces données sont enregistrées par des mécanismes du système d'exploitation appelés pistes d'audit

- **Avantages**
  - Les IDS basés sur l'hôte, avec leur capacité à surveiller les événements locaux d'un hôte, peuvent détecter des attaques qui ne peuvent pas être vues par un IDS basé sur le réseau.
  - Les IDS basés sur l'hôte ne sont pas affectés par les commutateurs réseaux.
  - Lorsque les IDS basés sur l'hôte fonctionnent sur des pistes d'audit du système d'exploitation, ils peuvent aider à détecter les chevaux de Troie ou d'autres attaques impliquant des violations d'intégrité logicielle.
- **Inconvénient**
  - Les IDS basés sur l'hôte sont plus difficiles à gérer, car les informations doivent être configuré et géré pour chaque hôte surveillé.
  - Les IDS basés sur l'hôte ne sont pas bien adaptés pour détecter les analyses de réseau ou toute autre surveillance de ce type qui cible un réseau entier, car l'IDS ne voit que les paquets réseau reçus par son hôte.
  - Les IDS basés sur l'hôte peuvent être désactivés par certains déni de service attaques.
  - Les IDS basés sur l'hôte utilisent les ressources informatiques des hôtes qu'ils surveillent, infligeant ainsi un coût de performance au systèmes surveillés

1.3.2.2 Le type d’analyse

Il existe deux approches principales pour analyser les événements afin de détecter les attaques : l’approche par signature et l’approche comportement. L’approche par signature, dans laquelle l'analyse cible un élément connu pour être « mauvais », est la technique utilisée par la plupart des systèmes commerciaux. L’approche comportementale, dans laquelle l'analyse recherche des schémas d'activité anormaux, continue de faire l'objet de nombreuses recherches. L’approche comportementale est utilisée de manière limitée par un certain nombre de SDI. Il existe des forces et des faiblesses associées à chaque approche.[Bace 1999]

a. Analyse par signature

L’analyse par signatures analysent les activités du système à la recherche d'événements correspondant à un modèle ou à une signature prédéfinie décrivant une attaque bien connue. Ils collectent le trafic réseau et procèdent ensuite à son analyse. L'analyse est basée sur une comparaison de motifs (pattern matching en français filtrage de pattern). Le système contient une base de données de modèles d'attaque et recherchera des similitudes avec eux et lorsqu'une correspondance est détectée, l'avertissement se déclenchera.

Ces systèmes sont vraiment efficaces pour détecter les attaques mais ils génèrent un grand nombre de faux positifs. Il faut donc que la période pendant laquelle ils sont régulés (tuning period) soit la plus courte possible.

Le bon fonctionnement d'un tel système dépend non seulement d'une bonne installation et configuration, mais aussi sur le fait que la base de données où les modèles d'attaque sont stockés est mise à jour régulièrement

- **Avantages**
  - Elle est très efficace pour détecter les attaques sans générer un nombre écrasant de fausses alarmes.
  - Elle peut diagnostiquer rapidement et de manière fiable l'utilisation d'un outil ou d'une technique d'attaque spécifique. Cela peut aider les responsables de la sécurité à prioriser les mesures correctives.
  - Elle peut permettre aux gestionnaires de système, quel que soit leur niveau d'expertise en sécurité, de suivre les problèmes de sécurité sur leurs systèmes, en lançant un processus de gestion des incidents
- **Inconvénients**
  - Elle ne peut détecter que les attaques qu'ils connaissent, elle doit donc être constamment mis à jour avec les signatures des nouvelles attaques.
  - Elle est pour la plupart conçus pour utiliser des signatures étroitement définies qui les empêchent de détecter des variantes d'attaques courantes. Les détecteurs de mauvaise utilisation basés sur l'état peuvent surmonter cette limitation, mais ne sont pas couramment utilisés dans les IDS commerciaux.

b. Analyse comportementale

Les modèles comportementaux sont apparus bien plus tard que les IDS à signatures. Ils ont pour principe la détection d'anomalies. Leur mise en œuvre comprend toujours une phase

d'apprentissage au cours de laquelle ils vont " découvrir " le fonctionnement " normal " des éléments surveillés. Une fois cet apprentissage effectué ces IDS signaleront les divergences par rapport au fonctionnement de référence, les modèles comportementaux peuvent être élaborés à partir d'analyses statistiques ou de techniques proches de l'intelligence artificielle[Bourouh and Kanoun 2017].

La principale caractéristique des IDS comportementaux est la détection des nouveaux types d'attaque, en effet ces IDS ne sont pas programmés pour reconnaître des attaques spécifiques mais signalent toute activité " anormale ". De ce fait une attaque ne doit pas nécessairement être connue d'avance ; dès lors qu'elle représente une activité anormale elle peut être détectée par l'IDS comportemental. Du fait même de leur conception ces IDS sont incapables de qualifier la criticité des attaques. De plus, ces IDS signaleront par exemple tout changement dans le comportement d'un utilisateur qu'il soit hostile ou non. De fréquents ajustements sont nécessaires afin de faire évoluer le modèle de référence de sorte qu'il reflète l'activité normale des utilisateurs et réduire le nombre de fausses alertes générées.

- **Avantages**
  - Elle détecte les comportements inhabituels et ont ainsi la capacité de détecter les symptômes d'attaques sans connaissance spécifique des détails.
  - Elle peut produire des informations qui peuvent à leur tour être utilisées pour définir des signatures pour l’analyse par signature.
- **Inconvénients**
  - Elle produit généralement un grand nombre de fausses alarmes dues aux comportements imprévisibles des utilisateurs et des réseaux.
  - Elle nécessite souvent des tests d'entraînements étendue d'enregistrements d'événements système afin de caractériser les modèles de comportement normaux.

1.3.2.3 Le type de réponse

Une fois que les SDIs ont obtenu des informations sur les événements et les ont analysées pour trouver des symptômes d'attaques, ils génèrent des réponses. Certaines de ces réponses impliquent la communication des résultats et des constatations à un endroit prédéfini. D'autres impliquent des réponses automatisées plus actives. Bien que les chercheurs soient tentés de sous-estimer l'importance de bonnes fonctions de réponse dans les SDIs, elles sont en réalité

très importantes. Les SDIs commerciaux prennent en charge un large éventail d'options de réponse, souvent classées en réponses actives, réponses passives ou un mélange des deux.

a. Réponses passives

Dans ce type d'IDS, le responsable de la sécurité ou les utilisateurs du système sont informés de ce qui s'est passé à travers des alarmes, notification, des traps SNMP et des plugins [Bace and Mell 2001]

Il est également utile d'alerter l'administrateur du site à partir duquel l'attaque a été lancée, mais il est possible que l'attaquant puisse surveiller l'email de l'organisation ou qu'il ait utilisé une fausse IP pour l'attaque. Dans ce cas, il serait inutile de l'alerter.

b. Réponses actives

Les réponses actives sont des actions automatiques prises lorsque certains types d'intrusions sont détectés. Deux catégories différentes peuvent être définies :

- Collecte d'informations supplémentaires : il consiste à incrémenter le niveau de sensibilité du capteur afin d'obtenir plus d'indices de l'attaque possible (par exemple, attraper tous les packages de la source qui a lancé l'attaque, au cours d'une certaine période).
- Modification de l'environnement : une autre réponse active peut arrêter l'attaque ; Par exemple, dans le cas d'une connexion TCP, la session peut être fermée en injectant des segments TCP RST à l'attaquant et à la victime, ou de filtrer l'adresse IP de l'intrus ou du port attaqué, au routeur d'accès ou au pare-feu dans afin d'éviter les attaques futures

1.3.2.4 Le temps de détection

Deux groupes principaux peuvent être identifiés, ceux qui détectent des intrusions en temps réel (en ligne) et ceux qui traitent des données d'audit avec un délai (hors ligne), ce qui ne signifie pas une heure réelle.

Certains systèmes de détection en ligne peuvent également effectuer une détection hors ligne sur des données d'audit historiques. Ce type de systèmes combinant les deux types de temps de détection est appelé hybride. [Syngress 2003]

1.3.3 Points Forts et Limites des SDIs

Bien que les systèmes de détection d'intrusion soient un ajout précieux à l'infrastructure de sécurité d'une organisation, il y a des choses qu'ils ont bien, et d'autres choses qu'ils ne font pas bien. Au fur et à mesure que vous planifiez la stratégie de sécurité pour les systèmes de votre organisation, il est important que vous compreniez ce que les IDS devraient faire confiance à faire et quels objectifs peuvent être mieux servis par d'autres types de mécanismes de sécurité[Amudha et al. 2013]

a. Points Forts

Les systèmes de détection d'intrusion effectuent bien les fonctions suivantes :

- Surveillance et analyse des événements système et des comportements utilisateur ;
- Tester les états de sécurité des configurations du système ;
- Basculer l'état de sécurité d'un système, puis suivant toute modification apportée à ce système ;
- Reconnaître les habitudes des événements système correspondant à des attaques connues ;
- Reconnaître les modes d'activité qui varient statistiquement de l'activité normale ;
- Gérer les mécanismes d'audit et de journalisation du système d'exploitation et les données qu'ils génèrent ;
- Alerter le personnel approprié par des moyens appropriés lorsque des attaques sont détectées. ;
- Mesurer l'application des politiques de sécurité codées dans le moteur d'analyse ;
- Fournir des stratégies de sécurité des informations par défaut ;
- Permettre aux experts non-sécurité d'effectuer des fonctions de surveillance de sécurité importantes. ;

b. Limites

Les systèmes de détection d'intrusion ne peuvent pas effectuer les fonctions suivantes :

- Compenser les mécanismes de sécurité faibles ou manquants dans l'infrastructure de protection. Ces mécanismes comprennent des pare-feu, d'identification et d'authentification, de cryptage de liaison, de mécanismes de contrôle d'accès et de détection et d'éradication du virus détectant ;

- Instantanément la détection, la déclaration et la réponse à une attaque, lorsqu'il existe un réseau lourd ou une charge de traitement ;
- Détecter des attaques ou des variantes nouvellement publiées d'attaques existantes ;
- Répondre efficacement aux attaques lancées par des attaquants sophistiqué ;
- Enquêter automatiquement aux attaques sans intervention humaine ;
- Résister aux attaques destinées à la vaincre ou à les contourner ;
- Compenser les problèmes liés à la fidélité des sources d'information ;
- Traiter efficacement avec les réseaux commutés ;

1.4 Conclusion

Dans ce chapitre nous avons expliqué ce qu'est la sécurité informatique dans une première partie, ensuite nous avons donné un petit aperçu de la politique de sécurité informatique, ses exigences et objectifs pour enfin parles des attaques et des menaces au sein d'un système informatique, nous avons également illustré les différents dispositifs de sécurité informatique.

Dans une deuxième partie, nous avons présenté les systèmes de détection d'intrusion puis nous avons décrit les différents types de SDI classés selon la source d'information, le type d'analyse et le type de réponse, nous avons enfin clôturé ce chapitre en citant quelques points forts et limites des SDIs

Chapitre 2 : L’apprentissage automatique et les SDIs

2.1 Introduction

La détection d'intrusion réseau traditionnelle est basée sur des règles, où chaque type d'attaques nécessite d'être étudié manuellement au préalable et le processus de détection est effectué sur la base des signatures d'attaques où des règles de sécurité sont associées. Cependant, ce type d'approches de détection n'a pas été suffisamment adapté à l'échelle de réseau à croissance rapide et ne peut pas faire face à des attaques de volume, de complexité et de volatilité croissantes.

Pour la sécurité des réseaux à grande échelle, un système de détection d'intrusion réseau doit non seulement être capable d'identifier rapidement et correctement les attaques connues, mais aussi être suffisamment adaptatif et intelligent pour les attaques inconnues et évoluées, pour lesquelles l'Apprentissage Automatique (AA)(en anglais Machine Learning (ML))est entré en jeu.

Contrairement aux approches basées sur des règles, les solutions basées sur le AA peuvent faire une hypothèse de détection abstraite en apprenant les distinctions sous-jacentes entre le trafic normal et les attaques réseau et peuvent identifier les attaques réseau sans nécessiter beaucoup d'intervention humaine [Laskov*et al.*, 2005].

Les réseaux de neurones ont plusieurs avantages dans la mise en œuvre d'un système de détection d'intrusion. Ils sont très efficaces et rapides dans la tâche de classification. Ils sont capables d'apprendre et d'identifier facilement les nouvelles menaces qui leur sont soumises. Les réseaux de neurones sont capables de traiter les données incomplètes, imprécises et provenant de sources multiples. La rapidité naturelle des réseaux de neurones permet de réduire les dommages lorsque la menace est détectée. La flexibilité qu'offrent les réseaux de neurones est également l'un des atouts pour la détection d'intrusion.

Dans ce chapitre nous allons présenter l'apprentissage automatique, les Réseaux de Neurones (RN) avec leurs différents types et modèles ainsi que l'apprentissage profond. Puis nous verrons quelques travaux sur les SDIs de la littérature faisons un recours aux RNs.

2.2 Apprentissage automatique

L'apprentissage automatique est un domaine de l'intelligence artificielle dans lequel il est question de concevoir des modèles informatiques capables d'apprendre à partir d'un ensemble de données donné avec une intervention humaine minimale. C'est un ensemble de méthodes utilisées pour détecter automatiquement des modèles dans les données, puis utiliser les modèles extraits pour prédire les données futures ou effectuer d'autres types de tâches de prise de décision. Un modèle de gain de machine peut être soit prédictif s'il fait des prévisions pour les conditions futures, soit descriptif si son objectif est d'acquérir des connaissances à partir des données ou être à la fois prédictif et descriptif. En utilisant la théorie des statistiques dans la construction des modèles mathématiques, la tâche principale des algorithmes d'apprentissage automatique est d'extrapoler l'inférence à partir d'un échantillon donné.[Murphy, 2012].

Les techniques de ML se composent de deux étapes [Murphy, 2012] :

1) Phase d'apprentissage

Premièrement, la machine apprend en explorant des modèles. La liste des caractéristiques utilisées pour résoudre la tâche est connue sous le nom de vecteur de caractéristiques. Un vecteur de caractéristiques est utilisé pour résoudre le problème et il s'agit d'un sous-ensemble de l'ensemble de données. La machine utilise une sorte d'algorithmes pour convertir la réalité en modèle. Par conséquent, l'étape d'apprentissage est utilisée pour encapsuler les données dans un modèle.

2) Phase de test

Une fois le modèle créé, nous devons tester ce modèle en utilisant de nouvelles données. Ces nouvelles données seront converties en un vecteur de caractéristiques ; il est appliqué à un modèle et donne une prédiction. Il n'est pas nécessaire de changer les règles ou de rééduquer le modèle. Nous pouvons utiliser le modèle précédent pour tirer des conclusions sur les nouvelles données.

2.2.1 Techniques d'apprentissage automatique

L'apprentissage automatique se divise en quatre catégories en fonction de la nature de l'expérience. Ils sont :

a. Apprentissage supervisé

Cette approche a pour objectif de concevoir un modèle reliant des données d'apprentissage à un ensemble de valeurs de sortie (un comportement).[Larochelle 2009]

- Réseau bayésien
- Régression du processus gaussien
- Apprentissage paresseux
- Arbres de décision
- Algorithme du voisin le plus proche
- Réseaux informationnels flous
- Prise en charge des machines vectorielles
- Agrégation Bootstrap
- Régression linéaire
- Classificateur naïf de Bayes
- Modèles cachés de Markov
- Réseau de neurones artificiels Forêts aléatoires

b. Apprentissage non supervise

L'apprentissage non supervisé utilise des données non étiquetées et trouve des structures, des modèles, des connaissances dans les données. Ils sont classés en partitionnement(clustering) et en association.[Larochelle 2009]

Divers algorithmes entrent dans cette catégorie :

- Algorithme a priori
- Algorithme des K-moyennes
- Partitionnement à liaison unique
- Partitionnement flou
- Facteur de valeur aberrante locale

c. Apprentissage par renforcement

Les données en entrée sont les mêmes que pour l'apprentissage supervisé, cependant l'apprentissage est guidé par l'environnement sous la forme de récompenses ou de pénalités données en fonction de l'erreur commise lors de l'apprentissage.[Larochelle 2009]

Les algorithmes utilisés dans cette catégorie :

- Q-Learning
- Difference Temporelle (DT)
- Processus de décision de Markov
- Réseaux accusatoires profonds

d. Apprentissage semi-supervisé

Ce type d'apprentissage utilise un ensemble de données étiquetées et non-étiquetées. Il se situe ainsi entre l'apprentissage supervisé qui n'utilise que des données étiquetées et l'apprentissage non-supervisé qui n'utilise que des données non-étiquetées. Ainsi, Il a été démontré que l'utilisation de données non-étiquetées, en combinaison avec des données étiquetées, permet d'améliorer significativement la qualité de l'apprentissage. Un autre intérêt provient du fait que l'étiquetage de données nécessite l'intervention d'un expert.[Larochelle 2009]

Quelques algorithmes :

- Co-training
- Séparation à faible densité
- Modèles génératifs
- Méthodes basées sur des graphes

2.3 Apprentissage profond

L'apprentissage profond est un sous-champ de l'apprentissage automatique des représentations de données d'apprentissage. Exceptionnel efficace pour l'apprentissage des modèles.

Les algorithmes de l'apprentissage approfondi tentent d'apprendre (à plusieurs niveaux) la représentation en utilisant une hiérarchie de plusieurs couches, Si on fournit au système des tonnes d'informations, il commence à le comprendre et à réagir de manière utile.

La plupart des modèles d'apprentissage profond modernes sont basés sur une approche de réseau de neurones artificiels (2.4).

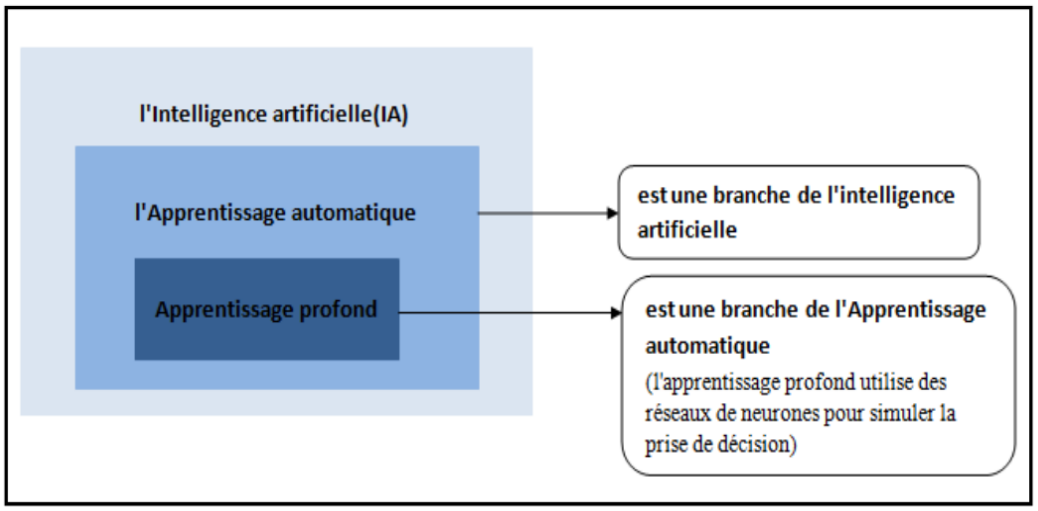

Figure 2-1 : La relation entre trois branches de l'intelligence artificielle (IA), (AA) et (APP)[Berri 2019]

2.4 Réseaux de Neurones Artificiels (RNA)

Les réseaux de neurones artificiels ont fait l'objet d'un intérêt soutenu de nombreux travaux depuis plus d'une vingtaine d'année grâce à leur capacite à résoudre des problèmes non linéaires par apprentissage.

Les réseaux de neurones artificiels sont des programmes informatiques qui sont biologiquement inspirés des réseaux de neurones biologiques et conçus pour simuler la façon dont le cerveau humain traite l'information[Agatonovic-Kustrin and Beresford, 2000].

Un neurone artificiel (formel) est un processeur élémentaire stimulé par des neurones qui le précèdent, qu'on appellera ses *entrées*, à chacune de ces entrées est associé un *poids* représentatif de la force de la connexion w et, en fonction de cette stimulation, on lui attribue une valeur qu'on appellera sa *sortie*. Qui se ramifie ensuite pour alimenter un nombre variable de neurones avals [Bourouh and Kanoun, 2017].

.

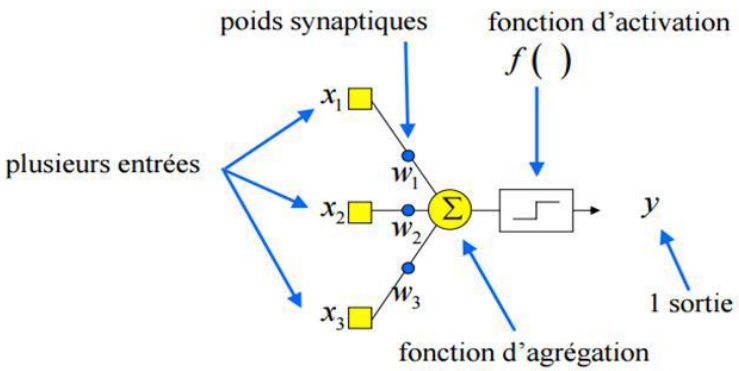

Figure 2-2 Structure d'un réseau de neurone formel[Patrice 2009]

Pour introduire les réseaux de neurones, nous allons commencer par présenter l'architecture des réseaux de neurones et ensuite par présenter un modèle composé d'un seul neurone, appelé modèle du perceptron. Celui-ci va nous permettre de mettre en évidence les mécanismes de base de tout réseau de neurones.

2.4.1 Architecture des réseaux de neurones

On distingue deux structures de réseau, en fonction du graphe de leurs connexions, c'est-à-dire du graphe dont les nœuds sont les neurones et les arêtes sont les connexions entre ceux-ci :

- Les réseaux de neurones statiques (ou acycliques, ou non bouclés).
- Les réseaux de neurones dynamiques (ou récurrents, ou bouclés).

a. Les réseaux de neurones non bouclés

Un réseau de neurones non bouclé est un ensemble de neurones « connectés » entre eux, l'information circulent des entrées vers les sorties sans « retour en arrière ». On peut alors représenter le réseau par un graphe acyclique dont les nœuds sont les neurones et les arêtes sont les connexions entre ceux-ci. Si l'on se déplace dans le réseau, à partir d'un neurone quelconque, en suivant les connexions et en respectant leurs sens, on ne peut pas revenir au neurone de départ. La représentation de la topologie d'un réseau par un graphe est très utile, notamment pour les réseaux bouclés, les neurones qui effectuent le dernier calcul de la composition de fonctions sont les neurones de sortie; ceux qui effectuent des calculs intermédiaires sont les neurones cachés [Hamid and Tassadit 2013].

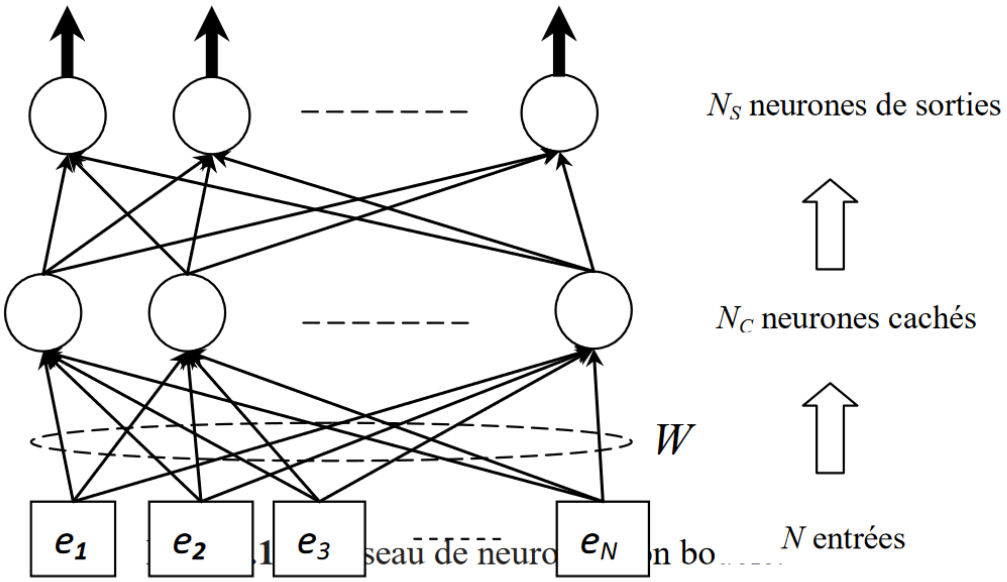

Figure 2-3 Architecture d'un réseau de neurones non bouclé[Hamid and Tassadit 2013]

b. Les réseaux de neurones bouclés

L'architecture la plus générale pour un réseau de neurones est le « réseau bouclé », dont le graphe des connexions est cyclique, lorsqu'on se déplace dans le réseau en suivant le sens des connexions, il est possible de trouver au moins un chemin qui revient à son point de départ (un tel chemin est désigné sous le terme de « cycle »). La sortie d'un neurone du réseau peut donc être fonction d'elle-même, cela n'est évidemment concevable que si la notion de temps est explicitement prise en considération. Ainsi, à chaque connexion d'un réseau de neurones bouclé (ou à chaque arête de son graphe) est attaché, outre un poids comme pour les réseaux non bouclés, un retard, multiple entier (éventuellement nul) de l'unité de temps choisie. Une grandeur, à un instant donné, ne pouvant pas être fonction de sa propre valeur au même instant, tout cycle du graphe du réseau doit avoir un retard non nul. Les connexions récurrentes ramènent l'information en arrière par rapport au sens de propagation défini dans un réseau multicouches. Ces connexions sont le plus souvent locales. Pour éliminer le problème de la détermination de l'état du réseau par bouclage, on introduit sur chaque connexion « en retour » un retard qui permet de conserver le mode de fonctionnement séquentiel du réseau.[Bourouh and Kanoun 2017]

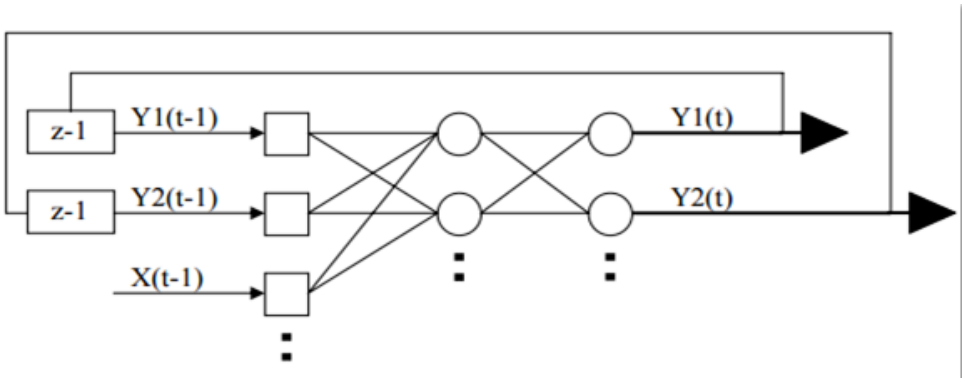

Figure 2-4 Architecture d'un réseau de neurone bouclé[Bourouh and Kanoun 2017]

2.4.2 Modèle de Réseaux de neurones

2.4.2.1 Le modèle du perceptron

Dans sa version la plus simple, le perceptron est un réseau de neurones composé de seulement un neurone, qui prend en entrée *n* données binaires. Chacune de ses entrées *i* est pondérée par un poids noté W_i. Le neurone peut prendre les états "1" ou "0" (respectivement actif ou non-actif) en fonction de ses entrées pondérées et d'un biais noté $\beta \in \mathbb{R}$. Cet état représente la sortie du modèle. Il est donc possible de représenter le perceptron comme une fonction paramétrique $f_\theta : \{0,1\}^n \rightarrow \{0,1\}$avec θl'ensemble de ses paramètres, c'est-à-dire le biais βet les poids $\mathbf{W}= (w_1, \ldots, w_n)$. La sortie d'un perceptron pour un vecteur $x \in \{0,1\}^n$en entrée est calculée tel que :

$$f(\boldsymbol{x}, \mathbf{W}) = H(z(\boldsymbol{x}, \mathbf{W}) + \beta)) \qquad (2\text{-}1)$$

Avec$H(t)$ la fonction de Heaviside définie pour tout $t \in \mathbb{R}$ comme $H(t) = 1_{\{t>0\}}$ et $z(\boldsymbol{x},\mathbf{W})$ la somme pondérée des entrées :

$$z(x, W) = W^T x = \sum_{j=1}^{n} w_j x_j) \qquad (2\text{-}2)$$

Intuitivement, les poids $w_1, \ldots, w_n$ représentent l'importance accordée à chaque entrée pour l'activation du perceptron. Pour rappel, 1_A est la fonction indicatrice qui est égale à 1 si la

condition A est vérifiée et 0 sinon. Le biais βpeut être vu comme l'ajoutd'un seuil à la difficulté d'activation du perceptron. En effet, si la somme pondérée $z(\boldsymbol{x},\mathbf{W})$ dépasse β (l'opposé du biais), le perceptron s'active, sinon il reste inactif.

Pour simplifier les notations, nous allons inclure le biais _ dans le vecteur de poids en ajoutant une constante en entrée x_0= 1 (le biais devient donc w_0). Nous obtenons la fonction paramétrique $f_\theta : \{0,1\}^{n+1} \times \mathbb{R}^{n+1} \rightarrow \{0,1\}$telle que :

$$f(x,W) = H\big(z(x,W)\big) = H(\sum_{j=0}^{n} w_j\, x_j. \qquad (2\text{-}3)$$

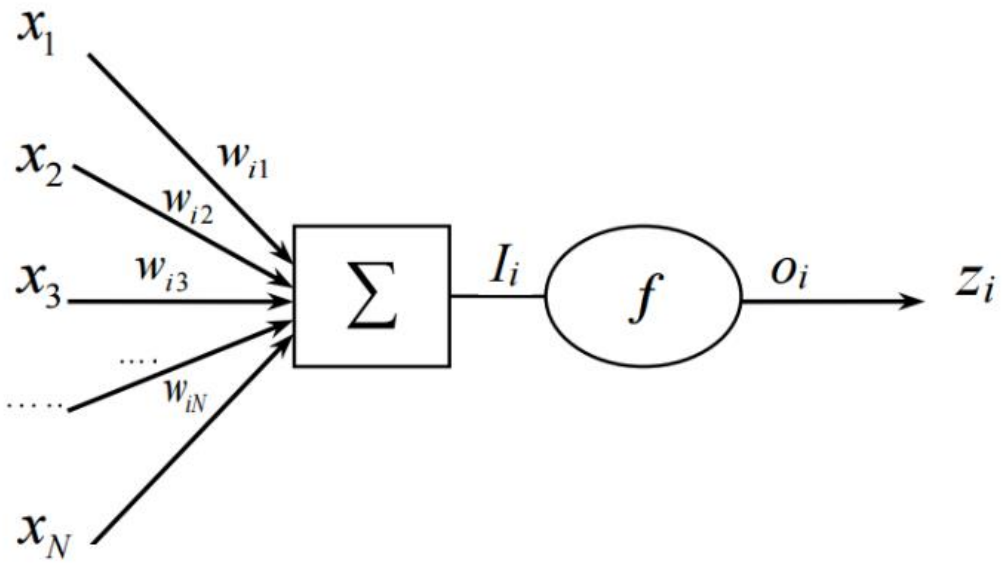

Figure 2-5 Modèle du perceptron[SABRY et al. 2015]

2.4.2.2 Le perceptron multicouche

Le perceptron multicouche, appelé aussi MLP (pour Multi-layer Perceptron), est un réseau de neurones plus général que le perceptron. Il est composé d'une multitude de neurones interconnectés et organisés en couches successives. Un MLP peut être représenté par un graphe acyclique dans lequel chaque nœud représente un neurone. Les arcs orientés représentent les relations entre chaque neurone : un arc du nœud *i* au nœud *j* signifie que le neurone *j* prend la valeur d'activation du neurone *i* en entrée. La Figure ci-dessous montre une représentation graphique d'un MLP avec 3 couches ayant respectivement 5, 4, 3 neurones[Hardy 2019].

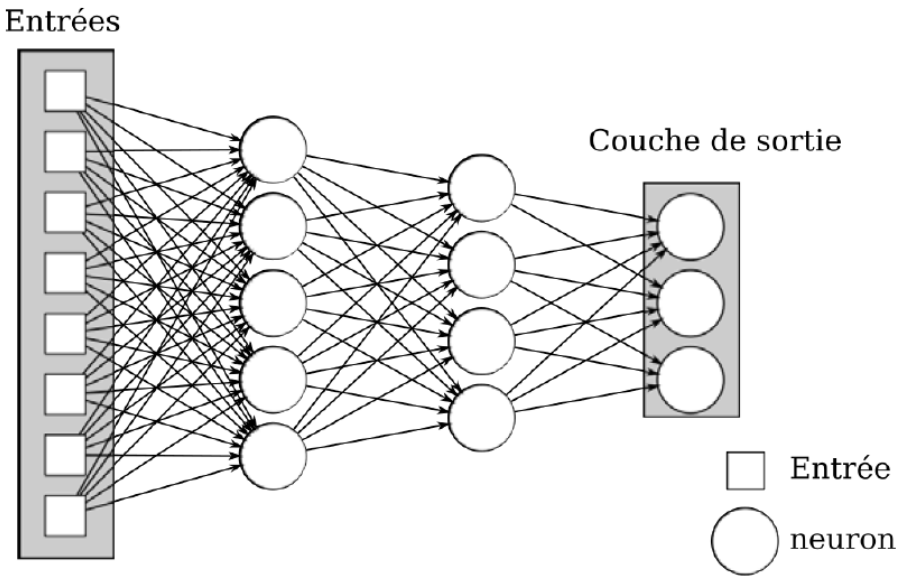

Figure 2-6Modèle du perceptron multicouche[Hardy 2019]

Comme illustré dans la figure 2-6, chaque neurone de la première couche prend en entrée, l'entrée du MLP. Chaque couche suivante reçoit en entrée les valeurs d'activation de la couche précédente (c'est-à-dire, le vecteur contenant les valeurs de chaque neurone de la couche précédente). La sortie du MLP est composée de la valeur d'activation de chaque neurone de la dernière couche, appelée couche de sortie. Le vecteur d'activation de la couche *l*, composé de *s* neurones peut être calculé suivant le vecteur d'entrées $\mathbf{e} \in \mathbb{R}^p$ (Les entrées du MLP ou le vecteur d'activation de la couche précédente), de la manière suivante :

$$a^{(l)} = f^{(l)}\left(\boldsymbol{e}, W^{(l)}, \beta^{(l)}\right) = \phi^{(l)}(W^{(l)}\boldsymbol{e} + \beta^{(l)}) \qquad (2\text{-}4)$$

Avec $W = (w_1,\ldots,w_s)^T \in \mathbb{R}^{s\times p}$ et $\beta^{(l)} = (\beta_1,\ldots,\beta_s)^T \in \mathbb{R}^s$ avec respectivement $\mathbf{w}_i$.
Le vecteur des poids du neurone *i* et β_i son biais. La fonction $\phi^{(l)}$ est une fonction d'activationappliquée individuellement à chaque neurone de la couche *l*. Dans la section précédente, nous avons vu la fonction de Heaviside et la fonction sigmoïde. Dans unMLP, tous les neurones d'une même couche ont la même fonction d'activation $\phi^{(l)}$.

Le MLP est représenté par une fonction *f* qui prend en entrée des données$x \in \mathbb{R}^n$et un ensemble de paramètres $\theta = \{W^{(l)}, \beta^{(l)} | l \in \{1,\ldots,L\}\}$ correspondant à l'ensemble des matrices $W^{(l)}$ et des vecteurs $\beta^{(l)}$ pour toutes les couches $l = 1, \ldots, L$, et donne en sortie $y \in$

$\mathbb{R}^m$Rm. Comme décrit dans l'équation d'un perceptron (voir équation1.1), il est possible d'inclure le vecteur$\beta^{(l)}$ dans la matrice $W^{(l)}$ en ajoutant une entrée constante pour chaque couche (ce qui rajoute une colonne à chaque matrice $W^{(l)}$).La fonction f est une composition de fonctions $f^{(l)}$ associées à chaque couche l du réseau. Par exemple, avec un MLP à trois couches, nous avons :

$$y = f(x, \theta) = f^{(3)}(f^{(2)}(f^{(1)}(x, W^{(1)}), W^{(2)}), W^{(3)}) \quad (2\text{-}5)$$

Un MLP avec un nombre de couches plus grand ou égal à 2 est un approximateur universel de fonctions, c'est-à-dire, qu'il est capable de représenter toutes sortes de fonctions si ses paramètres sont correctement ajustés (sous certaines conditions sur la fonction d'activation des couches cachées. Pour illustrer ceci, nous prenons l'exemple de la fonction OU EXCLUSIF avec deux entrées. Soit un MLP avec deux entrées et deux couches, composées de 2 et 1 neurones. Les paramètres du MLP sont :

$$W^{(1)} = \begin{bmatrix} 0 & 1 & 1 \\ -1 & 1 & 1 \end{bmatrix} \quad (2\text{-}6)$$

$$W^{(2)} = [0 \quad 1 \quad -2] \quad (2\text{-}7)$$

Nous utilisons des unités linéaires rectifiées (appelé ReLU) utilisées régulièrement dans les réseaux de neurones modernes. Ce type de neurone utilise la fonction d'activation $\phi(z) = \max\{0, z\}$(Voir figure ci-dessous). La totalité du MLP est représentée graphiquement sur la Figure suivante.

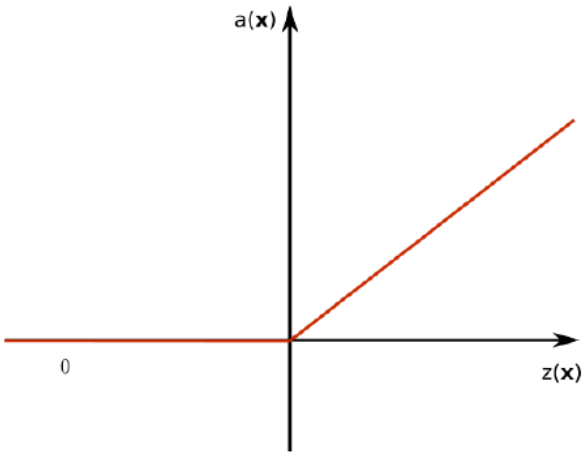

Figure 2-7 Fonction d'activation des unités linéaires rectifiées

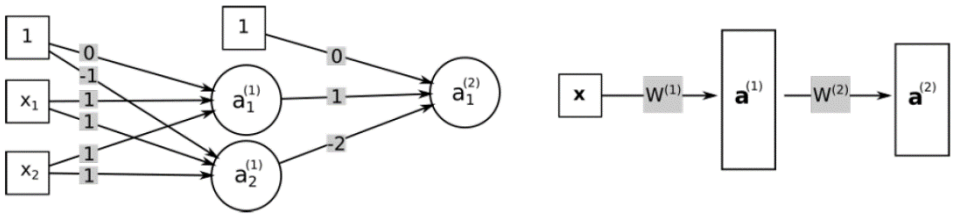

Figure 2-8 Réseau de neurones représentant la fonction OU EXCLUSIF avec deux représentations graphiques[Hardy 2019]

À gauche, chaque neurone est représenté par un cercle. Les poids sont représentés sur les connexions entre les neurones. De même, les biais représentés par la connexion entre une constante (carrés) et chaque neurone. À droite, dans ce style graphique, chaque couche est représentée par un rectangle. Les matrices de paramètres peuvent être indiquées sur les connexions entre les couches. L'avantage de cette seconde représentation est d'être plus compacte que la première.

Les couches intermédiaires de ce MLP transforment l'espace de représentation des données d'entrée comme montré dans Figure 2-6. Ce nouvel espace de représentation permet de séparer linéairement les sorties de la fonction visée (la fonction OUEXCLUSIF dans cet exemple). Les couches intermédiaires peuvent être vues comme des représentations des entrées à plus haut niveau.

2.4.2.3 Réseaux de neurones à compétition (carte de Kohonen)

Les réseaux compétitifs sont basés sur un mode d'apprentissage qui stimule la compétition entre les neurones dans le but de favoriser l'adaptation des neurones les plus réactifs aux données présentées en entrée du réseau.[Farid 2006]

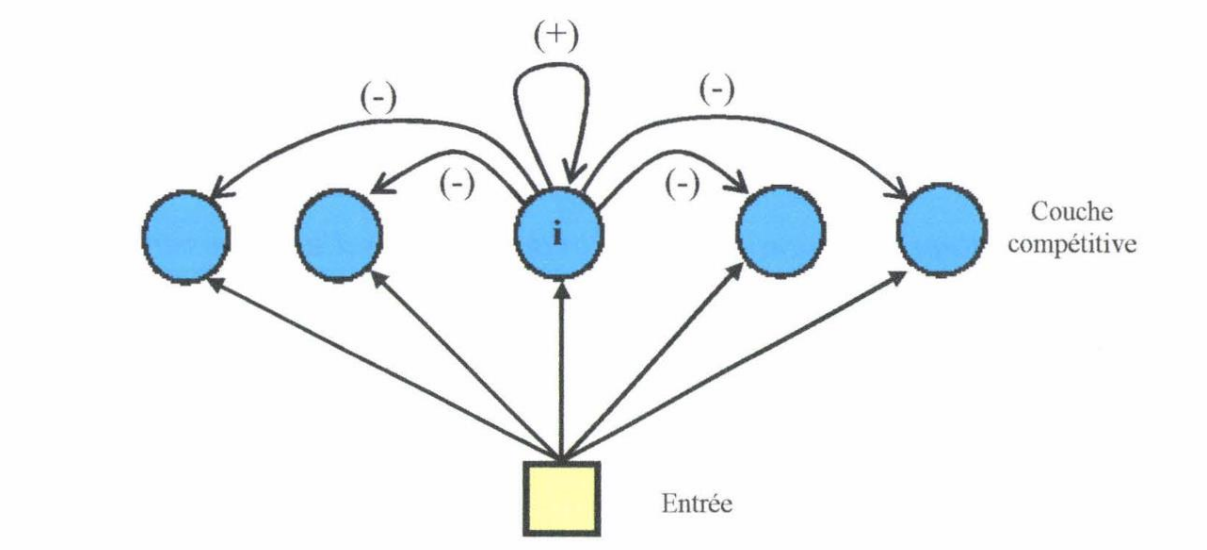

Figure 2-9 Modèle de réseau compétitif[Farid 2006]

Dans leur forme la plus simple, les réseaux compétitifs sont souvent constitués de deux couches. Une couche de neurones sur laquelle sont présentées en entrée les données issues de la base d'apprentissage, il s'agit de la couche d'entrée du réseau. Unecouche de sortie qui permet de restituer globalement les résultats des calculs fournis par le processus compétitif Figure 2-9 Cette couche de sortie est également appelée couche compétitive. Le processus compétitif entre les neurones est réalisé traditionnellement grâce à des connexions latérales inhibitrices. Ces connections latérales entre neurones voisins permettent à chaque neurone de sortie augmenter naturellement son niveau d'activation tout en inhibant le niveau d'activation de ses voisins. Une telle architecture, couplée à un apprentissage compétitif, permet de respecter, au cours de sa phase d'adaptation, l'organisation topographique des données dans l'espace d'entrée. [Farid 2006]

2.4.2.3.1 Architecture et topologie

L'architecture d'une carte auto-organisatrice de Kohonen, tel qu'elle est présentée sur la Figure 2-10.est composée de trois couches : une couche d'entrée, une couche de sortie et la couche compétitive souvent appelée couche de Kohonen. Sur cette dernière, on trouve un

ensemble de neurones interconnectés via des liaisons de voisinage. Chaque neurone sur la carte est caractérisé par son vecteur de poids ou vecteur de pondération. En termes de la projection de l'environnement externe sur la carte, le vecteur de poids représente la position du neurone sur l'espace d'entrée. Par conséquent, la dimension du vecteur de poids est égale à la dimension de l'espace d'entrée. Les échantillons prélevés de cet espace de données seront présentés sur la couche d'entrée élément par élément. Ensuite, ils seront envoyés vers tous les neurones de la couche compétitive. Les distances calculées par ces derniers seront envoyées à leur tour vers le comparateur de la couche de sortie pour déterminer la classe associée à l'échantillon sur cette couche de sortie (voir la Figure 2-10).[Abadi 2018]

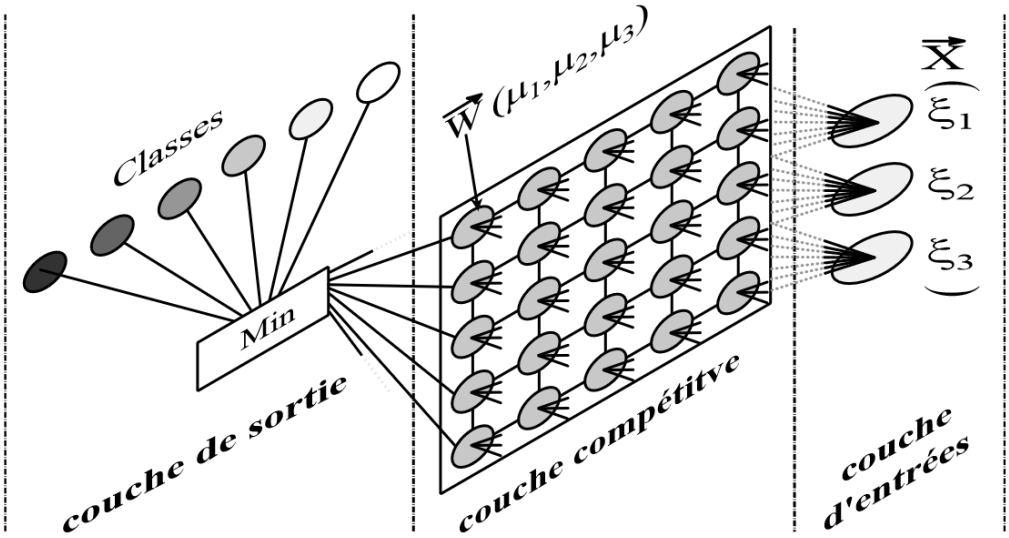

Figure 2-10 Architecture de la carte autoorganisés de Kohonen[Abadi 2018]

Au cours de la phase d'apprentissage de la carte auto-organisatrice, les échantillons en entrée sont injectés d'une façon itérative. Chaque échantillon est traité à part par le réseau sans attendre un retour de l'extérieur. On parle donc du principe de l'apprentissage non-supervisé.

Les échantillons de l'espace d'entrée sont sélectionnés d'une manière aléatoire au cours de cette phase et permettent de représenter statistiquement l'espace d'entrée. L'objectif de la carte auto-organisatrice est d'isoler et retrouver les relations topologiques des vecteurs d'entrée en les projetant sur la couche compétitive de la carte auto-organisatrice. La création de cette projection s'effectue par l'adaptation des vecteurs de poids du neurone élu vainqueur ainsi que ses voisins. Cela permet de créer des clusters représentant la majorité des classes

existantes avec une densité neuronale relativement similaire à celle de l'espace d'entrée, tout en gardant la même topologie de distribution de ces neurones sur la carte. Les neurones appartenant au même cluster ont les poids très proches même si parfois physiquement peuvent être éloignés dans la carte des neurones.

Chaque neurone est identifié par sa position topologique sur la carte. A base de cette identité, on peut calculer le taux de voisinage entre les neurones selon leur topologie de voisinage(connexions inter-neuronales) ainsi que la topologie de distribution des neurones sur la couche compétitive.[Abadi 2018]

2.4.2.3.2 Algorithme Self- Organizing Map (SOM) de Kohonen

L'algorithme SOM se base sur une approche d'apprentissage compétitif non-supervisé.[Cherif 2013]. L'objectif est d'adapter les paramètres libres de la carte auto-organisatrice, représentés par les vecteurs de poids de ses neurones, à la topologie de l'espace d'entrée, tout en gardant la disposition de voisinage des neurones définie au préalable. Cela s'effectue à base d'un algorithme itération supervisé. Les étapes de cet algorithme sont les suivantes :

1. Initialiser les paramètres de la carte auto-organisatrice : les vecteurs de poids Wl ,k , le rayon de voisinage R (t) et le taux d'apprentissage α(t)
2. Envoyer un vecteur d'entrée, associé à un échantillon, sur la couche d'entrée
3. Calculer la distance entre le vecteur d'entrée et les vecteurs de poids de chaque neurone sur la couche compétitive
4. Élire le neurone gagnant correspondant au neurone ayant la distance la plus petite.
5. Mettre à jour les vecteurs de poids du neurone gagnant ainsi que ceux de ses voisins d'ordre inférieur au rayon de voisinage
6. Procéder à la réduction du rayon de voisinage R (t) et du taux d'apprentissage α(t). Ensuite, reprendre l'algorithme à partir de l'étape (2)

Après plusieurs itérations des étapes de 2 à 6, la carte s'auto-adapte pour converger à une présentation réduite de l'espace d'entrée.

On remarque que le déroulement de l'algorithme de Kohonen, tel qu'il est présenté ci-dessus, passe par trois phases : une phase d'initialisation, une phase d'apprentissage et enfin une phase de rappel.

i. Phase d'initialisation

La phase d'initialisation est exécutée au départ. Cette phase consiste à initialiser les valeurs des éléments de vecteur de poids de chaque neurone sur le réseau. Cette initialisation peut influencer le temps de convergence de la couche compétitive vers la topologie de l'espace d'entrée. Dans la littérature, on trouve principalement trois types d'initialisation [Su et al. 1999]qui sont : initialisation par principaux composants de la base d'apprentissage "Principal Components Initialization (PCI) ", initialisation aléatoire "Random Initialization (RI) "et initialisation linéaire "Linear Initialization (LI) L'initialisation PCI consiste à initialiser chaque vecteur de poids par un vecteur choisi aléatoirement de la base d'apprentissage. L'initialisation RI consiste à initialiser les vecteurs des poids par des valeurs calculées à base d'une fonction linéaire d'une manière aléatoire. L'initialisation LI consiste à initialiser le vecteur de poids de chaque neurone avec des valeurs calculées d'une façon linéaire en fonction de sa position sur la grille.

ii. Phase d'apprentissage

La phase d'apprentissage est la phase de convergence de la couche compétitive vers une topologie similaire à celle de l'espace d'entrée. Pendant cette phase, les vecteurs de poids seront adaptés aux vecteurs de chaque échantillon de la base d'apprentissage qui seront présentés un par un, sur la couche d'entrée, pendant chaque itération. Une itération d'apprentissage passe par deux phases : la phase de compétition et la phase d'adaptation. [Abadi 2018].

iii. Phase de rappel

La phase de rappel est souvent appelée phase de décision ou phase de classification. En effet, c'est la phase d'utilisation de la carte auto-organisatrice de Kohonen une fois entrainée. Après la fin de la phase d'apprentissage, le réseau est prêt à réaliser des classifications des données de l'espace d'entrée réel.[Abadi 2018]

2.5 Les Réseaux de neurones profonds

Dans les algorithmes d'apprentissage classiques, des caractéristiques doivent être extraites des données brutes afin d'effectuer la tâche d'apprentissage. Le but étant d'avoir une représentation plus haut niveau des données. L'extraction de caractéristiques à partir des données brutes demande des bonnes connaissances sur celles-ci et sur la tâche d'apprentissage, ainsi qu'un travail d'ingénierie pour adapter les méthodes d'extraction. Cette opération est relativement coûteuse à la mise en place, dépend du contexte et une mauvaise

extraction des caractéristiques mène à de très mauvaises performances en termes d'apprentissage. L'idée des architectures profondes consiste à intégrer cette extraction de caractéristiques, normalement faite "à la main", par un processus d'apprentissage dans les premières couches du réseau de neurones.

Dans la session 2.4.2.2, nous avons vu que les couches intermédiaires permettent de transformer la représentation des données d'entrée en une représentation plus haut niveau.

Durant la phase d'apprentissage, chaque couche d'un MLP apprend une représentation de son entrée qui doit être intéressante pour les couches suivantes. Les informations contenues dans chacune de ces couches vont devenir de plus en plus haut niveau.

Le terme profond réfère donc au nombre de couches des réseaux de neurones profonds entre l'entrée et la sortie. Un réseau avec une seule couche cachée est appelé réseau peu profond, un réseau avec plus de 2 couches cachées est dit profond. De nos jours, il est possible de trouver des réseaux avec une centaine, voir un millier de couches pour les plus profonds [Szegedy*et al.*, 2014].

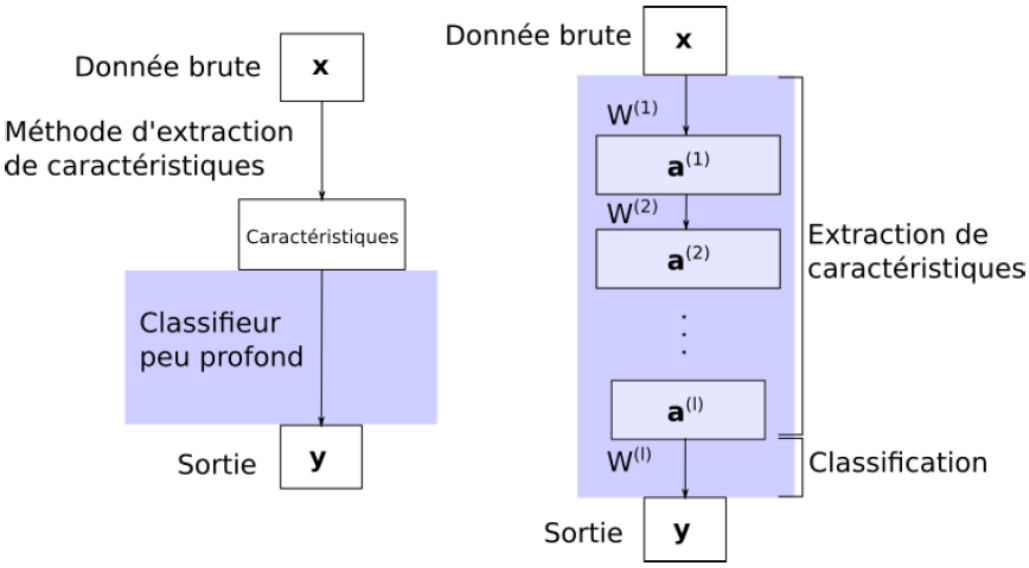

Figure 2-11 Différence entre la Méthode de l'apprentissage automatique et l'apprentissage profond[Hardy 2019]

La Figure 2-11 illustre la différence entre l'apprentissage automatique classique (à droite) et l'apprentissage profond (à gauche). La zone en bleu est la zone d'apprentissage.

2.5.1 Réseaux de Neurones à Convolutions (RNC)

Les réseaux de neurones à convolution (en anglais CNN :Convolution Neural Network) fut introduit par [Lecun*et al.*, 1998]. Ce qui fait la particularité de ce type de réseau c'est l'utilisation de l'opération de convolution dans les premières couches intermédiaires du réseau de neurones. À l'origine, cette opération est utilisée comme filtre dans le domaine de l'image ou du son afin de mettre en évidence des motifs ou réduire un type de bruits. Dans les RNC, le modèle apprend lui-même les filtres des différentes convolutions afin de mettre en évidence les motifs des données d'entrée qui sont utilisés dans les couches suivantes Un CNN classique est généralement composé de quatre types de couches :

- Les couches convolutives, qui contiennent plusieurs opérations de convolutions appliquées sur la même entrée,
- Les couches de pooling,
- Les couches ReLu,
- Les couches fully-connected.

2.5.2 Réseau de Neurones Récurrent (RNR)

Alors que les RNC sont principalement utilisés pour faire ressortir des relations spatialement proches (comme des relations entre pixels proches dans une image), les réseaux de neurones récurrents ont été développés afin de garder un contexte temporel pour chaque événement en entrée. Ils ont été particulièrement utilisés pour de l'analyse des séries temporelles, de données audios, ou de textes dans lesquelles le contexte est important afin d'analyser chaque nouvelle entrée. L'idée consiste à garder des informations au cours du temps à l'intérieur des couches de neurones afin de donner un contexte aux données analysées. La sortie du RNR, à un instant t, va dépendre non seulement de l'entrée à l'instant t mais également de l'état du RNR calculé à l'instant $t - 1$.

Dans sa version la plus simple, une couche d'un RNR peut être décrite comme une couche toute connectée l qui prend en entrée la couche précédente $l - 1$ à l'instant t concaténée à la sortie d'elle-même (c'est-à-dire, couche l) à l'instant $t - 1$.

2.5.3 Cellules LSTM

Les cellules Long Short-Term Memory (en français les réseaux récurrents à mémoire courte et long terme) ont été introduites par [Hochreiter and Schmidhuber 1997]. Le but étant de faire face aux problèmes de disparition du gradient lors que l'élément courant et son contexte sont trop éloignés dans le temps. L'idée principale des cellules LSTM est de garder

un état de mémoire $c \in [0,1]^n$ et 3 "portes" utilisées pour faire transiter l'information vers cette mémoire ou la faire sortir. Une première porte, dite porte d'oubli, sert à calculer les éléments de la mémoire c qui doivent être oublies (c'est-à-dire, mise a zéro).

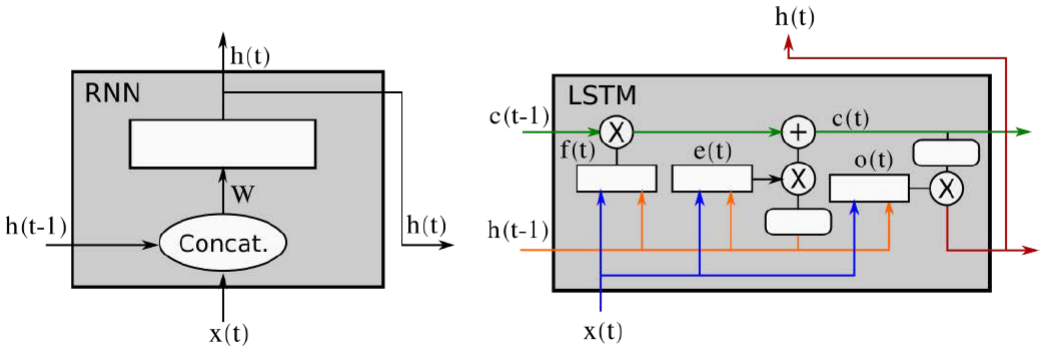

Figure 2-12 Réseau de neurone récurrent et Cellules LSTM

2.6 Les SDI et les RNAs

Pour pallier à de nombreux ces menaces, de nombreux travaux ont tenté de trouver des moyens efficaces pour détecter les intrusions avec des méthodes différentes. Voici ci -dessous quelques travaux :

[Labonne 2020] a proposé des techniques efficaces en déployant des SDIs bases sur des anomalies sur des réseaux réels. Dans laquelle il a effectué un état de l'art complet sur les classificateurs de réseau de neurones pour les bases de données KDD-Cup [Carley et al. 2009] et NSL-KDD [Unb 2019].Le résultat de son travail a déterminé que les deux types de réseaux de neurones les plus performants sur son travail sont les MLP et les cartes de Kohonen. Il a également mis en évidence plusieurs axes d'amélioration en termes d'optimisation des paramètres et de traitement des données.

[Wu and Guo 2019] de leur coté ont présenté une architecture de réseaux de neurone profond, Lunet, qui utilise d'une part les réseaux de neurones a convolution (RNC) pour apprendre les caractéristiques spéciales sur les données de traffic et les LSTMs (Long Short-term Memory) pour les caractéristiques temporels d'autre part. Dans leur travail, pour éviter la perte d'informations due aux différents objectifs d'apprentissage des deux types de réseaux neuronaux, ils ont synchronisé a la fois les deux réseaux pour apprendre les données d'entrée

avec la même granularité. Un des points forts de Lunet, est capable de tirer efficacement partie des avantages des RNC et du LSTM.

Quant à [Wu et al. 2019] et son équipe, ils se sont intéresse aux techniques de l'apprentissage par transfert. En effet ,Ils ont proposé un nouveau Framework de classification des intrusions dans les réseaux utilisant l'apprentissage par transfert a partir du modèle VVG-16, un modèle de RNC proposé par [Simonyan and Zisserman 2014]. Leur Framework extrait les caractéristiques en s'appuyant sur les pré-entrainés formés sur l'ensemble de donnés ImageNet, créer par[Fei-Fei 2009], lors de l'étape initiale, et enfin, appliqué un réseau de neurone profond aux caractéristiques extraites pour la classification des intrusions. Ils ont également mise en œuvre d'autre modèle tels que le VVG19, le MobileNet, le ResNet-50 ou Inception V3 pour évaluer et comparer leur performance.

2.7 Conclusion

Dans ce deuxième chapitre, nous avons introduit le concept de l'apprentissage automatique et de l'apprentissage profond. Nous avons également présenté quelques notions essentielles relatives aux réseaux de neurones, notamment une distinction entre RNs bouclés et les RNs non bouclés. Nous avons ensuite illustré quelques modèles de réseaux de neurones tout en apportant un accent sur les réseaux de neurones profonds.

Nous avons mis l'accès sur l'utilisation des réseaux de neurones pour la conception d'un système de détection d'intrusion avec des recherches notamment là-dessus.

Chapitre 3 : Conception de la l’approche proposée

3.1 Introduction

Dans le domaine de la sécurité informatique, des technologies de références sont nécessaire pour pallier à d'innombrables attaques que font face les systèmes informatiques. Et de ce fait la création d'un système de détection d'intrusion est prime sur le marché pour pallier les certains inconvénients d'avoir des outils sécuritaires comme les anti-virus et autres.

L'objectif de notre travail est de réaliser un SDI qui permet une amélioration des taux des vrais positive tout en réduisant les fausses alarmes ainsi que le temps, facteur très important dans les applications en temps réel.

Nous allons présenter dans ce chapitre notre approche proposée. Elle est développée en trois phases comme le montre la figure 3.1 : sélection des caractéristiques, apprentissage de la carte auto-adaptative de Kohonen puis validation de notre approche ainsi que son évaluation. Cette dernière étape est réalisée en utilisant la base de données NSL-KDD[Unb 2019] qui nécessite une étape de prétraitement afin d'être utilisé, il s'agit d'un codage et d'une normalisation.
Pour réaliser notre système de détection d'intrusion nous avons opté pour un réseau de neurones qui ont la capacité de traiter des problèmes divers et varies, de représenter n'importe quelle fonctions (linéaires ou pas, simple ou complexe) et plus particulièrement la carte de Kohonen qui permet à son tour une réduction de dimensionnalité d'où la rapidité de détection.

3.2 Description de la Base NSL-KDD

L'ensemble de données NSL-KDD[Unb 2019] est la version raffinée de l'ensemble de données KDD cup99[Carley et al. 2009].De nombreux types d'analyse ont été effectués par de nombreux chercheurs sur l'ensemble de données NSL-KDD en utilisant différentes techniques et outils avec un objectif universel pour développer un système de détection d'intrusion efficace.

L'ensemble de données NSL-KDD est une version raffinée de son prédécesseur. Il contient des enregistrements essentiels de l'ensemble de données KDD complet. Il y a une collection de fichiers téléchargeables à disposition pour les chercheurs. Il y a huit différents ensembles de donnés dans le NSL-KDD qui sont répertoriés dans le Tableau 3-1.

Tableau 3-1 Contenu de NSL-KDD

	Nom du Fichier	Description
1	KDDTrain+.ARF	L'ensemble de trains NSL-KDD complet avec des étiquettes binaires au format ARFF
2	KDDTrain+.TXT	L'ensemble de trains NSL-KDD complet, y compris les étiquettes de type d'attaque et le niveau de difficulté au format CSV
3	KDDTrain+_20Percent.ARFF	Un sous-ensemble de 20 % du fichier KDDTrain+.arff
4	KDDTrain+_20Percent.TXT	Un sous-ensemble de 20 % du fichier KDDTrain+.txt
5	KDDTest+.ARFF	L'ensemble de test NSL-KDD complet avec des étiquettes binaires au format ARFF
6	KDDTest+.TXT	L'ensemble de tests NSL-KDD complet, y compris les étiquettes de type d'attaque et le niveau de difficulté au format CSV
7	KDDTest-21.ARFF	Un sous-ensemble du fichier KDDTest+.arff qui n'inclut pas les enregistrements avec un niveau de difficulté de 21 sur 21
8	KDDTest-21.TXT	Un sous-ensemble du fichier KDDTest+.txt qui n'inclut pas les enregistrements avec un niveau de difficulté de 21 sur 21

a. Avantages de NSL-KDD

L'ensemble de données NSL-KDD présente les avantages suivants par rapport à l'ensemble de données KDD d'origine (KDD 99).[Unb 2019]:

- Il n'inclut pas les enregistrements redondants dans la rame, de sorte que les classificateurs ne seront pas biaisés vers des enregistrements plus fréquents.
- Il n'y a pas d'enregistrements en double 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 enregistrements fréquents.
- Le nombre d'enregistrements sélectionnés dans chaque groupe de niveaux de difficulté est inversement proportionnel au pourcentage d'enregistrements dans l'ensemble de données KDD d'origine. En conséquence, les taux de classification des différentes méthodes d'apprentissage automatique varient dans une plage plus large, ce qui rend plus efficace une évaluation précise des différentes techniques d'apprentissage.

- Le nombre d'enregistrements dans le train et les ensembles de test est raisonnable, ce qui rend abordable l'exécution des expériences sur l'ensemble complet sans avoir besoin de sélectionner au hasard une petite partie. Par conséquent, les résultats d'évaluation des différents travaux de recherche seront cohérents et comparables.

b. Caractéristiques de la base de données NSL-KDD : attribut et instance

L'ensemble de données NSL-KDD contient une variété d'attributs, qui peuvent être utiles pour mesurer les attaques. Le jeu de données NSL-KDD un certain nombre d'instance non identiques entre le fichier KDD-test et KDD-train (jeu de donnes de formation).

Tableau 3-2 Nombre d'instance de NSL-KDD

	KDD-test	**KDD-train**
Nombre d'instance	22544	125973

NSL-KDD contient 41 attributs plus un attribut classe, ces attributs sont présents dans le tableau suivant accompagnés de leurs descriptions.

Tableau 3-3 Attributs de la base NSL-KDD

Attribut	**Description**
Duration	La durée de la connexion
Protocol-type	Protocole utilisé dans la connexion
Flag	Statut de la connexion - Normal ou Erreur
Service	Service réseau de destination utilisé
Src_bytes	Nombre d'octets de données transfères de la source à la destination en une seule connexion
Dst_bytes	Nombre d'octets de données transférés de la destination à la source dans une connexion unique
Land	Si les adresses IP source et de destination et les numéros de port sont _égaux, cette variable prend la valeur 1 sinon 0
Wrong_fragment	Nombre total de fragments incorrects dans cette connexion
Urgent	Nombre de paquets urgents dans cette connexion. Les paquets urgents sont des paquets avec le bit urgent activé
Hot	Nombre d'indicateurs < hot > dans le contenu tels que : entrer dans un répertoire système, créer des programmes et exécuter des programmes
Num_failed_logins	Nombre de tentatives de connexion échouées
logged_in	Etat de connexion : 1 si la connexion a réussi ;

	0 sinon
Num_compromised	Nombre de conditions compromises
Root_shell	1 si la coque racinaire (root shell) est obtenue ; 0 sinon
Su_attempted	1 si la commande <su root > a été tentée ou utilisée ; 0 sinon
Num_root	Nombre d'accès "root" ou nombre d'opérations effectuées en tant que root dans la connexion
Num_file_creations	Nombre d'opérations de création de fichiers dans la connexion
Num_shells	Nombre d'invites du shell
Num_access_files	Nombre d'opérations sur les fichiers de contrôle d'accès
Num_outbound_cmds	Nombre de commandes sortantes dans une session ftp
Is_host_login	1 si la connexion appartient à la liste "hot", c'est-_a-dire root ou admin ; sinon 0
Is_guest_login	1 si la connexion est une connexion < invité > ; 0 sinon
Count	Nombre de connexions au même hôte de destination que la connexion actuelle au cours des deux dernières secondes
Srv_count	Nombre de connexions au même service (numéro de port) que la connexion actuelle au cours des deux dernières secondes
Serror_rate	Pourcentage de connexions qui ont activé l'indicateur <flag > (4) s0, s1, s2 ou s3, parmi les connexions regroupées dans count (23)
Srv_serror_rate	Pourcentage de connexions qui ont activé l'indicateur < flag > (4) s0, s1, s2 ou s3, parmi les connexions regroupées dans srv_count(24)
Rerror_rate	Pourcentage de connexions qui ont activé l'indicateur < flag > (4) REJ, parmi les connexions regroupées dans count (23)
srv_rerror_rate	Pourcentage de connexions qui ont activé l'indicateur < flag > (4) REJ, parmi les connexions regroupées dans srv_count (24)
Same_srv_rate	Le pourcentage de connexions qui étaient au même service, parmi les connexions regroupées en count (23)
Diff_srv_rate	Le pourcentage de connexions qui étaient à différents services, parmi les connexions regroupées en count (23)
srv_diff_host_rate	Le pourcentage de connexions qui étaient vers différentes machines de destination parmi l'agrégat des connexions dans srv_count (24)
dst_host_count	Nombre de connexions ayant la même adresse IP d'hôte de destination

dst_host_srv_count	Nombre de connexions ayant le même numéro de port
dst_host_same_srv_rate	Le pourcentage de connexions qui étaient à différents services, parmi les connexions regroupées dans dst_host_count (32)
dst_host_diff_srv_rate	Le pourcentage de connexions qui étaient à différents services, parmi les connexions regroupées dans dst host count (32)
dst_host_same_src_port_rate	Pourcentage de connexions qui se trouvaient sur le même port source, parmi les connexions regroupées dans dst_host_srv_count (33)
dst_host_srv_diff_host_rate	Pourcentage de connexions qui étaientvers différentes machines de destination, parmi les connexions regroupées dans dst_host_srv_count (33)
dst_host_serror_rate	Pourcentage de connexions qui ont activé l'indicateur (4) s0, s1, s2 ou s3, parmi les connexions regroupées dans dst_host_count (32)
dst_host_srv_serror_rate	Pourcentage de connexions qui ont activé l'indicateur (4) s0, s1, s2 ou s3, parmi les connexions regroupées dans dst_host_srv_count (33)
dst_host_rerror_rate	Pourcentage de connexions qui ont activé l'indicateur (4) REJ, parmi les connexions agrégées dans dst host count (32)
dst_host_srv_rerror_rate	Pourcentage de connexions qui ont activé l'indicateur (4) REJ, parmi les connexions agrégées dans dst_host_srv_count (33)
Classe	Classification de l'entrée de traffic

Dans l'ensemble de données, il existe 4 classes d'attaques différentes : déni de service (DoS), sonde, utilisateur à racine (U2R) et distant à local (R2L). Une brève description de chaque attaque peut être vue dans la section1.2.6

3.2.1 Etape de prétraitement

Le prétraitement des données est l'une des étapes critiques du processus d'exploration de données qui effectue la préparation et la transformation de l'ensemble de données d'origine.[Gnanaprasanambikai and Munusamy 2018]

3.2.1.1 Codage

Les types d'entités de la base NSL-KDD peuvent être divisés en 4 types :

- 4 catégoriques (textuelles) : 2, 3, 4, 43

- 6 binaires : 7, 12, 14, 20, 21, 22
- 23 discrets (numériques) : 8, 9, 15, 23–42
- 10 continus (numériques) : 1, 5, 6, 10, 11, 13, 16, 17, 18, 19

Les attributs 2(protocol_types) ,3(service),4(flag)étant textuelles et sachant que nos modèles n'acceptent que des valeurs numériques, nous avons converti les 3 attributs mentionnes vers des données numériques.

Tenant en compte que la sortie de notre modèle est numérique, l'attributs class doit être aussi code.

- **Attributs protocol_types**

Tableau 3-4AttributsProtocol_types

Normale	**Coder**
ICMP	1
TCP	2
UDP	3

- **Attributs flag**

Tableau 3-5 Attributs flag

Normal	**Coder**
OTH	1
REJ	2
RSTO	3
RSTOS0	4
STSP	5
SH	6
S0	7
S1	8
S2	9
S3	10
SF	11

- **Attributs service**

Tableau 3-6 Attributs services

Normal	**Coder**	**Normal**	**Coder**	**Normal**	**Coder**
aol	1	imap4	26	rje	51
auth	2	IRC	27	shell	52
bgp	3	iso_tsap	28	smtp	53
courier	4	klogin	29	sql_net	54
csnet_ns	5	kshell	30	ssh	55
ctf	6	ldap	31	sunrpc	56
daytime	7	link	32	supdup	57
discard	8	login	33	systat	58

domain	9	mtp	34	telnet	59
domain_u	10	name	35	tftp_u	60
echo	11	netbios_dgm	36	tim_i	61
eco_i	12	netbios_ns	37	time	62
ecr_i	13	netbios_ssn	38	urh_i	63
efs	14	netstat	39	urp_i	64
exec	15	nnsp	40	uucp	65
finger	16	nntp	41	uucp_path	66
ftp	17	ntp_u	42	vmnet	67
ftp_data	18	other	43	whois	68
gopher	19	pm_dump	44	X11	69
harvest	20	pop_2	45	Z39_50	70
hostnames	21	pop_3	46		
http	22	printer	47		
http_2784	23	private	48		
http_443	24	red_i	49		
http_8001	25	remote_job	50		

3.2.1.2 Normalisation

Certaines valeurs dans l'ensemble de donnés sont très variés et constituent un grand intervalle. La normalisation est un processus de transformation dans lequel un attribut numérique est mis à l'échelle dans une plage plus petite telle que 0,0 à 1,0. Dans notre approche, les méthodes/techniques appliquées à la normalisation des données sont :

$$\chi_{nouv} = \frac{\mathrm{X}_{anc} - \mathrm{X}_{min}}{\mathrm{X}_{max} - \mathrm{X}_{min}} \qquad (3\text{-}1)$$

Avec :

- χ_{nouv} : La nouvelle valeur obtenue.
- X_{anc} : la valeur a normalisé
- X_{min} : La valeur minimale du champ
- X_{max} : la valeur maximale du champ

3.3 Approche Proposée

Enfin de réaliser notre application, nous nous sommes bases sur la carte auto-adaptatif de Kohonen déjà explique dans le chapitre précèdent. L'apprentissage non supervise a l'aide de la carte de Kohonen fournit un moyen simple et efficace de classer l'ensemble de données.

Pour traiter les données en temps réel pour la classification, nous pensons que la carte de Kohonen est mieux adaptée en raison de son taux de conversion élevé et rapide. Aussi elle préserver les mappages topologiques entre les représentations, une caractéristique qui est souhaitée lors de la classification. Elle permet également de conserver la topologie de l'ensemble d'entrée et elles relèvent des corrélations qui ne sont pas faciles a identifier.

Nous avons choisi une structure rectangulaire qui donne de meilleur résultat. En effet dans notre structure, l'information est associée à chaque hub, et les hubs structurent une section transversale multidimensionnelle sur un espace (généralement un réseau bidimensionnel). Les nœuds sont des entrées compétitives et sont pris en charge chacun à tour de rôle, et le nœud qui a la réponse la plus fondée à l'information est proclamé le « nœud gagnant » et a la possibilité d'attribuer son poids à l'entrée et de le renvoyer comme rendement.

La Carte de Kohonen prend un jeu de données multidimensionnel avoir des tas et des tas de lignes et de colonnes qui sont de (12 x 12) dimensions de l'ensemble de données et de réduire la dimensionnalité de l'ensemble de données. Le but de SOM est de réduire le nombre de colonnes en sortie de deux dimensions.

3.3.1 Algorithme de notre approche proposée

Les étapes de notre approche sont définies comme suit :

- Étape 1 : prétraitement puis sélection des caractéristiques
- Étape 2 : réalisation d'un RN de Kohonen en commençant par la création d'une grille composée de nœuds (12x12), chacun ayant un vecteur de poids de l'élément de nos caractéristiques.
- Étape 3 : initialisation aléatoire des valeurs du vecteur de poids dans un intervalle ouvert [0 ,1].
- Étape 4 : Sélection d'un point d'observation aléatoire dans l'ensemble de données.
- Étape 5 : calcule de la distance euclidienne de ce point aux différents neurones du réseau.
- Étape 6 : sélection les neurones qui ont la distance minimale au point. Ce neurone est appelé nœud gagnant.
- Étape 7 : mettre à jour les poids du nœud gagnant afin de le rapprocher du point.

- Étape 8 : à l'aide d'une fonction Gaussienne de voisinage de la moyenne du nœud gagnant, mettre également à jour le poids des voisins du nœud gagnant pour se rapprocher du point. Le rayon de voisinage est le sigma dans la fonction Gaussienne
- Étape 9 : Répétez les étapes 3 à 8 et mettez à jour le poids après chaque observation ou un lot d'observations (Batch Learning), jusqu'à ce que le réseau converge vers un point où le voisinage cesse de diminuer.

Pour déterminer le nœud gagnant, un processus itératif est effectué par chaque nœud du réseau et la distance euclidienne entre le vecteur de poids de chaque nœud et l'entrée actuelle vecteur est calculé. Le nœud avec le vecteur de poids le plus proche du vecteur d'entrée est étiqueté comme nœud gagnant. La distance euclidienne est calculée en ((3-2).

$$Distance\ Euclidienne = \sqrt{\sum_{i=0}^{n} (V_i - W_i)^2} \tag{3-2}$$

Avec :

V : le vecteur d'entrée actuel

W : le vecteur de poids de chaque nœud du réseau.

Après avoir déterminé le nœud gagnant, l'étape suivante consiste à déterminer lesquels des autres nœuds du réseau, qui ne sont pas nœud gagnant, se trouvent dans le voisinage. Une fois ces nœuds identifiés, on procédera à la modification de leurs vecteurs de poids. Par conséquent, le rayon du voisinage est calculé, en utilisant le théorème de Pythagore pour déterminer si chaque nœud est à distance radiale ou non. La zone de voisinage se rétrécit avec le temps, car elle est directement proportionnelle au rayon du voisinage, qui est calculé à partir de la fonction exponentielle décroissante, définie en (3-3)

$$\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\lambda}\right), t = 1,2,3 \ldots \tag{3-3}$$

Où $\sigma_0$désigne la largeur de la trame en temps « t0 » et λ désigne un temps constant « t » qui est le pas de temps courant. Après plusieurs itérations, le voisinage sera ajusté à la taille d'un seul nœud, le nœud gagnant, qui déterminera la valeur du rayon, qui est nécessaire pour identifier si un nœud n'est pas dans le voisinage. Si un nœud est situé dans le voisinage, le vecteur de poids doit être ajusté. Le nœud gagnant et chacun des nœuds situés dans le voisinage ont un vecteur de poids ajusté en (3-4)

$$W(t+1) = W(t) + \theta(t)L(t)(V(t) - W(t)) \tag{3-4}$$

Où "t" est le temps et "L" est une petite variable appelée taux d'apprentissage, qui diminue avec le temps, l'équation (3-4) indique que le poids au temps "t +1" est défini sur les nœuds voisins à partir du poids instantané actuel " W(t)", plus une fraction "L(t)", la différence entre le poids actuel du nœud "W(t)" et l'entrée du vecteur de poids à l'instant présent "V(t)".

Le taux d'apprentissage comme le rayon du voisinage sont calculés en utilisant une fonction décroissance exponentielle pour déterminer sa valeur dans la variation temporelle. Le taux d'apprentissage est représenté à l'équation (3-5)

$$L(t) = L_0 \exp\left(-\frac{t}{\lambda}\right), t = 1,2,3\ldots \tag{3-5}$$

Dans l'équation (3-4), $\theta(t)$représente la fonction de voisinage, qui est la distance d'un nœud au nœud gagnant sur leur apprentissage dans l'instant du temps actuel. Elle est calculée comme suite :

$$\theta(t) = \exp\left(-\frac{dist^2}{2\sigma^2(t)}\right), t = 1,2,3\ldots \tag{3-6}$$

Où « dist » est la distance entre un nœud et le nœud gagnant et « σ » est le rayon du voisinage, énoncé à l'équation (3-3). La fonction diminue également avec le temps.

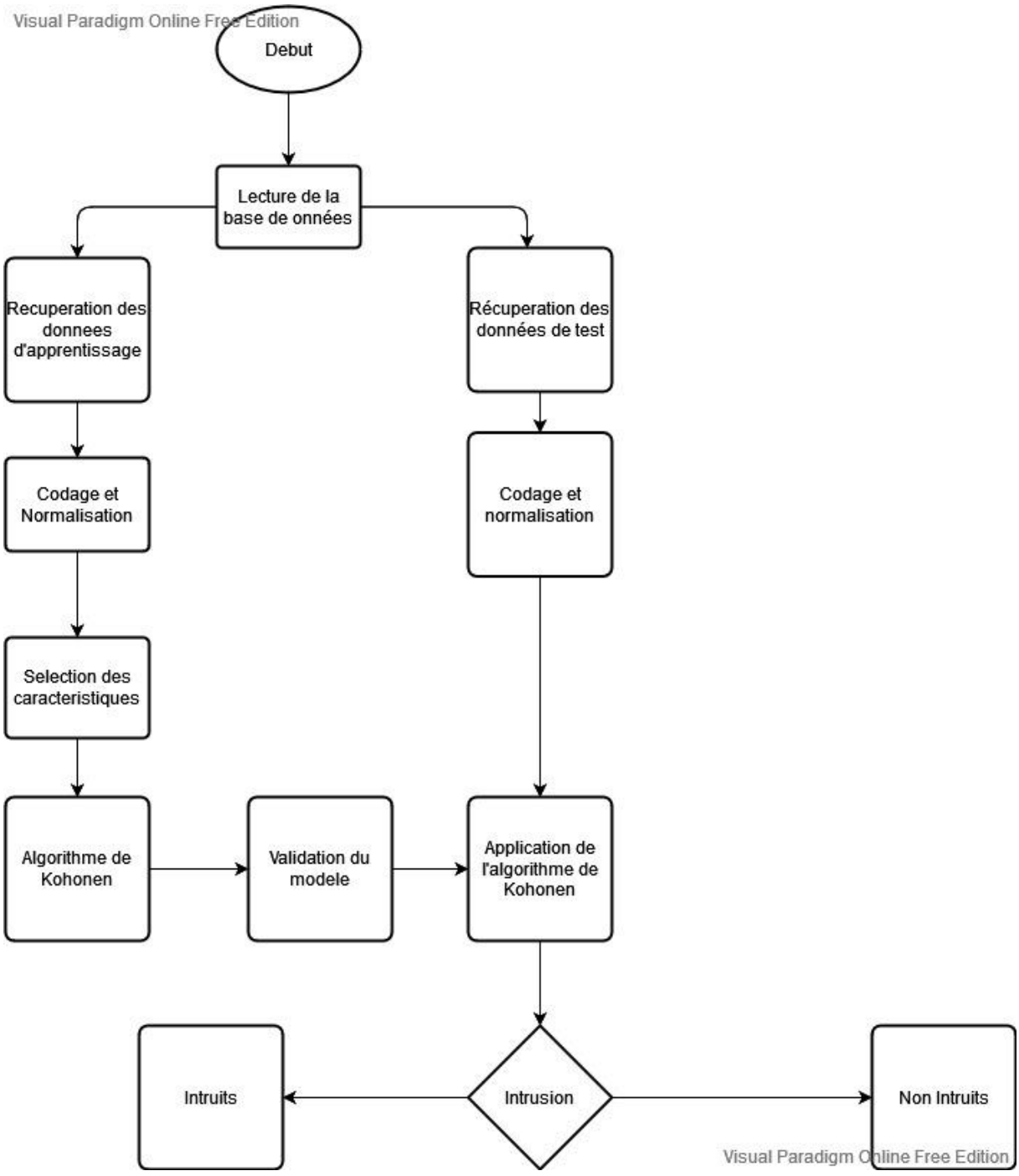

Figure 3-1 Organigramme de notre approche proposée

3.3.2 Reduction de dimensionnalité : Sélection des caractéristiques

La sélection des fonctionnalités est importante pour améliorer l'efficacité des algorithmes d'exploration de données. C'est le processus de sélection d'un sous-ensemble de

caractéristiques d'origine selon certains critères, et c'est une technique importante et fréquemment utilisée dans l'exploration de données pour la réduction des dimensions. Elle réduit le nombre de fonctionnalité non pertinentes, redondantes ou bruyantes et entraine des effets palpables sur les applications : accélération d'un algorithme, amélioration de la précision de l'apprentissage et conduit a une meilleure compréhensibilité du modèle[Su et al. 1999]

Pour réduire le temps de détection et améliorer les taux des vrais positifs et négatifs, nous nous sommes basés sur le travail déjà établi par [Belgrana et al. 2020].Nous avons alors réduit le nombre d'attribut des paquets (trames) où nous avons retenu 23 caractéristiques et cela à partir de 42 attributs de base, il est question d'opter pour les attributs ayant un taux de variance faibles. Il s'agit d'une méthode non basée sur le modèle, ce choix est dû à son adaptabilité pour tous type de méthode.

Et lors de la sélection des caractéristiques de la base de NSL-KDD, les auteurs dans [Belgrana et al. 2020] ont opté pour la méthode des taux de variance ou la variance est calculée en utilisant la formule suivante de [Mukhopadhyay et al. 2008] :

$$Var(x) = \sigma = 1/n \sum_{i=1}^{n} (x_i - \mu)^2 \qquad 3\text{-}7$$

La table présentée ci-dessous correspond aux attributs de la base NSL-KDD ainsi que leurs variances calculées via l'équation 3.7.

Tableau 3-7 Taux de variance des valeurs des attributs de la base NSL-KDD

Variable (attribut du vecteur NSL-KDD)	**Variance**
logging_in	0,239128934
dst_host_same_svr_rate	0,201553931
srv_serror_rate	0,199827528
serror_rate	0,199321042
dst_host_srv_serror_rate	0,198619391
dst_host_serror_rate	0,197831281
same_srv_rate	0,193266727
dst_host_srv_count	0,188465967
dst_host_count	0,15135401
srv_rerror_rate	0,104746697
rerror_rate	0,102678108
dst_host_srv_rerror_rate	0,102053492
dst_host_same_src_port_rate	0,095478469

dst_host_rerror_rate	0,093976729
protocol_type	0,079863859
Flag	0,07232626
srv_diff_host_rate	0,067511352
Service	0,059988616
Count	0,050214717
dst_host_diff_srv_rate	0,035691163
diff_srv_rate	0,032513027
srv_count	0,020204899
dst_host_srv_diff_host_rate	0,01267051
is_guest_login	0,009333868
wrong_fragment	0,007141883
Duration	0,00368446
root_shell	0,001339758
Hot	7,80E-04
su_attempted	5,10E-04
Land	1,98E-04
num_file_creations	1,27E-04
num_shells	1,23E-04
num_access_files	1,22E-04
num_failed_logins	8,19E-05
Urgent	2,29E-05
src_bytes	1,81E-05
num_root	1,07E-05
num_compromised	1,02E-05
dst_bytes	9,42E-06
is_host_login	7,94E-06
num_outbound_cmds	0

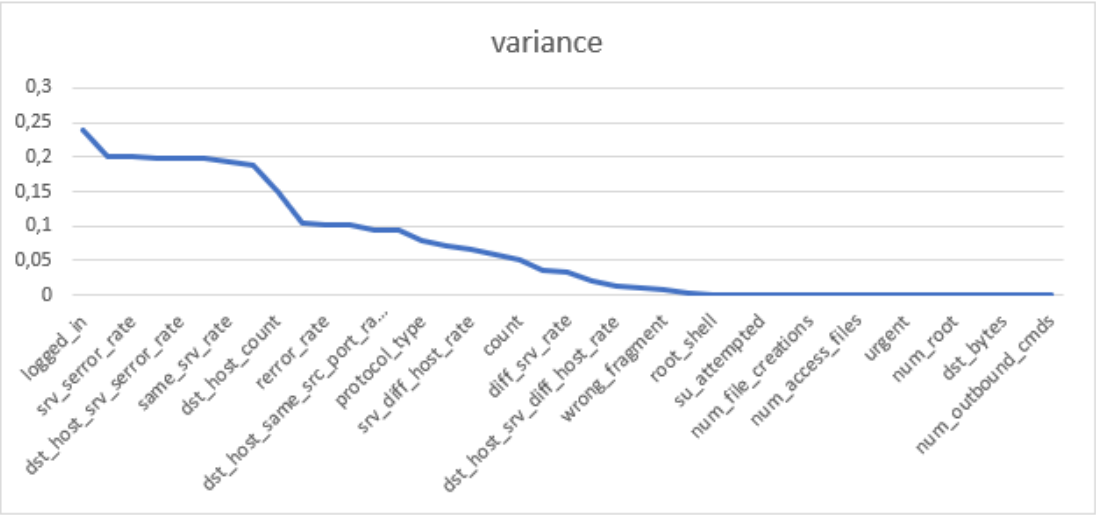

Figure 3-2 Courbe de variance des attributs de la base NSL-KDD[Abdallah MOHAMMED et al. 2020]

3.3.3 Notre carte auto-adaptative de Kohonen

a. Architecture de la carte topographique

La carte topologique est une grille compose d'unités de neurone ou chaque neurone est lié aux autres neurones et connecte à toutes les unités d'entrée dont le nombre correspond a la dimension des données d'entrée. Les vecteurs d'entrée et les vecteurs de poids de tous les neurones ont les mêmes dimensions.

Dans notre cas, nous avons utilisé une grille de dimension 12x12 et comme vecteur d'entrée qui est le résultat de la sélection de caractéristique obtenu sur la base de données NSL-KDD.

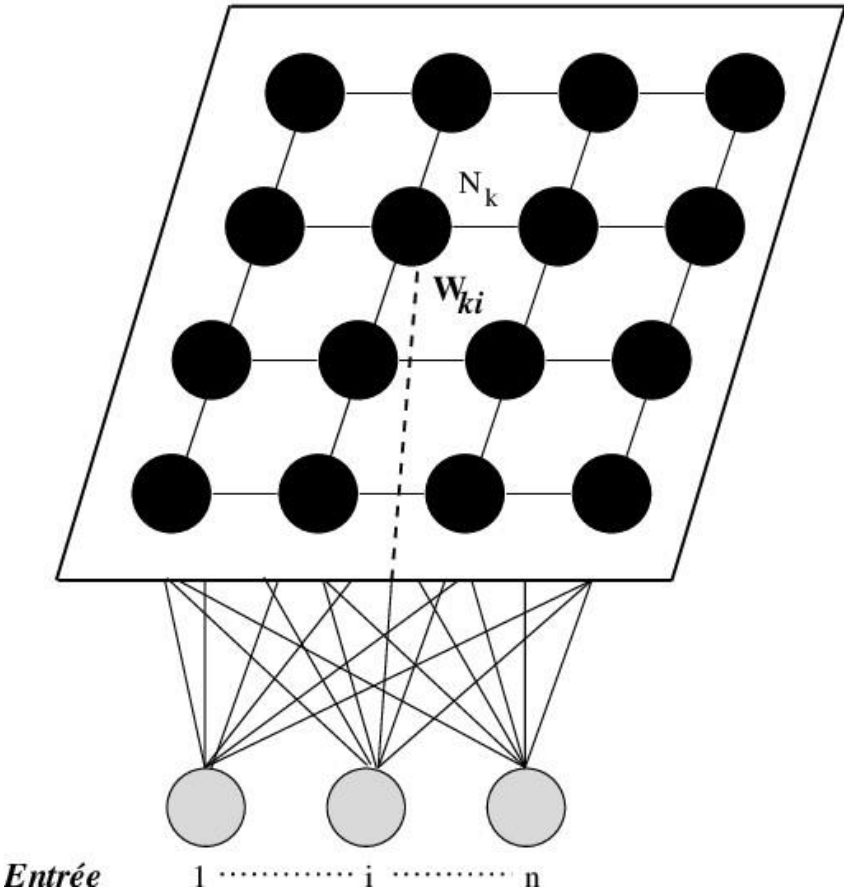

Figure 3-3 Architecture de la carte de Kohonen

b. Phase d'apprentissage

L'algorithme correspondant à la phase d'apprentissage est illustré dans la section3.3.1.

Cette étape correspond à l'application de l'algorithme sur l'ensemble des données d'apprentissage de NSL-KDD. Une fois cette phase achevée, nous affichons la carte de Kohonen obtenue et qui sera utilisée dans la phase de test (voir Figure 3-4).

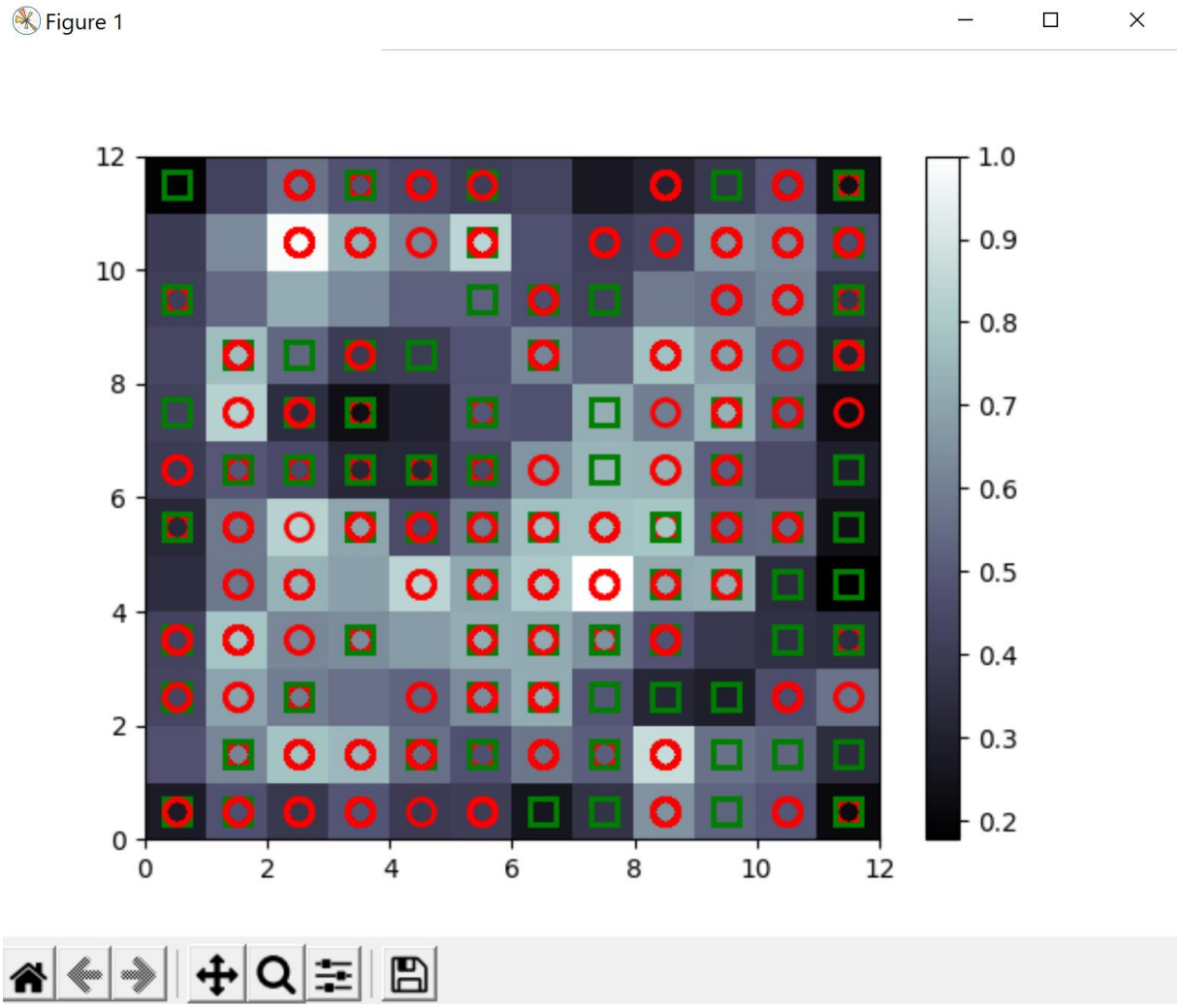

Figure 3-4 Résultat de l'apprentissage

Dans la carte d'auto organisation obtenu et illustré par la figure 3-4, le cercle rouge signifie que le paquet reçu est une intrusion et la carre vert signifie le contraire. Et si nous examinons notre valeur aberrante, la zone de couleur blanche est une intrusion potentielle élevée que nous détectons. Dans les autres zones de couleur représente le degré de négligence.

c. Phase de test

Elle correspond à l'usage de la carte déjà établie dans la phase précédente. Cette réutilisation s'effectuera à l'ensemble des données de test pour reproduire le schéma et enfin distinguer les intrusions potentielles (voir section3.4).

3.4 Résultats obtenus

3.4.1 Erreur de quantification

C’est une métrique statistique qui représente la différence entre les données et les résultats obtenus en laissant un réseau de neurones autoorganisé apprendre les données. Dans des travaux antérieurs, l'erreur de quantification de la sortie SOM a été exploitée comme une métrique qualitative de réseau de neurones, [Kohonen et al. 2009], ou plus récemment en tant que métrique pour la mise à l'échelle dynamique de précision dans l'apprentissage des réseaux de neurones [Taras and Stuart 2019].

Tableau 3-8 erreur de quantification

Erreur de quantification
0.32567554200162974

3.4.2 Rapport de classification

Le rapport de classification concerne les mesures clés d'un problème de classification. Nous avons la visualisation du rapport de classification en affichant les valeurs de la précision, du rappel, du score f1 et du support décrit dans le Tableau 3-9.

Il existe quatre façons de vérifier si les prédictions sont bonnes ou fausses [Kohli 2019]:

- **VN / Vrai Négatif** : le modèle prédit négatif, et la valeur réelle est négative
- **VP / Vrai Positif** : le modèle prédit positif, et la valeur réelle est positive
- **FN / Faux Négatif** : le modèle prédit négatif, mais la valeur réelle est positive (Type d’erreur 1)
- **FP / Faux Positif** : le modèle prédit positif, mais la valeur réelle est négative (Type d’erreur 2)

a. Précision

La précision est la capacité d'un classificateur à ne pas étiqueter une instance comme positive qui est en fait négative. Pour chaque classe, elle est défini comme le rapport des vrais positifs à la somme des vrai positifs et des faux positifs [Kohli 2019].

Précision : Précision des prédictions positives

$$\boldsymbol{Precision = VP/(VP + FP)} \quad 3\text{-}8$$

b. Rappel

Le Rappel est la capacité d'un classificateur de trouver tous les cas positifs. Pour chaque classe, il est défini comme le rapport des vrais positifs à la somme des vrais positifs et des faux négatifs[Kohli 2019].

Rappel : Fraction des cas positifs qui ont été correctement identifient

$$\boldsymbol{Rappel = VP/(VP + FN)} \quad 3\text{-}9$$

c. Score F1

Le score F1 est une moyenne harmonique pondérée de la précision et du rappel telle que le meilleur score est de 1,0 et le pire est de 0,0. Les scores F1 sont inférieurs aux mesures de précision car ils intègrent la précision et le rappel dans leur calcul. En règle générale, la moyenne pondérée de F1 doit être utilisée pour comparer les modèles de classificateur, et non la précision globale.[Kohli 2019]

$$\boldsymbol{Score\ F1 = 2 * (Rappel * Precision)/(Rappel + Precision)} \quad 3\text{-}10$$

d. Support

Le support est le nombre d'occurrences réelles de la classe dans l'ensemble de données spécifié. Un support déséquilibré dans les données d'apprentissage peut indiquer des faiblesses structurelles dans les scores rapportés du classificateur et pourrait indiquer la nécessité d'un échantillonnage stratifié ou d'un rééquilibrage. Le support ne change pas entre les modèles mais diagnostique à la place le processus d'évaluation.[Giulio Laurenti 2020]

Tableau 3-9 Rapport de Classification

	Précision	**Rappel**	**Score F1**	**Support**
Anormale	0.89	0.75	0.81	3208
Normale	0.73	0.87	0.79	2428
Moyenne macro	0.81	0.81	0.80	5636
Moyenne pondéré	0.81	0.80	0.80	5636

Pour la classe anormale, la précision est supérieure au rappel, ce qui signifie qu'il y a plus de Faux Négatif que de Faux Positif. Pour cette même la classe, le rappel est supérieur à la précision, ce qui signifie qu'il a moins de Faux Négatif que de Faux Positif

Outre les métriques d'évaluation, le rapport de classification comprend des informations supplémentaires tels que [Giulio Laurenti 2020]:

- **Moyenne macro**

Elle représente la moyenne arithmétique d'une métrique entre les deux classes.

Moyenne macro(précision) = (PA + PN) / 2 = (0.88+0.73) / 2 = 0.89 = 89%

Avec :

PA : Précision Anormale

PN : Précision Normale

- **Moyenne pondérée**

Elle est calculée en divisant la somme (métrique d'intérêt x poids)par la somme des poids

Moyenne pondéré (précision) = (PA x SA + PN x SN) / (SA+SN) = (0.89*3208 + 0.73*2428) / (3208+2428) = 0.81= 81%

Avec :

PA : Précision Anormale

PN : Précision Normale

SA : Support Anormale

SN : Support Normale

Tableau 3-10 Rapport de Classification sans la sélection des caractéristiques

	Précision	**Rappel**	**Score F1**	**Support**
Anormale	0.85	0.73	0.80	3208
Normale	0.72	0.85	0.78	2428
Moyenne macro	0.79	0.78	0.79	5636
Moyenne pondéré	0.78	0.79	0.79	5636

3.5 Présentation de notre application

3.5.1 Présentation des outils

3.5.1.1 Langage utilisé

- **Python**

Python est l'un des langages de programmation plus largement utilisé, interprété, orienté objet, et de programmation de haut niveau de la langue avec la sémantique dynamique, utilisé pour la programmation à usage général. Il a été créé par Guido van Rossum, et tout d'abord publié le 20 février 1991.[Python Institute 2018].

Le langage Python doit sa popularité à plusieurs avantages qui profitent aussi bien aux débutants qu'aux experts. Tout d'abord, il est facile à apprendre et à utiliser. Ses caractéristiques sont peu nombreuses, ce qui permet de créer des programmes rapidement et avec peu d'efforts. De plus, sa syntaxe est conçue pour être lisible et directe.

Un autre avantage du Python est sa popularité. Ce langage fonctionne sur tous les principaux systèmes d'exploitation et plateformes informatiques. De plus, même s'il ne s'agit clairement pas du langage le plus rapide, il compense sa lenteur par sa versatilité.

Python est le langage de programmation le plus utilisé dans le domaine du Machine Learning, du Big Data et de la Data Science.[LeBigData 2021]

3.5.1.2 Logiciels

- **Visual Studio Code**

C’est un éditeur de code développé par Microsoft en 2015. Contrairement à ce à quoi Microsoft a eu l’habitude de nous habituer durant des années, il est l’un de ces premiers produits open source et gratuit, et surtout disponible sur les systèmes d’exploitation Windows, Linux et Mac.[Webnet 2019]

Visual Studio Code combine la simplicité d'un éditeur de code source avec de puissants outils de développement, tels que la complétion et le débogage de code IntelliSense.[VisualStudioCode 2020]

3.5.1.3 Bibliothèques logicielles

- **Tkinter**

C’est le graphique libre d'origine pour le langage Python, permettant la création d'interfaces graphiques.

Elle nous a permis à faire notre interface.

- **Pandas**

C’est une bibliothèque écrite pour le langage de programmation Python permettant la manipulation et l'analyse des données. Elle propose en particulier des structures de données et des opérations de manipulation de tableaux numériques et de séries temporelles.

Pandas est un logiciel libre sous licence BSD, ce qui permet sa réutilisation avec peu de restrictions

- **Numpy**

Est la librairie fondamentale du calcul scientifique avec Python, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux (array).

Les calculs avec Numpy sont particulièrement optimisés car les tableaux sont homogènes (ils ne contiennent que des valeurs d’un même type) et de taille fixée à la création

- **Csv-Python**

Le format CSV (Comma Separated Values) est le format d'importation et d’exportation utilisé par les feuilles de calcul et les bases de données. Csv est une bibliothèque qui implémente des classes pour lire et écrire des données tabulaires au format CSV.

Elle permet d'écrire les données dans le format Excel ou lire les données de fichier généré par Excel.

- **Matplotlib**

C'est une bibliothèque de traçage Python à deux dimensions qui produit des images de qualité. Elle peut être utilisée dans les scripts Python, les serveurs d'applications Web, etc. Matplotlib permet de générer des histogrammes, des graphiques à barres, des diagrammes d'erreur, des diagrammes de dispersion, etc., avec seulement quelques lignes de code.

Nous avons utilisé cette bibliothèque pour tracer des graphes afin de présenter le résultat de notre approche.

3.5.1.4 Configuration du matériel utilisé

La configuration du matériel utilisé pour la réalisation de notre application est la suivante :

- Processeur Intel(R) Core (TM) i7-4750HQ CPU @ 2.00GHz, 1995 MHz, 4 cœur(s), 8 processeur(s) logique(s)
- Carte graphique NVIDIA GTX 960m 2G0 DDR5.
- RAM d'une taille de 8 GO.
- Disque dur (SSD) d'une taille de 118 GO.
  Système d'exploitation Windows 10 Pro 64 bits.

3.5.2 Présentation de l’application

a. Chargement de la base

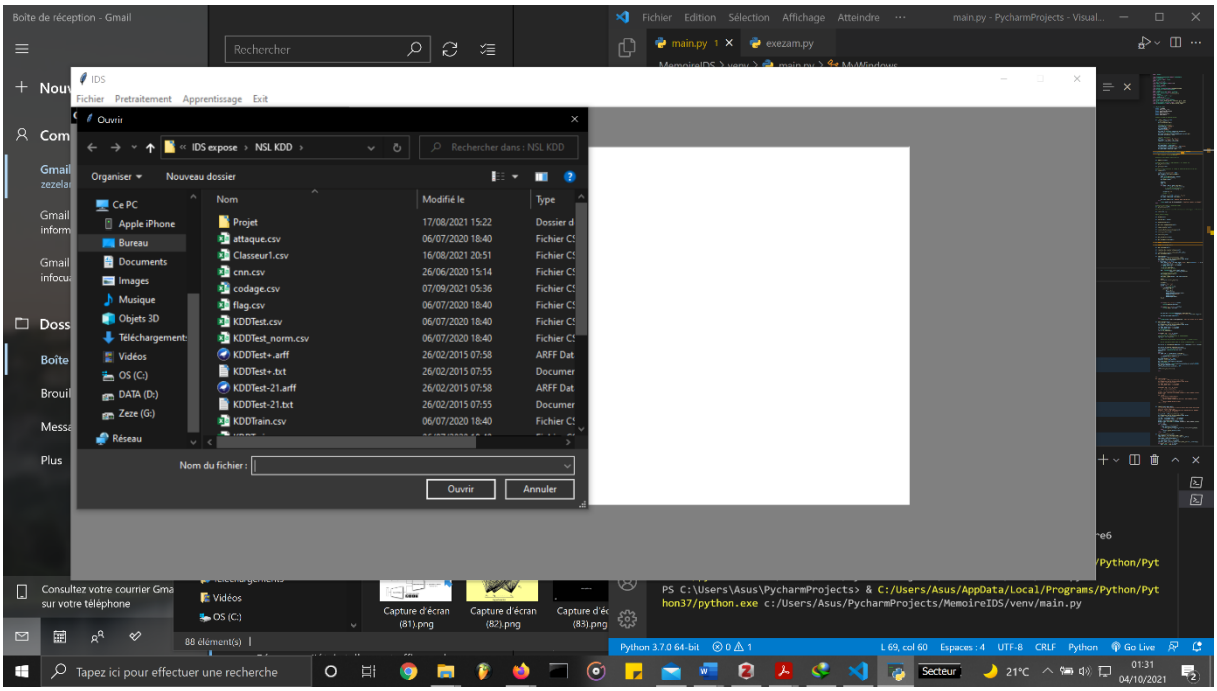

Figure 3-5 Interface pour le chargement de la base de données NSL-KDD

b. Lecture de la base

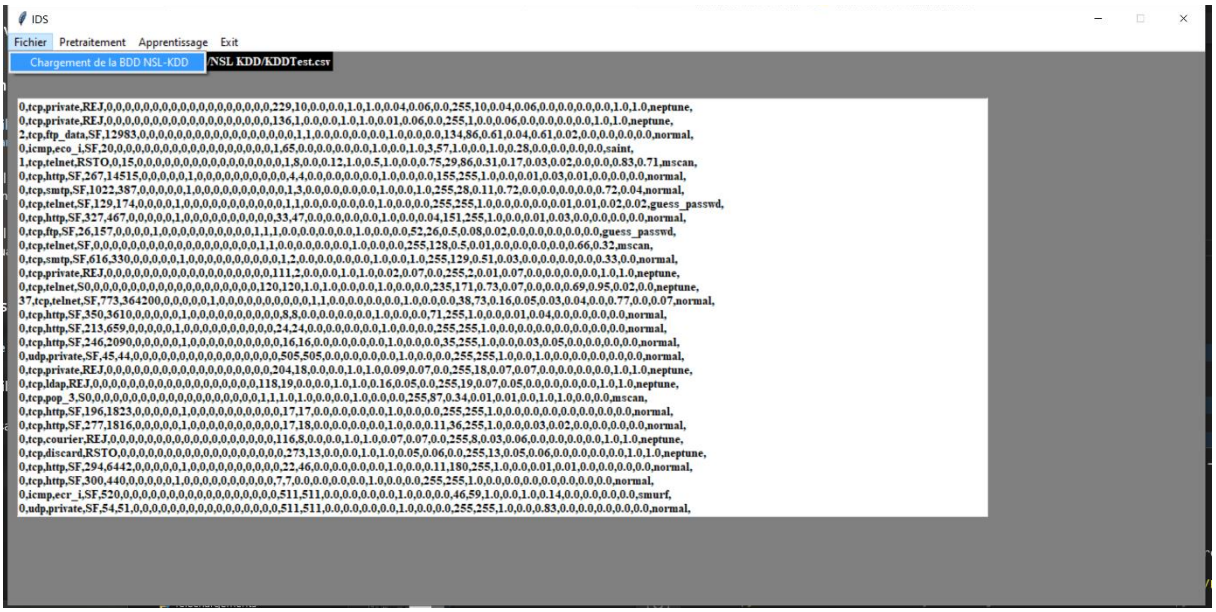

Figure 3-6Interface de lecture

c. Codage de la base

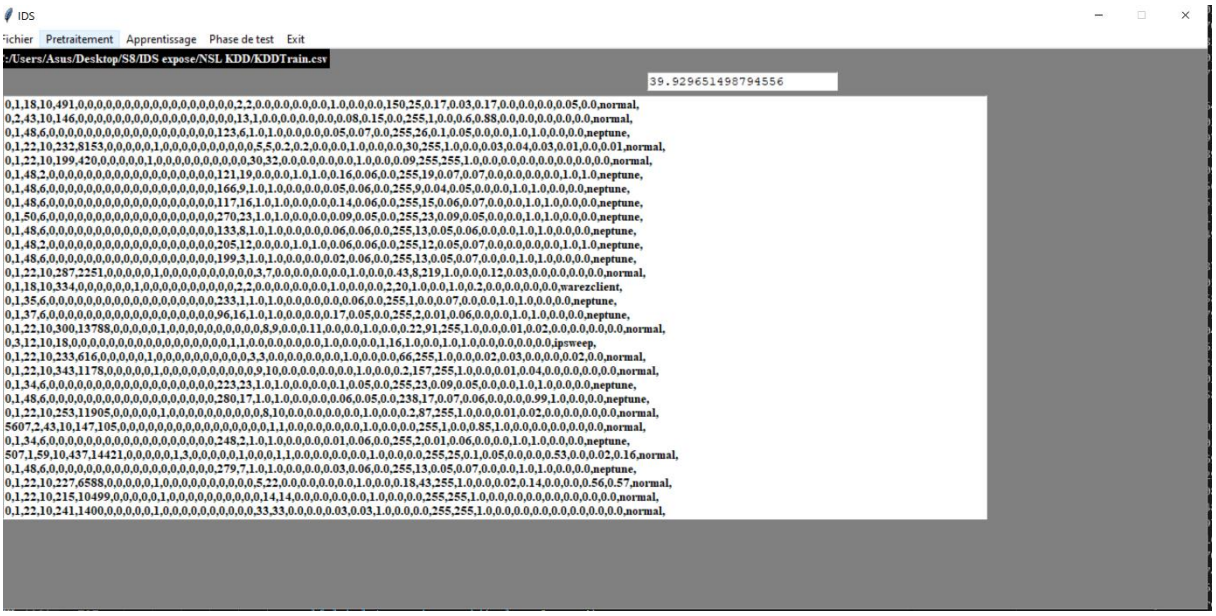

Figure 3-7Interface du codage de la base

d. Normalisation de la base

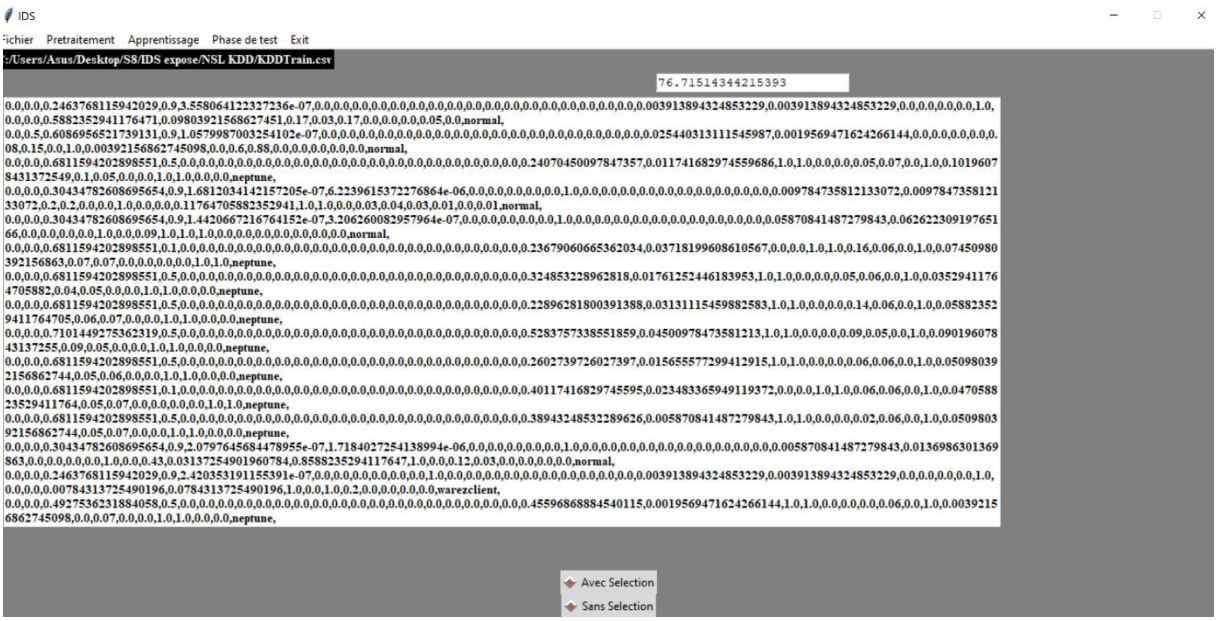

Figure 3-8Interface de la normalisation de la base

e. Sélection des caractéristiques

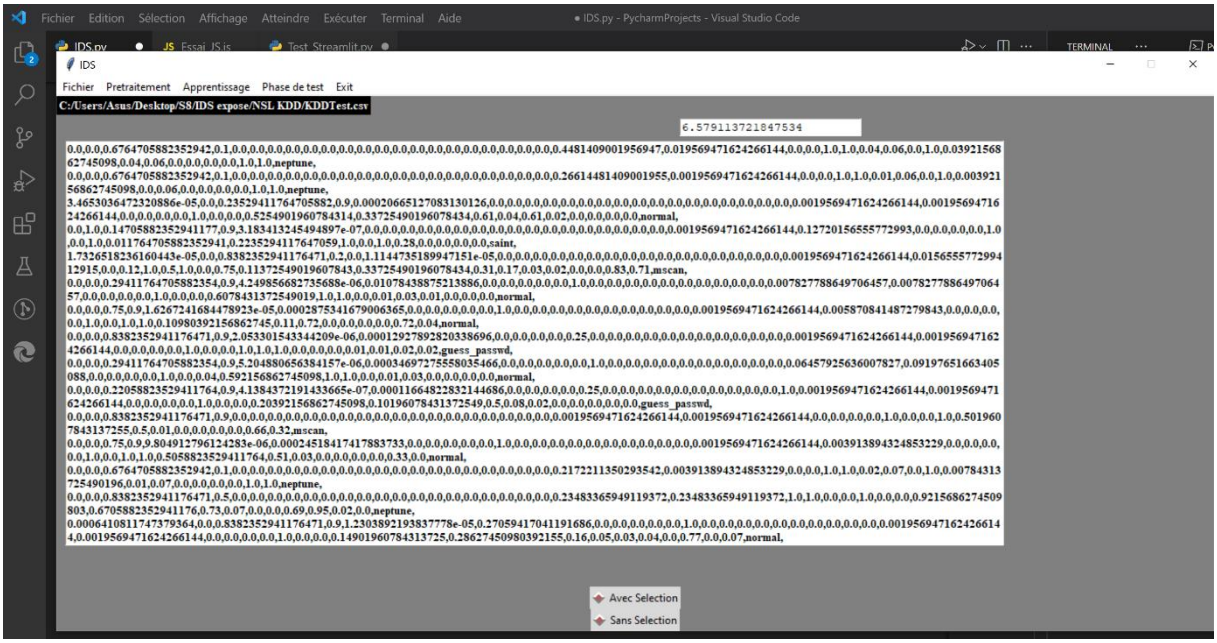

Figure 3-9 Interface de la réduction des caractéristiques

f. Apprentissage la carte de Kohonen

Figure 3-10 Détermination des dimensions de la carte l'apprentissage de l'algorithme de Kohonen

g. Résultats après l’apprentissage

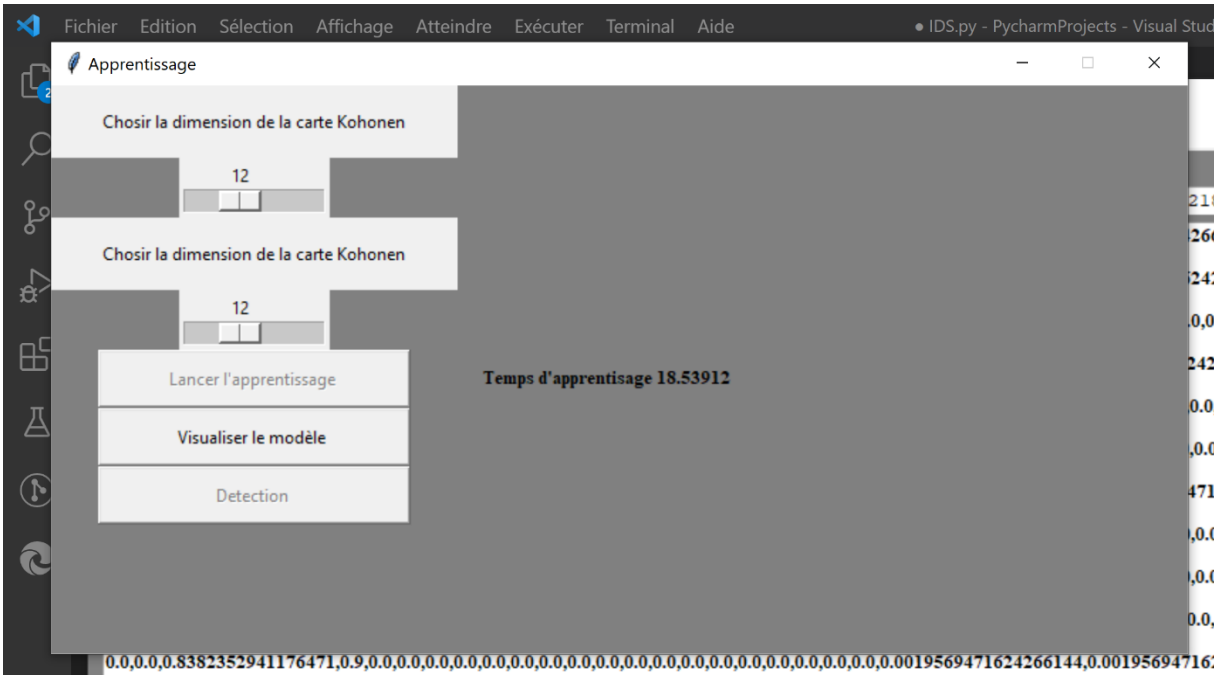

Figure 3-11 Interface après apprentissage

h. Téléchargement de la base de test

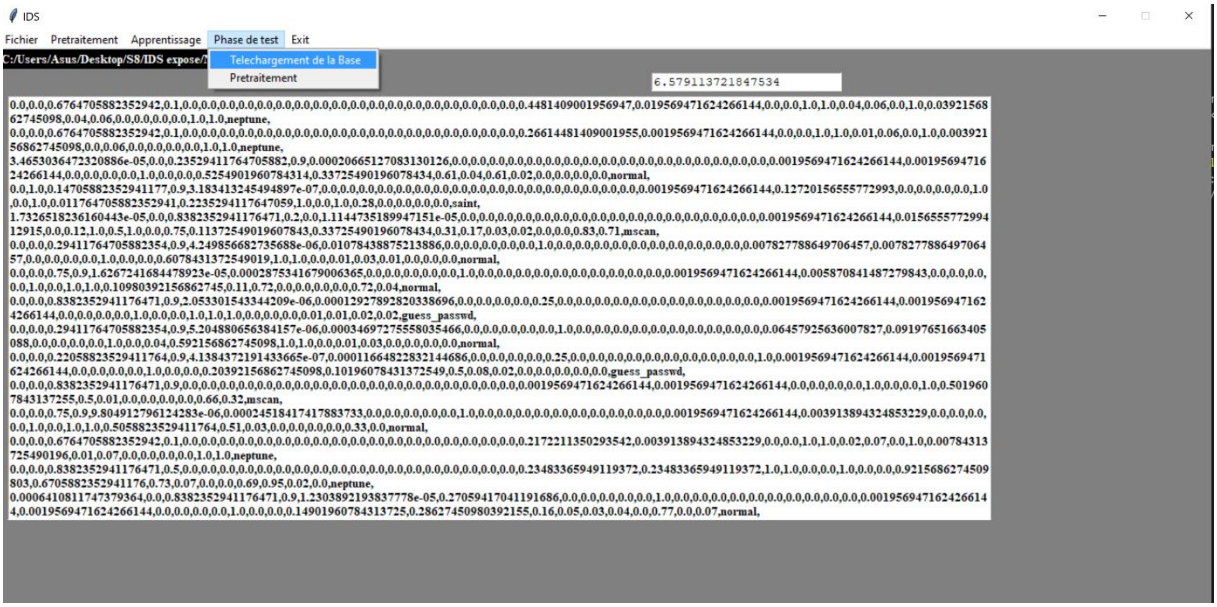

Figure 3-12 Interface de base de test

i. Prétraitement de la base de test

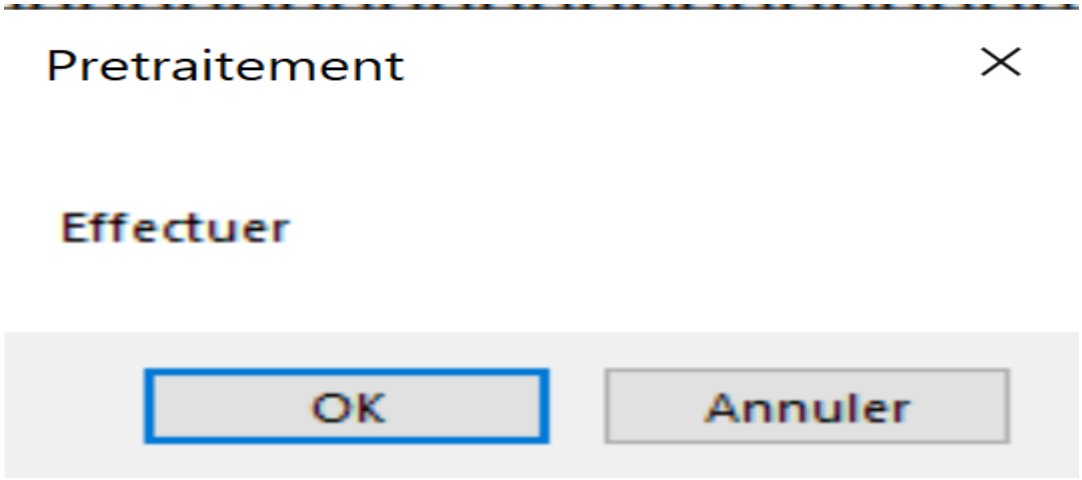

Figure 3-13Prétraitement de la base de test

j. Lancement de la détection

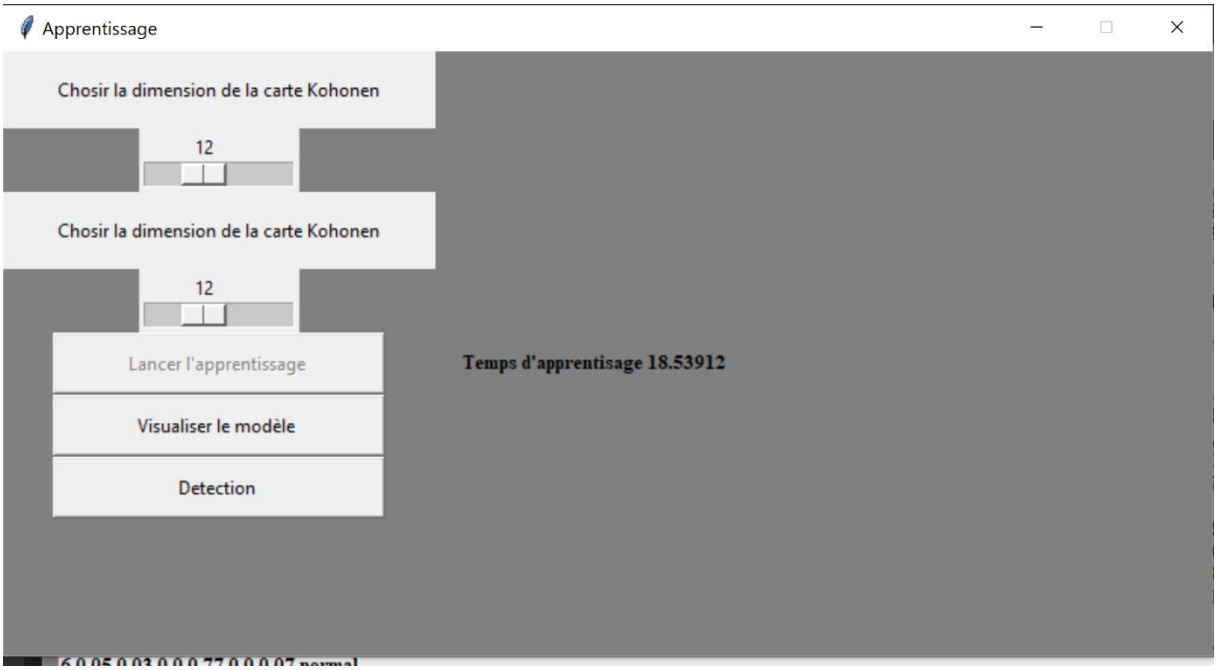

Figure 3-14 Interface de la détection

k. Affichage des résultats de détection

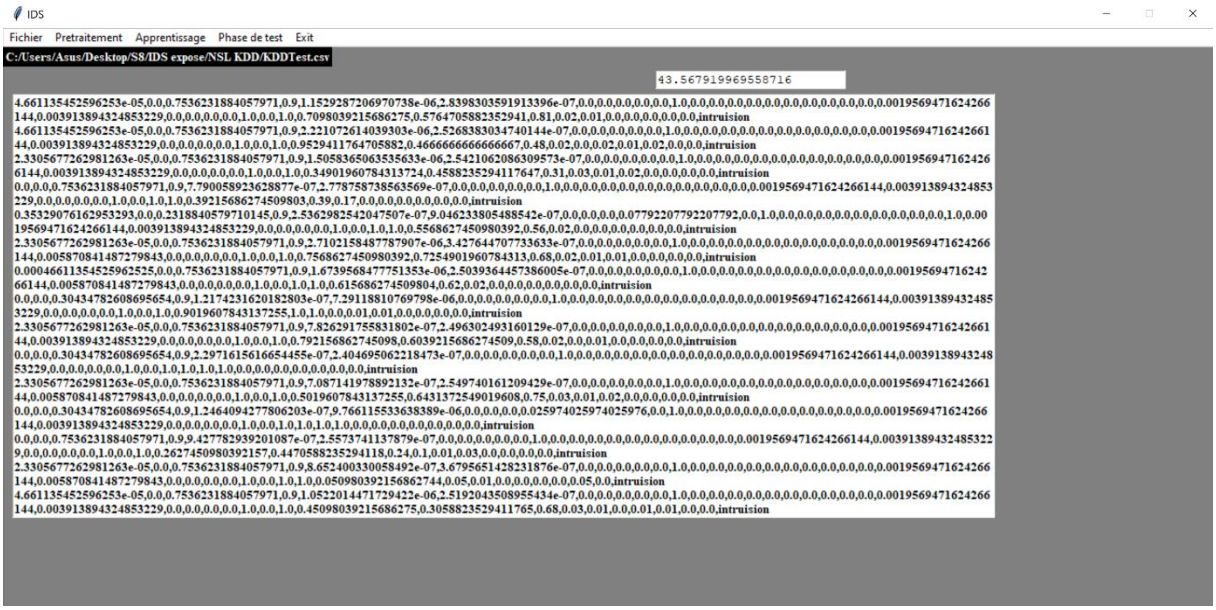

Figure 3-15 Interface d'affichage des résultats de détection d'intrusion

l. Résultats obtenus via notre approche

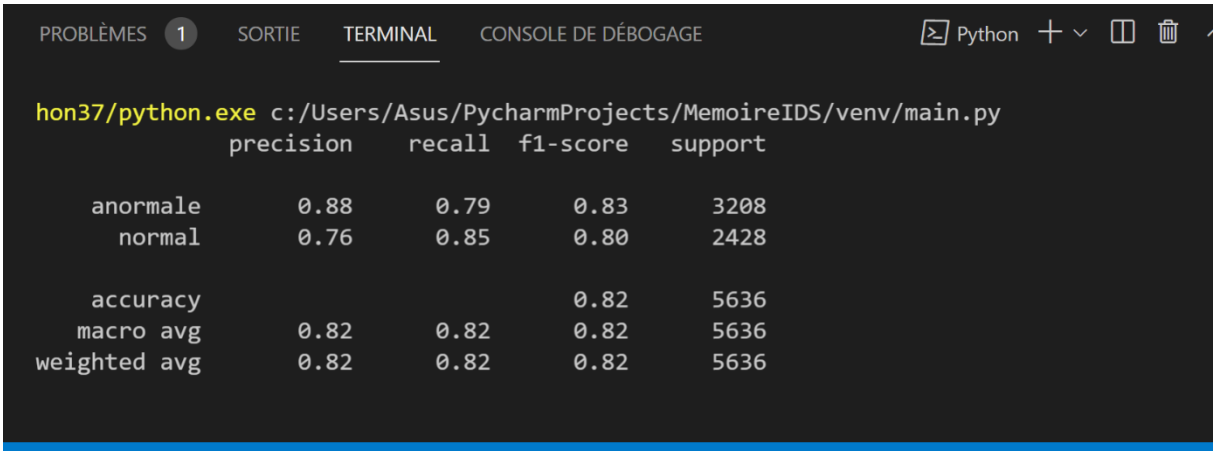

Figure 3-16 Affichage des différents résultats obtenus

3.6 Conclusion

Dans ce chapitre, nous avons présenté en premier lieu, une description de la base de données NSL-KDD que nous avons utilisée dans notre approche proposée, ensuite nous l'avons prétraitée, cependant nous avons détaillé les étapes de son codage et de sa normalisation. Nous avons par la suite procédé à une sélection des caractéristiques, puis nous avons appliqué notre carte de Kohonen. Nous avons également décrit les outils utilisés lors de l'implémentation de notre approche. Nous avons enfin clôturé ce chapitre en illustrant les résultats que obtenus de notre SDI ainsi que les différentes interfaces de notre application.

Conclusion générale

L’internet de nos jours est utilisé par d’innombrables personnes, de ce fait les attaques informatiques augmentent de jour en jours, cette augmentation présente un réel problème pour les réseaux informatiques et tous appareils liés à cette gigantesque toile. Une des solutions qui peuvent assurer la sécurité des réseaux les SDIR. Or dans notre PFE nous avions comme objectif de réaliser un SDIR qui soit à la fois rapide et performant.

Tout au long de ce mémoire, nous nous sommes situés dans une approche basée sur les RNs, plus précisément la carte auto-adaptative de Kohonen qui permet une meilleure compréhension des modèles et d’analyser les comportements des paquets. Les capacités de clustering du SOM permettent l’identification du modèle caché dans les données qui seraient autrement difficiles à détecter.

Nous avions utilisé la base de données NSL-KDD, qui contient une base d’apprentissage et de test, ce qui représente un atout pour pouvoir apprendre au RN puis l’évaluer. Puisqu’il s’agit d’une application en temps réel, nous avons fait recours à une méthode de réduction de dimensionnalité non basée sur le modèle en utilisant la variance.

Afin d’évaluer notre approche nous avons utilisé les estimateurs de la précision, du rappel, du score F1, les résultats obtenus sont satisfaisants en matière de détection.

Comme perspective nous proposons la réalisation d'un modèle de classification via une carte autoorganisée de Kohonen capable de distinguer les types d'attaques au lieu de les classer en deux classes normal et anormal, ainsi qu’une adaptation client-serveur d’où l’attaque provient d’un appareil à distance. Il est aussi possible d’utiliser une autre forme de carte Kohonen supérieur appelé gaz neuronal qui permet de pallier certains problèmes venant de la carte de Kohonen.

.

Bibliographie

ABADI, M. 2018. Réalisation d'un réseau de neurones "SOM" sur une architecture matérielle adaptable et extensible à base de réseaux sur puce "NoC." https://tel.archives-ouvertes.fr/tel-01868313.

ABDALLAH MOHAMMED, C., MOHAMED AMINE, H., AND FATIMA ZOHRA, B. 2020. *Système de Détection D'intrusions Informatiques et la selection des caractéristiques*. .

ABOUD, S. 2009. Soft Computing Techniques for Protecting. *MASAUM Journal of ComputingVolume 1*.

AGATONOVIC-KUSTRIN, S. AND BERESFORD, R. 2000. Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research. *Journal of Pharmaceutical and BiomedicalAnalysis22*, 5, 717–727.

AHMED CHAOUKI LOKBANI. 2017. Le problème de sécurité par le Data Mining. http://rdoc.univ-sba.dz/handle/123456789/201/browse?type=author&order=ASC&rpp=35&value=LOKBANI%2C+Ahmed+Chaouki.

AMUDHA, P., KARTHIK, S., AND SIVAKUMARI, S. 2013. Classification Techniques for Intrusion Detection An Overview. *International Journal of Computer Applications76*, 16, 33–40.

AVAST. 2019. Votre guide essentiel des virus informatiques. *Votre guide essentiel des virus informatiques*. https://www.avast.com/fr-fr/c-computer-virus.

BACE, R. 1999. *Intrusion Detection*. Sams Publishing, Indianapolis, IN.

BACE, R. AND MELL, P. 2001. *NIST special publication on intrusion detection systems*. BOOZ-ALLEN AND HAMILTON INC MCLEAN VA.

BEN BRAHIM, E. AND AMICHE, S. 2017. Mise en place d'une solution de détection d'intrusion. https://wikimemoires.net/2021/02/menaces-informatiques-principales-categories-types/.

BERRI, N. 2019. Utilisation des techniques de Deep Learning pour l'extraction des concepts à partir des documents textuels. .

BLOCH, L., WOLFHUGEL, C., MAKARÉVITCH, N., QUEINNEC, C., AND SCHAUER, H. 2013. *Sécurité informatique: Principes et méthodes à l'usage des DSI, RSSI et administrateurs*. Eyrolles.

BOUROUH, M. AND KANOUN, Z. 2017. Détection d'intrusions à base des réseaux de neurones et algorithmes génétiques. http://dspace.univ-tlemcen.dz/handle/112/12355.

CARLEY, K.M., LAMBADARIS, L., KRANAKIS, E., ET AL. 2009. 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009, Ottawa, Canada, July 8-10, 2009. *CISDA*.

CHERIF, A. 2013. Réseaux de neurones, SVM et approches locales pour la prévision de séries temporelles. http://www.theses.fr/2013TOUR4003.

CISCO. Qu'est-ce qu'un pare-feu ? *Cisco*. https://www.cisco.com/c/fr_fr/products/security/firewalls/what-is-a-firewall.html.

DELUZARCHE, C. 2019. Cyberattaque. *Futura*. https://www.futura-sciences.com/tech/definitions/piratage-cyberattaque-18946/.

FARID, B. 2006. Reconstruction de surfaces d'objets 3D à partir de nuages de points par réseaux de neurones 3D-SOM. http://www.theses.fr/2006LIL10010.

FEI-FEI, L. 2009. ImageNet. https://www.image-net.org/.

GIULIO LAURENTI, P. 2020. Confusion Matrix and Classification Report. *The Startup*. https://medium.com/swlh/confusion-matrix-and-classification-report-88105288d48f.

GNANAPRASANAMBIKAI, L. AND MUNUSAMY, N. 2018. Data Pre-Processing and Classification for Traffic Anomaly Intrusion Detection Using NSLKDD Dataset. *Cybernetics and Information Technologies18*, 3, 111–119.

HAMID, A. AND TASSADIT, A.Z. 2013. Commande par modèle inverse neuronal, application à la commande en vitèsse d'un moteur à courant continu. https://dl.ummto.dz/handle/ummto/7544.

HARDY, C. 2019. Contribution au développement de l'apprentissage profond dans les systèmes distribués. https://tel.archives-ouvertes.fr/tel-02284916.

HOCHREITER, S. AND SCHMIDHUBER, J. 1997. Long Short-Term Memory. *Neural Computation9*, 8, 1735–1780.

KAZIENKO AND DOROSZ. 2003. Intrusion Detection Systems (IDS) : Part I. .

KOHLI, S. 2019. Understanding a Classification Report For Your Machine Learning Model. *Medium*. https://medium.com/@kohlishivam5522/understanding-a-classification-report-for-your-machine-learning-model-88815e2ce397.

KOHONEN, T., NIEMINEN, I.T., AND HONKELA, T. 2009. On the Quantization Error in SOM vs. VQ: A Critical and Systematic Study. *Advances in Self-Organizing Maps*, Springer, 133–144.

LABONNE, M. 2020. Anomaly-based network intrusion detection using machine learning. https://tel.archives-ouvertes.fr/tel-02988296.

LAROCHELLE, H. 2009. Étude de techniques d'apprentissage non-supervisé pour l'amélioration de l'entraînement supervisé de modèles connexionnistes. .

LASKOV, P., DÜSSEL, P., SCHÄFER, C., AND RIECK, K. 2005. Learning Intrusion Detection: Supervised or Unsupervised? *Image Analysis and Processing – ICIAP 2005*, Springer, 50–57.

LEBIGDATA, +BASTIEN. 2021. Python : tout savoir sur le principal langage Big Data et Machine Learning. *LeBigData.fr*. https://www.lebigdata.fr/python-langage-definition.

LECUN, Y., BOTTOU, L., BENGIO, Y., AND HAFFNER, P. 1998. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE86*, 2278–2324.

MUKHOPADHYAY, A., ROY, A., DAS, S., DAS, S., AND ABRAHAM, A. 2008. Population-variance and explorative power of Harmony Search: An analysis. *2008 Third International Conference on Digital Information Management*, 775–781.

MURPHY, K.P. 2012. *Machine Learning: A Probabilistic Perspective*. The MIT Press, Cambridge, MA.

NORTON. 2021. Qu'est-ce qu'un logiciel antivirus et en ai-je besoin ? *norton.com*. https://fr.norton.com/internetsecurity-malware-what-is-antivirus-and-do-i-need-it.html.

PATRICE, W. 2009. Réseaux de neurones artificiels : architectures et applications. https://docplayer.fr/9536135-Reseaux-de-neurones-artificiels-architectures-et-applications.html.

PYTHON INSTITUTE. 2018. About Python | Python Institute. https://pythoninstitute.org/what-is-python/.

SABRY, A.H., BACHA, A., AND BENHRA, J. 2015. Utilisation des réseaux de neurones pour le tuning des Algorithmes d'optimisation par colonies de fourmis Application aux chaines logistiques. *Xème Conférence Internationale : Conception et Production Intégrées*.

SELEZNYOV, A. 2002. An anomaly intrusion detection system based on Intelligent user recognition. *Jyväskylä studies in computing* 22.

SIMONYAN, K. AND ZISSERMAN, A. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. .

SU, M.-C., LIU, T.-K., AND CHANG, H.-T. 1999. An efficient initialization scheme for the self-organizing feature map algorithm. 1906–1910 vol.3.

SYNGRESS. 2003. *Cisco Security Professional's Guide to Secure Intrusion Detection Systems*. Syngress.

SZEGEDY, C., LIU, W., JIA, Y., ET AL. 2014. Going Deeper with Convolutions. *arXiv:1409.4842 [cs]*.

TARAS, I. AND STUART, D.M. 2019. Quantization Error as a Metric for Dynamic Precision Scaling in Neural Net Training. *arXiv:1801.08621 [cs]*.

UNB. 2019. NSL-KDD | Datasets | Research | Canadian Institute for Cybersecurity | UNB. https://www.unb.ca/cic/datasets/nsl.html.

VISUALSTUDIOCODE. 2020. Why Visual Studio Code? https://code.visualstudio.com/docs/editor/whyvscode.

WEBNET. 2019. Visual Studio Code. *Le blog technique Webnet*. https://blog.webnet.fr/visual-studio-code/.

WU, P. AND GUO, H. 2019. LuNet: A Deep Neural Network for Network Intrusion Detection. *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 617–624.

WU, P., GUO, H., AND BUCKLAND, R. 2019. A Transfer Learning Approach for Network Intrusion Detection. .

YENDE, R. 2018. SUPPORT DE COURS DE SÉCURITÉ INFORMATIQUE ET CRYPTO. 141.

Résumé

L’informatique et en particulier l’Internet jouent un rôle grandissant dans notre société. Un grand nombre d’applications critiques d’un point de vue de leur sécurité sont déployées dans divers domaines comme le domaine militaire, la santé, le commerce électronique, etc. La sécurité des systèmes informatiques devient alors une problématique essentielle tant pour les individus que pour les entreprises ou les états. La mise en place d’une politique de sécurité autour de ces systèmes est donc primordiale. Les pares-feux ne sont aujourd’hui plus suffisants. Les Systèmes de Détection d’Intrusion (SDIs) sont capables de repérer des menaces que les pares-feux ne soupçonnent pas.

L’objectif de ce travail est de proposer des mécanismes de détection d’intrusion en temps réel tout en augmentant le taux de précision et en réduisant les fausses alarmes ainsi que le temps de réponse en faisant appel aux Réseaux de Neurones Artificiels (RNAs).

Les RNAs sont des paradigmes issus de l’Intelligence Artificielle (IA) qui ont montré leur efficacité en classification. Dans le cadre de ce travail nous proposons une méthode de détection d’intrusion en utilisant un algorithme issu de l’intelligence artificielle appliquant un apprentissage automatique, il s’agit de la Carte auto-organisatrice de Kohonen. Afin d’évaluer notre approche nous avons utilisé la base de données NSL-KDD.

Notre approche proposée permet de classer les paquets comme normaux ou intrusifs. Les résultats expérimentaux montrent que le système est performant avec un taux de reconnaissance satisfaisant.

Mots Clés :

Sécurité Informatique (SI), Systèmes de Détection d'Intrusion (SDI), Réseau de Neurones Artificiel (RNA), carte auto-adaptative de Kohonen, sélection des caractéristiques, Intelligence Artificiel (IA), classification.

Abstract

Computing and in particular the Internet play an increasing role in our society. A large number of security-critical applications are deployed in various fields such as military, healthcare, e-commerce, etc. The security of computer systems then becomes an essential issue for individuals as well as for companies or states. The establishment of a security policy around these systems is therefore essential. Firewalls are no longer sufficient. Intrusion Detection Systems (SDIs) are able to spot threats that firewalls don't suspect.

The objective of this work is the realization of an intrusion detection system (IDS) based on artificial intelligence, machine learning using arterial neural networks (Kohonen self-organizing card) which thanks to the implementation of its algorithm allowed us to deal with intrusions.

As part of this work we propose a detection method using the NSL-KDD database. The algorithm used allows you to classify the data as normal or abnormal. The experimental results show that the system is efficient with a satisfactory recognition rate.

Keywords:

IT Security, Intrusion Detection Systems (SDI), Artificial Neural Network (RNA), Artificial Intelligence (AI), Self-Organizing Map, Feature Selection, classification.