Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire

Université Abou Bakr Belkaid– Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en Informatique

Option: *Système d'Information et de Connaissance (S.I.C)*

Thème

Détection d'intrusions à base des réseaux de neurones et algorithmes génétiques

Réalisé par :

- **BOUROUH Mouloud**
- **KANOUN Zakaria**

Présenté le : *03 juillet 2017 devant le jury composé de MM.*

- ***Mr : MAHFOUD Houari****. ……* *(Président)*
- ***Mr :GHOUTI Abdellaoui****……* *(Encadreur)*
- ***Mme :KHITRI Souad.*** *……* *(Examinatrice)*

Année universitaire : 2016-2017

Remerciement

En premier lieu, nous remercions Dieu le très haut qui nous a donné le courage et la volonté de réaliser ce modeste travail.

Nous tenons à saisir cette occasion et adresser nos profonds remerciements et nos profondes reconnaissances à toutes personnes qui nous ont aidé de près ou de loin dans la réalisation de ce mémoire.

Nous remercions M. Ghouti Abdellaoui, en tant que Directeur de mémoire, pour ses précieux conseils et son orientation tout au long de notre recherche.

Nos remerciements aux membres de jury qui ont accepté de juger notre travail.

Nos remerciements s'étendent également à M. Ammar Boulaiche, professeur à l'université de Jijel pour ses orientation et conseils.

Enfin nous exprimons notre profonde reconnaissance à tous responsables et enseignants de l'université de Tlemcen qui ont contribué à notre formation.

Dédicace

Je dédie ce modeste travail

A tous ceux qui me connaissent, en particulier,

A mon père et à la mémoire de ma mère.

A ma femme et mes deux enfants ;

Imad et Nour el Houda.

A mes frères et sœurs.

A tous mes amis et collègues.

Sans oublier tous les professeurs qui ont contribué à ma formation de l'enseignement primaire, moyenne et secondaire jusqu'à l'enseignement supérieur.

Mouloud

Dédicace

Je dédie ce modeste travail à :

A tous ceux qui, de près ou de loin, ont participé à mon éducation, m'ont aidé Dans les moments difficiles à surmonter mes problèmes.

A ceux qui ont partagé avec moi mes bonheurs et mes Soucis.

A mes très chers parents pour leurs Sacrifices et leur Soutien durant toutes mes années d'étude.

A ma très chère sœur Hadjer.

Et toute la famille « KANOUN ».

A tous mes amis du plus proche au plus loin.

Zakaria

Sommaire

Chapitre 3 : Optimisation et algorithmes génétiques

Chapitre 4 : Implémentation et discussion des résultats

Liste des Figures

Liste des Tableaux

- Introduction générale :

Le développement remarquable du domaine des nouvelles technologies de l'information et de la communication (NTIC) ces dernières années, et l'utilisation de l'outil informatique à grande échelle, plus l'accessibilité du réseau internet par un grand nombre d'utilisateurs avec leurs différentes intentions qui peuvent être parfois destructifs, cela rend les données sensibles ainsi que les ressources des utilisateurs et des sociétés vulnérables au vol, ou exploités pour des raisons malveillantes. Face à toutes ces menaces, la sécurité optimale des systèmes informatiques et des réseaux est devenue un enjeu stratégique et pour assurer cette sécurité, différents outils ont été utilisés, tels que les pare-feu et les anti-virus.

Malheureusement les systèmes antivirus ou les firewalls sont la plupart du temps inefficaces face à ces nouvelles menaces sophistiquées, dont la propagation peut s'avérer extrêmement rapide. C'est pour pallier ce manque que sont apparus récemment de nouvelles solutions de sécurité appelées systèmes de détection des intrusions(IDS) qui consistent à examiner le trafic réseau, collecter tous les événements, les analyser et générer des alertes en cas d'identification de tentatives malveillantes. Ces systèmes sont devenus jour après jour très utilisés dans les stratégies de sécurité des réseaux et systèmes informatiques. Néanmoins, le défi majeur pour les systèmes de détection d'intrusions réside dans leur capacité à déterminer tout comportement malicieux que ce soit de l'intérieur ou de l'extérieur du système informatique.

En effet, le domaine de la détection d'intrusion est très ouvert à la recherche et au développement, où des produits logiciels et des solutions pratiques commencent à apparaitre jour après jour et qui fonctionnent selon deux principaux modes: l'approche par scénario et l'approche comportementale.

L'approche par scénario, basée sur la comparaison du comportement d'utilisation du système avec des signatures d'attaques connues préalablement et ne permet pas la détection des nouvelles attaques sans mise à jour de la base de signatures ce qui représente un inconvénient majeur de cette approche.

Par contre, l'approche comportementale consiste à construire un modèle identifiant les comportements déviants du modèle comportemental normal. Ce modèle est le résultat d'une phase d'apprentissage sur une grande base de données et son avantage principal est la possibilité de détecter de nouvelles attaques.

Dans ce mémoire, nous essayons de réaliser un système de détection d'intrusion comportemental, capable de détecter des nouvelles attaques avec le maximum de taux de réussite et moindre fausses alarmes. Et dans ce but nous avons combiné les réseaux de neurones artificiels qui représentent un outil d'intelligence artificielle pour l'apprentissage et la classification avec l'algorithme génétique comme technique d'optimisation. Cette combinaison a prouvé sa performance dans de nombreux domaines d'applications et dans la résolution de problèmes variés.

Chapitre 1 :

Sécurité informatique et systèmes de détection d'intrusions

- Introduction :

Le progrès technologique, le développement des moyens de communications, l'ouverture du monde sur nouvelles technologies, et la transmission de divers types de données à travers les réseaux, ainsi que d'autre facteurs, apportent un danger d'accès et de manipulation des données par des personnes non autorisés, ou des concurrents. Donc la sécurité de l'information par une gamme de techniques et mécanismes d'authentification et de contrôle d'accès est devenue un besoin crucial afin de construire un système sécurisé déterminant et éliminant ces vulnérabilités.

Le système de détection des intrusions (IDS) est l'une de ces techniques qui offre un contrôle permanent des attaques ou suspectes et permettant ainsi de détecter toute tentative de violation de la politique de sécurité, c'est-à-dire toute intrusion.

Dans ce premier chapitre nous présentons deux parties, la première présente les principales notions de base de la sécurité informatique et les systèmes de détection d'intrusions, en commençant par les définitions des différentes notions de la sécurité informatique, puis les attaques informatiques et leurs classifications avec exemples. La deuxième partie présente les systèmes de détection d'intrusions, leur historique, définition, principe de fonctionnement, et critères de classification, ...etc.

A la fin de ce chapitre, nous citons quelques avantages et limites des systèmes de détection d'intrusions actuels.

I- La sécurité informatique :

1. Définitions :

La sécurité informatique : *« C'est l'ensemble des moyens mis en œuvre pour minimiser la vulnérabilité d'un système contre des menaces accidentelles ou intentionnelles. Elles caractérisent ce à quoi s'attendent les utilisateurs de systèmes informatiques en regard de la sécurité, La sécurité informatique vise généralement les cinq principaux objectifs suivants : (appelés aussi les propriétés de la sécurité informatiques »* **[1].**

- **Disponibilité** : demande que l'information sur le système soit disponible aux personnes autorisées.
- **Confidentialité** : permet d'assurer que l'information sur le système ne puisse être lue que par les personnes autorisées.
- **Intégrité** : L'intégrité permet de certifier que l'information sur le système ne puisse être modifiée que par les personnes autorisées (pas de divulgation à des tiers non autorisés).
- **Non-répudiation**: les acteurs impliqués dans la communication ne peuvent nier y avoir participé.

- **L'authentification** : l'authentification consiste à assurer l'identité d'un utilisateur, c'est-à-dire de garantir à chacun des correspondants que son partenaire est bien celui qu'il croit être. Un contrôle d'accès peut permettre (par exemple par le moyen d'un mot de passe qui devra être crypté) l'accès à des ressources uniquement aux personnes autorisées.

On retrouve aussi dans le domaine de la sécurité informatique l'utilisation notamment des termes suivants, vulnérabilité, intrusion, menace, attaque**. [2][3]**

- **Vulnérabilité** : faute créée durant le développement du système, ou durant l'opération, pouvant être exploitée afin de créer une intrusion.
- **Intrusion** : faute malveillante externe résultant d'une attaque qui a réussi à exploiter une vulnérabilité.
- **Menace** : possibilités et probabilités d'attaque contre la sécurité. Une menace est définie par le processus d'attaque, par la cible et par le résultat (conséquences de la réussite d'une attaque**).**
- **Attaque**: C'est n'importe quelle action qui a le but de menacer la sécurité des informations et de nuire au moins à l'une des propriétés de la sécurité informatique (disponibilité, Confidentialité, Intégrité, L'authentification). Il s'agit d'une tentative d'intrusion, nous abordons dans ce qui suit les différents buts et classes de ces attaques (tentatives d'intrusion).

2-Buts des attaques informatique**:**

Il existe plusieurs objectifs pour les attaques:

- **Interruption**: vise la disponibilité des informations (DoS, . . .)
- **Interception**: vise la confidentialité des informations (capture de contenu, analyse de trafic,. . .).
- **Modification**: vise l'intégrité des informations (modification, rejet,. . .).
- **Fabrication:** vise l'authenticité des Informations (Masquerade). **[5]**

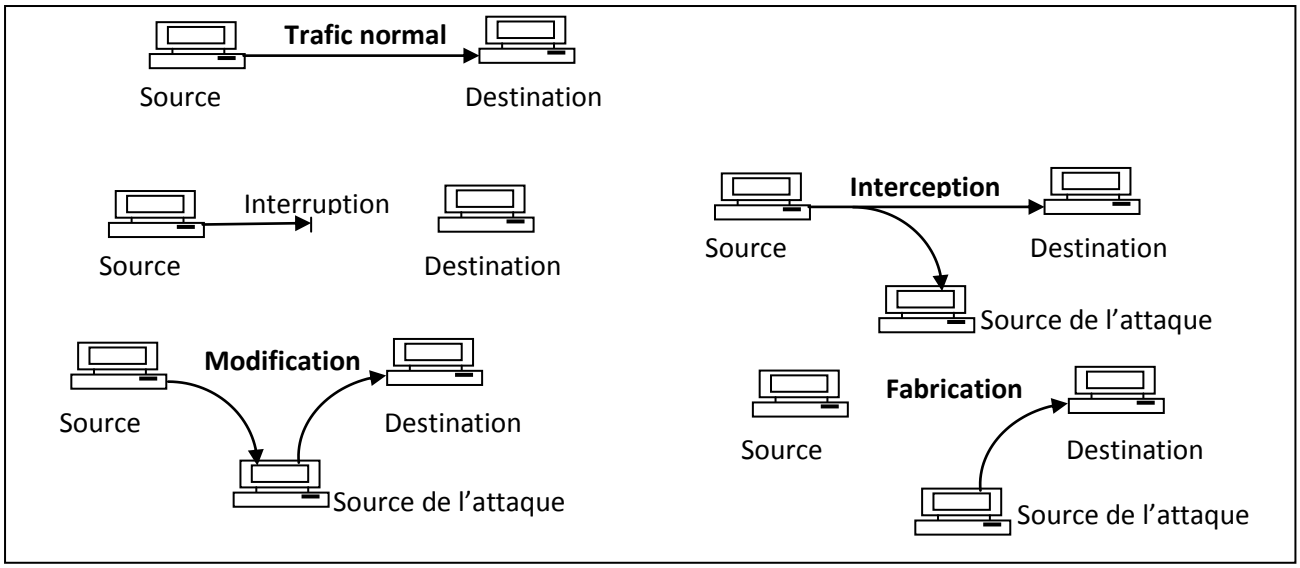


Figure 1.1 : Buts des attaques Informatiques [13]

3- Les différentes classes d'attaques informatiques:

Du point de vue de la sécurité informatique, une menace est une violation potentielle de la sécurité. Cette menace peut être accidentelle, intentionnelle (attaque), active ou passive, Ils existent dans la littérature plusieurs classifications d'attaques informatique selon des critères différents, parmi lesquelles :

3.1-Classification selon l'effet de l'attaque : Selon les effets résultant de l'attaque on peut classifier les attaques en deux groupes principaux : les attaques passives et les attaques actives.

- **Les attaques passives** : consistent à accéder, utiliser ou à observer le système cible sans modifier les données ou dysfonctionné les ressources de ce dernier, elles sont généralement indétectables (ex. : capture de contenu, analyse de trafic).
- **Les attaques actives** : consistent à effectuer des changements non autorisés sur les données des systèmes, à s'introduire dans des équipements réseau ou à perturber leurs fonctionnements, les attaques de ce type sont bien évidemment plus dangereuses.(ex. : mascarade et déni de service).

3.2-Classification selon la source de l'attaque : En termes de relation intrusion-victime, les attaques sont classées comme suit:

- **Les attaques internes :** provenant des employés de leur entreprise ou de leurs partenaires commerciaux ou clients,
- **Les attaques externes** : venant de l'extérieur, fréquemment via Internet.

3.3-Classification selon la cible de l'attaque:

- **Les attaques réseaux** : Les attaques réseaux s'appuient sur des vulnérabilités liées directement aux protocoles ou à leur implémentation. Il existe un grand nombre. Néanmoins, la plupart d'entre elles ne sont que des variantes des cinq attaques réseaux les plus connues aujourd'hui.
- **Les attaques applicatives :** Les attaques applicatives s'appuient principalement sur des vulnérabilités spécifiques aux applications utilisées.

4- Exemples d'attaques:

Il existe un nombre énorme d'attaques qui menacent les systèmes et les réseaux informatiques, néanmoins, la plupart d'entre elles ne sont que des variantes des autres.

Voici des exemples d'attaques les plus connues aujourd'hui ciblant les réseaux informatiques.

4.1- Attaques de Dénis de Services :(Denial Of Service [DOS]): est une attaque informatique ayant pour but de rendre indisponible un service, d'empêcher les utilisateurs légitimes d'un service de l'utiliser. Il peut s'agir de :

- L'inondation d'un réseau afin d'empêcher son fonctionnement ;
- La perturbation des connexions entre deux machines, empêchant l'accès à un service particulier ;

- L'obstruction d'accès à un service à une personne en particulier ;
- Également le fait d'envoyer des milliards d'octets à un box internet.

L'attaque par déni de service peut ainsi bloquer un serveur de fichiers, rendre impossible l'accès à un serveur web ou empêcher la distribution de courriel dans une entreprise, les principales attaques qu'on peut trouver sont Apache2, Back,Land, Mail bomb, SYN Flood, Ping of death, Process table, Smurf, Syslogd, Teardrop,Udp storm.

4-2-Probing (Sondage) : L'attaquant de cette classe commence par un sondage de la future victime, ce que l'on appelle scan, ce sondage va balayer chaque port IP afin de connaitre les services offerts par le système (OS, topologie du réseau, protections employées,..) une fois se achevé, la machine de l'intrus (celui qui réalise l'intrusion) tente alors d'identifier le système d'exploitation utilisé par cette victime et d'exploiter les informations qu'elle a récolté. Cette classe d'attaque est la plus étendue et qu'elle requit une expertise technique minime. Les exemples de ce type d'attaque sont :Ipsweep,Mscan, Nmap,Saint, Satan.

4-3- Attaques 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, Ffbconfig, Fdformat, Load module, Perl, Ps, Xterm.

4-4- Attaque 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.

4-5-L'usurpation d'adresse IP(IP Spoofing) : Le principe de fonctionnement de cette attaque est d'envoyer des paquets IP en utilisant une adresse IP source qui n'a pas été allouée à l'ordinateur qui émet ces paquets pour le but de masquer l'identité de l'attaquant lors d'une attaque d'un serveur ou n'importe quel cible dans le réseau , ou d'usurper l'identité d'un autre équipement du réseau pour bénéficier des services auxquels il a accès.

4-6-Les analyseurs réseau (sniffer): Est un dispositif permettant d'« écouter » le trafic d'un réseau, c'est-à-dire de capturer les informations qui y circulent, vu que les données dans un réseau non commuté sont envoyées à toutes les machines du réseau et dans une utilisation normale les machines ignorent les paquets qui ne leur sont pas destinés. Le sniffer peut également servir cette propriété à une personne malveillante ayant un accès physique au réseau pour collecter des informations (ex : les mots de passes),

Maisun sniffer peut aussi être utilisé comme un outil positif pour le but d'étudier et de capturer le trafic d'un réseau par les administrateurs réseaux et les détecteurs d'intrusion (IDS).

4.7-Balayage des ports : (port scanning) est une des activités considérées comme suspectes servant par les pirates informatiques pour découvrir les faiblesses potentiellement exploitables et chercher les ports ouverts sur un serveur de réseau en balayant les ports disponibles de la victime qui est potentiellement exécute de nombreux 'services' qui écoutent des 'ports' connus, les balayages de ports se font habituellement sur le protocole TCP pour le but d'ouvrir des connexions pour effectuer une intrusion, la même technique de balayage des ports est aussi utilisée par les administrateurs des systèmes informatiques pour contrôler la sécurité des serveurs de leurs réseaux.

4.8-TCP Session Hijacking : Le « **vol de session TCP** » est une technique consistant à intercepter une session TCP initiée entre deux machines afin de la détourner, dans la mesure où le contrôle d'authentification s'effectue uniquement à l'ouverture de la session, un pirate réussissant cette attaque parvient à prendre possession de la connexion pendant toute la durée de la session.

4.9-Les trappes (backdoor):C'est une fonction ou un programme permettant à un pirate de prendre le contrôle d'un ordinateur à distance. Il peut être placé dans un cheval de Troie[1] ou un virus.

4.10-Attaque par virus : Il s'agit d'un programme auto-reproductible et généralement destructeur qui contamine le disque dur ainsi que tous autres supports de stockage utilisés et qui peut faire exécuter à l'ordinateur des actions non désirées, le virus informatique peut donc se propager à l'intérieur même de l'ordinateur, en infectant petit à petit tous les fichiers. Il est donc destiné à modifier à notre insu le fonctionnement de l'ordinateur, certains virus peuvent simplement faire «beeper» le PC, d'autres peuvent détruire les données (formater, effacer le secteur de démarrage, voir détruire le matériel). **[4]**

5-Mécanismes de défense contre les attaques :

C'est l'ensemble de procédures ou dispositifs qui sont conçu pour détecter, prévenir ou récupérer les attaques qui menacent la sécurité informatique, il existe plusieurs outils de prévention contre-attaques informatiques, Nous avons cité ci-dessous quelques mécanismes**: [13].**

- **Chiffrement :** Algorithme généralement basé sur des clefs et transformant les données. Sa sécurité est dépendante du niveau de sécurité des clefs.
- **Signature numérique**: Données ajoutées pour vérifier l'intégrité ou l'origine des données.
- **Bourrage de trafic** : Données ajoutées pour assurer la confidentialité, notamment au niveau du volume du trafic.

1**Chevaux de Troie**: sont des programmes qui en plus d'une fonction classique, ont une fonction cachée nuisible, récupérer vos mots de passe, détruire le disque dur, etc.

- **Notarisation** : Utilisation d'un tiers de confiance pour assurer certains services de sécurité.
- **Contrôle d'accès** : Vérifie les droits d'accès d'un acteur aux données. N'empêche pas l'exploitation d'une vulnérabilité.
- **Antivirus** : Logiciel censé protéger l'ordinateur contre les logiciels (ou fichiers potentiellement exécutables) néfastes. Ne protège pas contre un intrus qui emploie un logiciel légitime, ou contre un utilisateur légitime qui accède à une ressource alors qu'il n'est pas autorisé à le faire.
- **Le pare-feu** : Un élément (logiciel ou matériel) du réseau informatique contrôlant les communications qui le travers. Il a pour fonction de faire respecter la politique de sécurité du réseau, celle-ci définissant quelles sont les communications autorisées ou interdites. N'empêche pas un attaquant d'utiliser une connexion autorisée pour attaquer le système. Ne protège pas contre une attaque venant du réseau intérieur (qui ne le traverse pas).
- **Détection d'intrusion** : Repère les activités anormales ou suspectes sur le réseau surveillé. Ne détecte pas les accès incorrects mais autorisés par un utilisateur légitime. Mauvaise détection : taux de faux positifs, faux négatifs.
- **Journalisation** ("logs") : Enregistrement des activités de chaque acteur. Permet de constater que des attaques ont eu lieu, de les analyser et potentiellement de faire en sorte qu'elles ne se reproduisent pas.
- **Analyse des vulnérabilités** ("Security audit") : Identification des points de vulnérabilité du système. Ne détecte pas les attaques ayant déjà eu lieu, ou lorsqu'elles auront lieu. **[4]**

Mais Aucun des mécanismes de sécurité ne suffit par lui-même, et pour cela dans la plupart du temps en vue d'atteindre un niveau acceptable de sécurité informatique plusieurs mécanismes sont utilisés en même temps.

II-Systèmes de détection et de prévention des intrusions :

1-Historique:

Les systèmes de détection ont connu une croissance rapide, le concept de système de détection d'intrusions a été introduit en 1980 par James Anderson dans l'effort d'amélioration de la vérification de la sécurité informatique et la capacité de surveillance, puis complété par le premier modèle de détection d'intrusions établit par Denning Dorothy en 1987, ensuite de nombreux prototypes sont apparus depuis 1988, et des grands budgets sont investis pour la recherche dans ce domaine. **[12]**

2-SYSTEMES DE DETECTION D'INTRUSIONS (IDS) :

La détection d'intrusion concerne l'ensemble des pratiques et mécanismes utilisés pour la détection d'erreurs pouvant conduire à une défaillance de sécurité, et/ou pour la détection d'attaques. Un IDS est l'implémentation des pratiques et mécanismes de détection d'intrusion. Les IDS ont donc pour rôle de détecter et/ou bloquer (on parle dans ce cas d'Intrusion Prévention System) des attaques survenantes au sein d'un système ou d'un réseau. Ils peuvent être déployés sur l'hôte surveillé, on parle alors de **H**ost-Based **I**ntrusion **D**etection **S**ystem (**HIDS**), ou bien sur le réseau surveillé, on parle alors de **N**etwork-Based **I**ntrusion **D**etection System (**NIDS**) **[2] [6]**.

3- Architecture des IDS et Principes de fonctionnement:

3.1-Architecture des IDS :

Plusieurs architectures ont été proposées pour décrire les différents éléments intervenants dans un système de détection d'intrusion. L'architecture la plus simple est composée de trois modules : la source de données, l'analyseur des données et le module des réponses. L'architecture générale d'un IDS proposée par **IDWG (I**ntrusion **D**étection **W**orking **G**roup) est montrée dans la (Figure 1.2).

Dans l'architecture proposée par le groupe IDWG de l'IETF[2]on trouve les trois modules cités précédemment couplés avec d'autres composants, Dans cette architecture, l'objectif été la définition d'un standard de communication entre les composants du système de détection d'intrusion. Cette architecture définit un format d'échange de message pour les IDS : **I**ntrusion **D**étection **M**essage **E**xchange **F**ormat (**IDMEF**), qui contient implicitement un modèle de données. Proposée par IDWG.

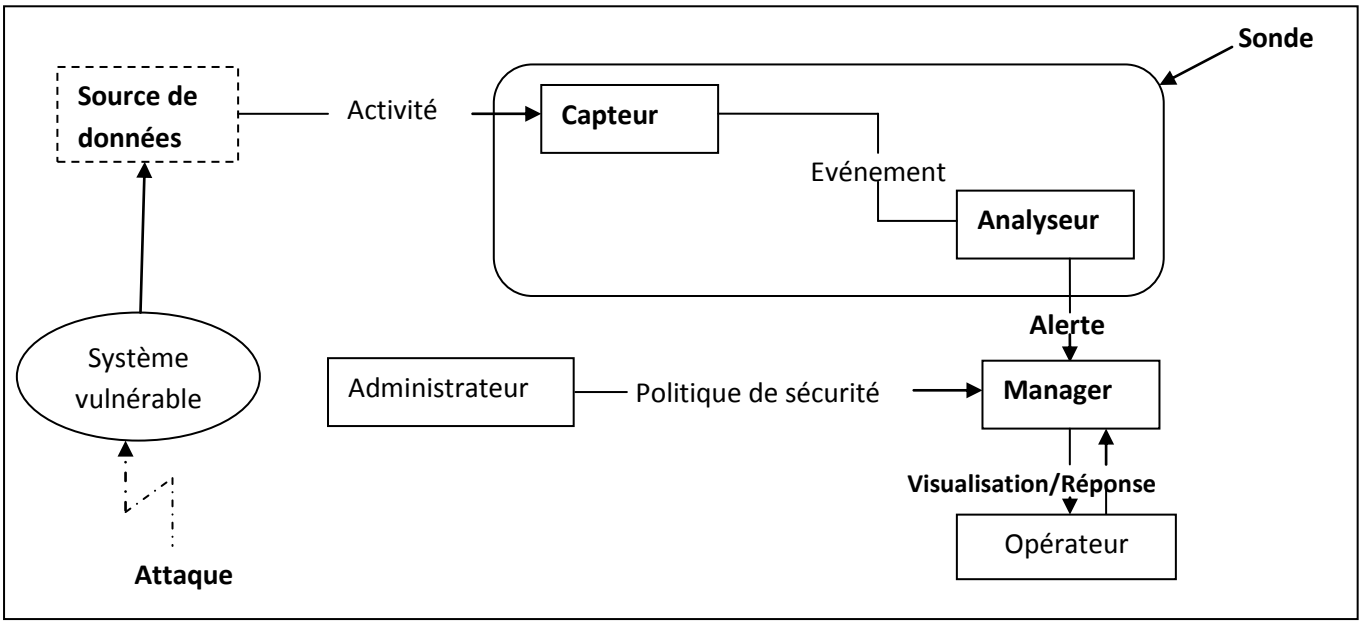


Figure 1.2 : Architecture d'un IDS proposée par IDWG [7]

2L'**I**nternet **E**ngineering **T**ask**F**orce, (**IETF)**: est un groupe informel, international, ouvert à tout individu, qui participe à l'élaboration des standards Internet. L'IETF produit la plupart des nouveaux standards d'Internet.

Cette architecture est composée des modules suivants :

- **Source de données**: C'est l'interface entre le système surveillé et l'IDS, elle fait la collecte d'informations sur les activités du système.
- **Capteur**: Chargé de filtrer et formater les informations brutes envoyées par la source de données. Le résultat de ce traitement sera un message formaté, appelé aussi événement, il représente l'unité de base dans un scenario d'attaque.
- **Analyseur**: Permet d'analyser les événements générés par le capteur. S'il détecte une activité intrusive il émet une alerte, qui est un message sous un format standard. Dans cette architecture, le capteur et l'analyseur forment ensemble une sonde.
- **Alertes :** Lorsqu'un IDS détecte une intrusion, il doit la signaler à l'administrateur à travers les alertes, Ces dernières générées par les IDS sont généralement stockées dans les journaux du système ou utilisés pour prendre des actions contre les attaques (cela dépend du type d'IDS : à réaction active ou passive). Cependant il existe une norme qui permet d'en formaliser le contenu, afin de permettre à différents éléments de sécurité d'inter-opérer. Ce format s'appelle **IDMEF** [3] (**I**ntrusion **D**étection **M**essage **E**xchange **F**ormat), où il est possible de les visualiser ultérieurement par un expert de sécurité**.**
- **Manager**: En plus de la notification des alertes, il offre à l'administrateur la possibilité de configurer une sonde et de gérer les alertes envoyées par l'analyseur.

3.2- Principe de fonctionnement des IDS :

La Figure 1.3 illustre le fonctionnement d'un IDS et l'enchainement de ses actions lors de la détection des intrusions**.**

3IDMEF (**I**ntrusion **D**étection **M**essage **E**xchange **F**ormat) Utilisé dans le cadre de la sécurité informatique, est un format de données servant à échanger des informations de sécurité collectées par les logiciels de détection d'intrusion et de prévention d'intrusion avec les systèmes de management qui communiquent avec eux.

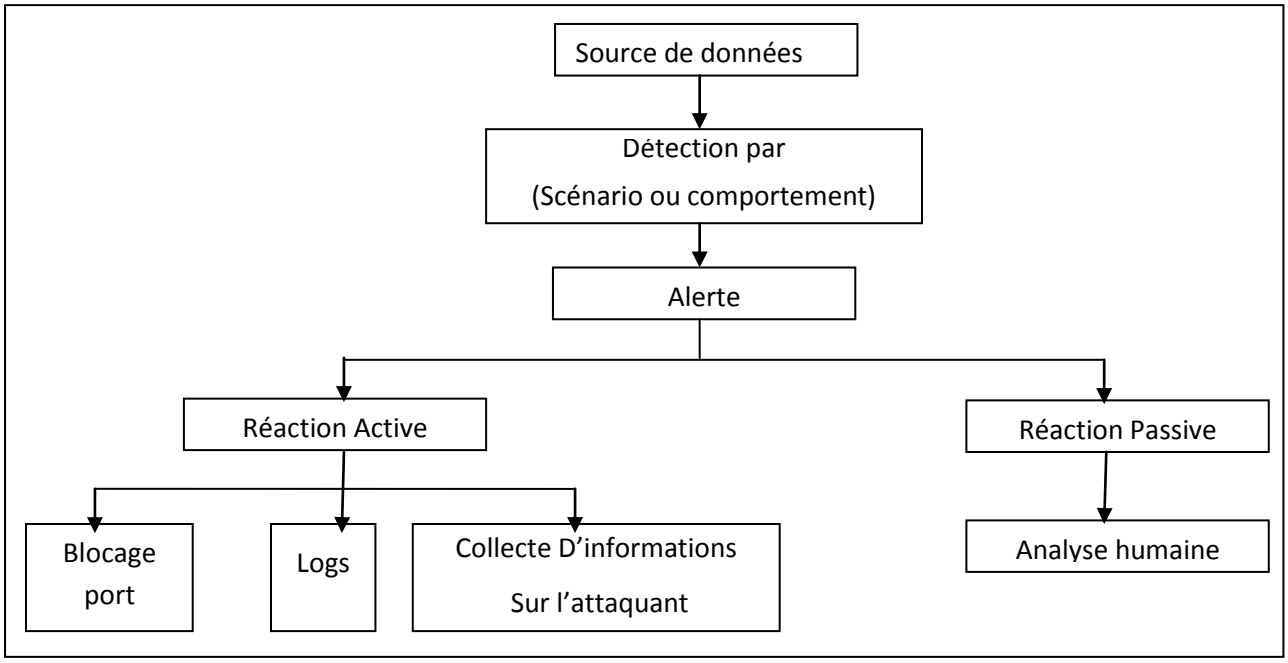


Figure1.3 : Fonctionnement d'un IDS [8]

4- Emplacement de l'IDS :

Il est très important de faire bien positionner le système de détection d'intrusion, cela nécessite de bien identifier les ressources à protéger et ce qui est le plus susceptible d'être attaqué, Il convient alors d'implémenter précautionneusement dans la zone convenable.

Il existe plusieurs endroits stratégiques où il convient de placer un IDS.

La Figure 1.4 illustre un réseau local ainsi que les trois positions que peut y prendre un IDS :

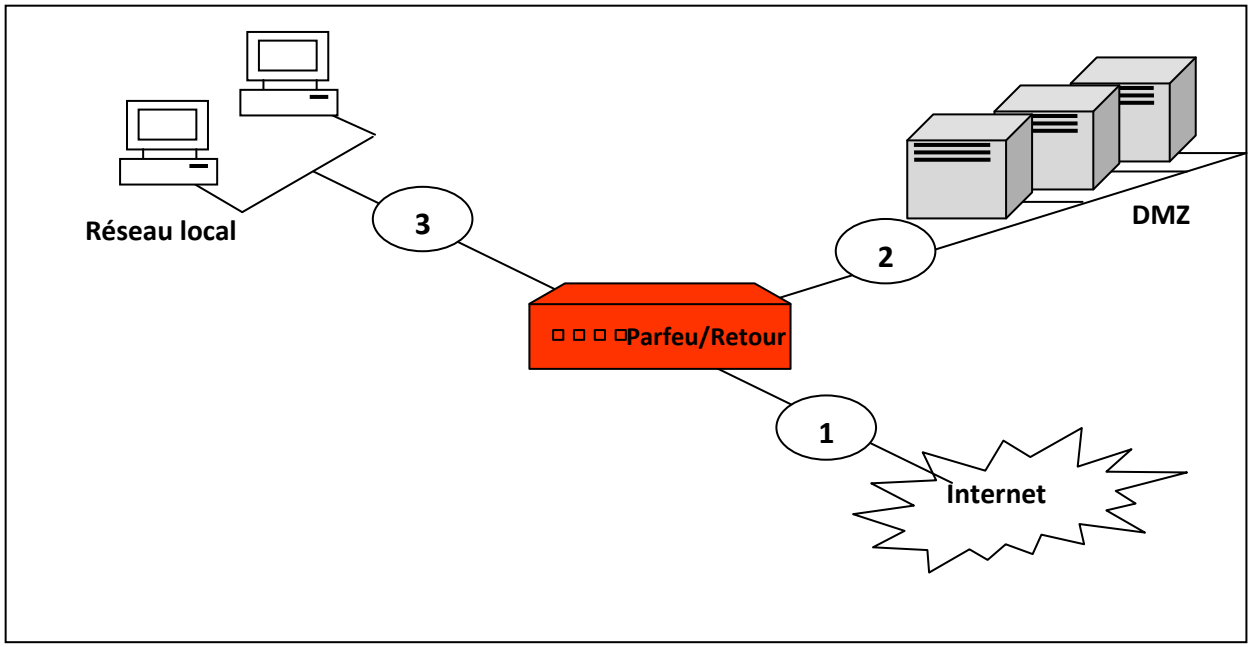


Figure1.4 : Emplacements des IDS

- **Position (1):** L'IDS Sur cette position sert à détecter l'ensemble des attaques frontales, provenant de l'extérieur, vers le firewall. Dans ce cas beaucoup d'alertes seront remontées ce qui rendra les logs difficilement consultables.
- **Position (2):** L'IDS placé sur la DMZ[4], utilisé pour détecter les attaques qui n'ont pas été filtrées par le firewall et qui relèvent d'un certain niveau de compétence. Les logs seront ici plus clairs à consulter puisque les attaques bénins ne seront pas recensées.
- **Position (3):** L'IDS dans cette position a pour objectif de rendre compte des attaques internes, provenant du réseau local de l'entreprise. Il peut être judicieux d'en placer un à cet endroit étant donné le fait que 80% des attaques proviennent de l'intérieur. De plus, si des trojans ont contaminé le parc informatique (navigation peu méfiante sur internet) ils pourront être ici facilement identifiés pour être ensuite éradiqués.

5- Classification des IDS :

Il existe plusieurs classifications des systèmes de détection des intrusions, nous avons choisi le modèle apparu dans la Figure 1.5 :

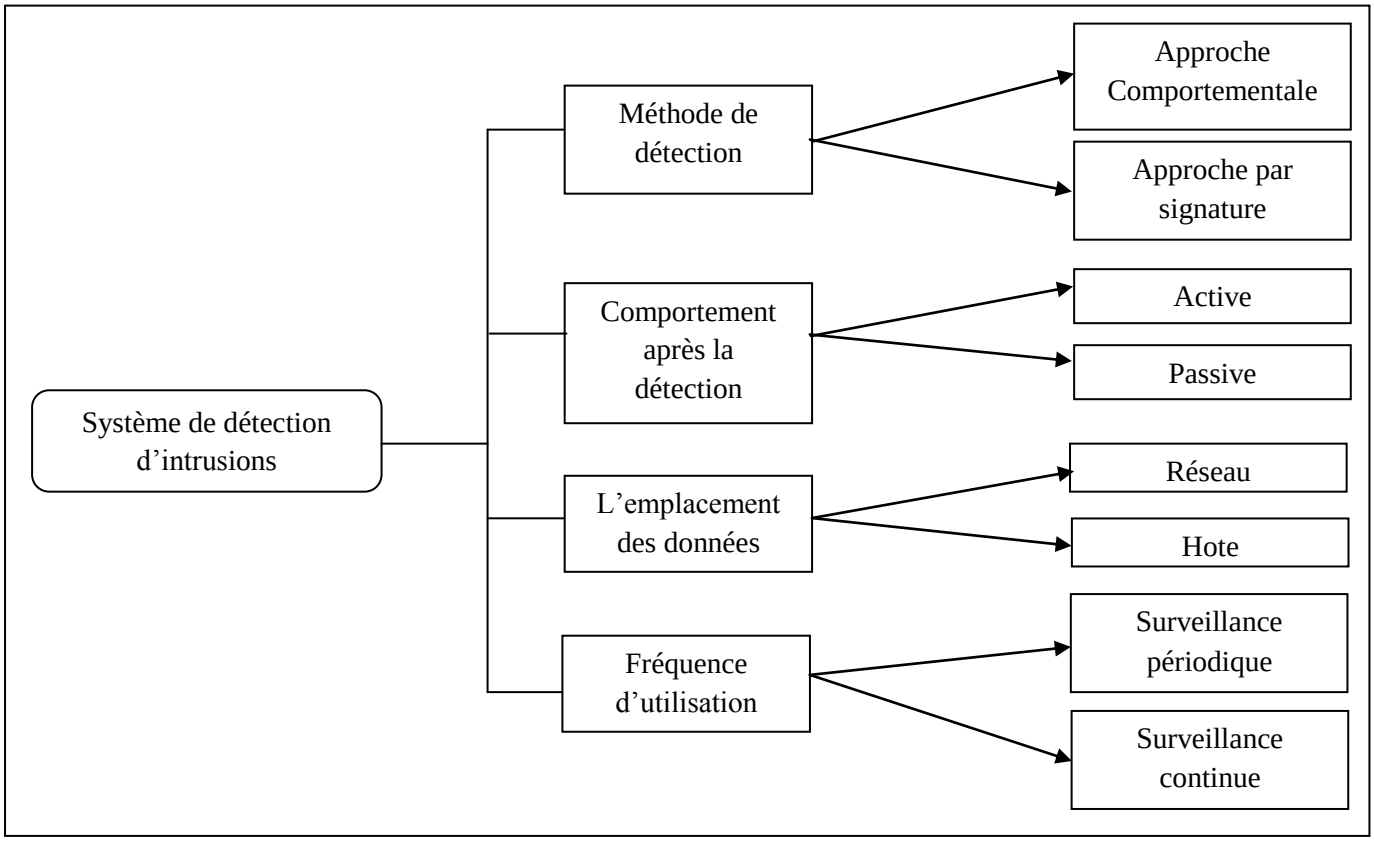


Figure1.5 : Classification d'IDS [9]

4 DMZ est un terme informatique désigne (zone démilitarisée), est 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 comme les serveurs.

5.1-Méthodes de détection: Comme le montre la Figure 1.6, il existe deux techniques principales de détection : par signatures (signature-based détection ou mésuse détection) ou par comportements (anomaly detection).

L'approche par signatures consiste à détecter des attaques en vérifiant si les observations correspondent à des attaques connues, tandis que l'approche par comportements (ou par détection d'anomalies) consiste à détecter une attaque en vérifiant que les observations ne correspondent pas à des comportements légitimes de référence. Certains IDS combinent les deux approches afin d'obtenir de meilleurs résultats.

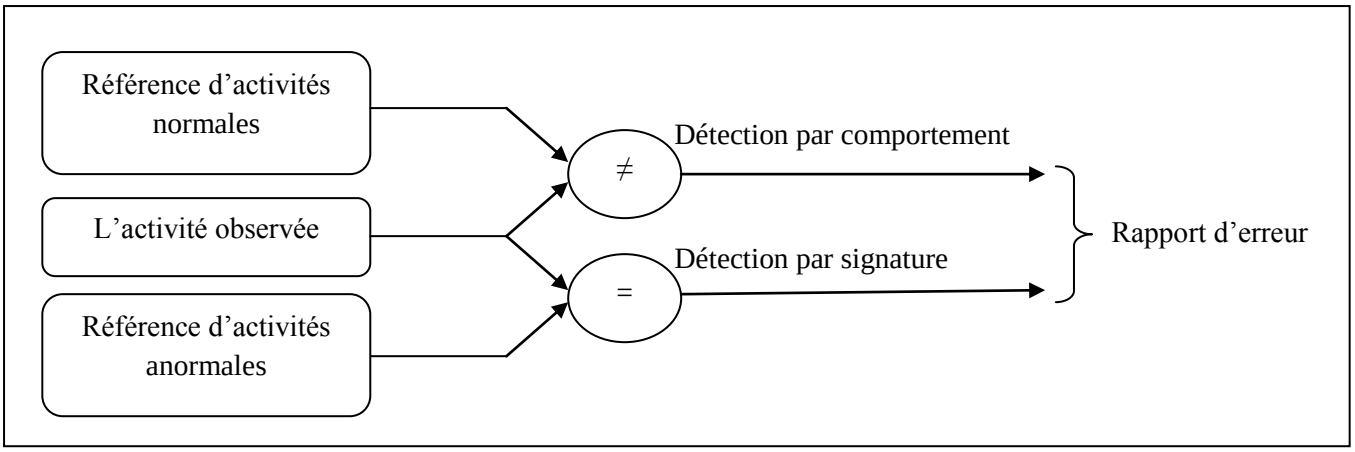


Figure 1.6 : Techniques de détection d'intrusions [6]

5.1.1-Approche 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. 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.

5.1.2-Approche par signature : Le concept de bibliothèque de signatures d'attaque est l'approche la plus basique et la plus ancienne. Cette approche consiste à rechercher dans l'activité de l'élément surveillé les

empreintes (ou signatures) d'attaques connues. Cette démarche appliquée à la détection d'intrusion, est très similaire à celle des outils antivirus et présente les mêmes inconvénients que celle-ci. Il est aisé de comprendre que ce type d'IDS est purement réactif ; il ne peut détecter que les attaques dont il possède la signature. De ce fait, il nécessite des mises à jour quotidiennes. De plus, ce système de détection est aussi bon qu'il est la base de signature. Si les signatures sont erronées ou incorrectement conçues, l'ensemble du système est inefficace. C'est pourquoi ces systèmes sont souvent contournés par les pirates qui utilisent des techniques dites " d'évasion " qui consistent à maquiller les attaques utilisées. Ces techniques de maquillage tendent à faire varier les signatures des attaques qui ainsi ne sont plus reconnues par l'IDS. Ce modèle est par contre très aisé à implémenter et à optimiser. Il permet la séparation du moteur logiciel de la base de signature qui peut ainsi être mise à jour indépendamment. Il permet également une classification relativement facile de la criticité des attaques signalées,

Les avantages et les Inconvénients des deux approches (comportementale, par signature) sont montrés dans le Tableau 1.1:

L'approche	Avantages	Inconvénients
comportementale	Détection d'intrusion inconnue possible	• Choix délicat des mesures à retenir pour un système cible donné. • Pour un utilisateur au comportement erratique, toute activité est « normale ». • En cas de profonde modification de l'environnement du système cible, déclenchement d'un flot ininterrompu d'alarmes (faux positifs). • L'utilisateur pouvant changer lentement de comportement dans le but d'habituer le système a un comportement intrusif (faux négatifs)
par signature	Prise en compte des comportements exacts des attaques potentiels.	• Base de règles délicates à construire. • Seules les attaques contenues dans la base sont détectés.

Tableau 1.1 : Comparaison entre les deux approches (comportementale et par signature)

5.2-Comportement après la détection : Ils existent deux types d'IDS ; actifs et passifs :

5.2.1-IDS à réponse passive : La réponse passive d'un IDS consiste à enregistrer les intrusions détectées dans un fichier de log qui sera analysé par le responsable de sécurité ou générer des alarmes, envoi d'un e-mail à un ou plusieurs utilisateurs, etc. Ceci permet de remédier aux failles de sécurité pour empêcher

les attaques enregistrées de se reproduire, mais elle n'empêche pas directement une attaque de se produire.

5.2.2-IDS à réponse active : La réponse active au contraire a pour but de stopper une attaque au moment de sa détection sans attendre l'intervention humaine, Pour cela on dispose de deux techniques : la reconfiguration du firewall et l'interruption d'une connexion TCP, La reconfiguration du firewall permet de bloquer le trafic malveillant au niveau du firewall, en fermant le port utilisé ou en interdisant l'adresse de l'attaquant. Cette fonctionnalité dépend du modèle de firewall utilisé, tous les modèles ne permettant pas la reconfiguration par un IDS. De plus, cette reconfiguration ne peut se faire qu'en fonction des capacités du firewall.

L'IDS peut également interrompre une session établie entre un attaquant et sa machine cible, de façon à empêcher le transfert de données ou la modification du système attaqué **[10]**.

5.3-Emplacement de données : Il existe des IDS qui surveillent l'état de la sécurité au niveau du réseau par la capture et l'analyse des paquets qui circulent à travers le réseaux (NIDS : Network Intrusion Détection System) , et d'autres surveillent l'état de la sécurité au niveau des hôtes et analysent les informations produites par le système d'exploitation ou par des applications installées dans les machines locales (HIDS :Host Intrusion Détection System), quelques IDS hybrides utilisent les NIDS et HIDS pour avoir des alertes plus pertinentes.**[11]**

5.3.1-Systèmes de détection d'intrusion réseaux (NIDS) :

L'IDS réseau ou (NIDS : Network Intrusion Détection System) surveille le trafic réseau. Il se place sur un segment réseau et "écoute" le trafic. Ce trafic sera ensuite analysé afin de détecter les signatures d'attaques ou les différences avec le fonctionnement de référence,

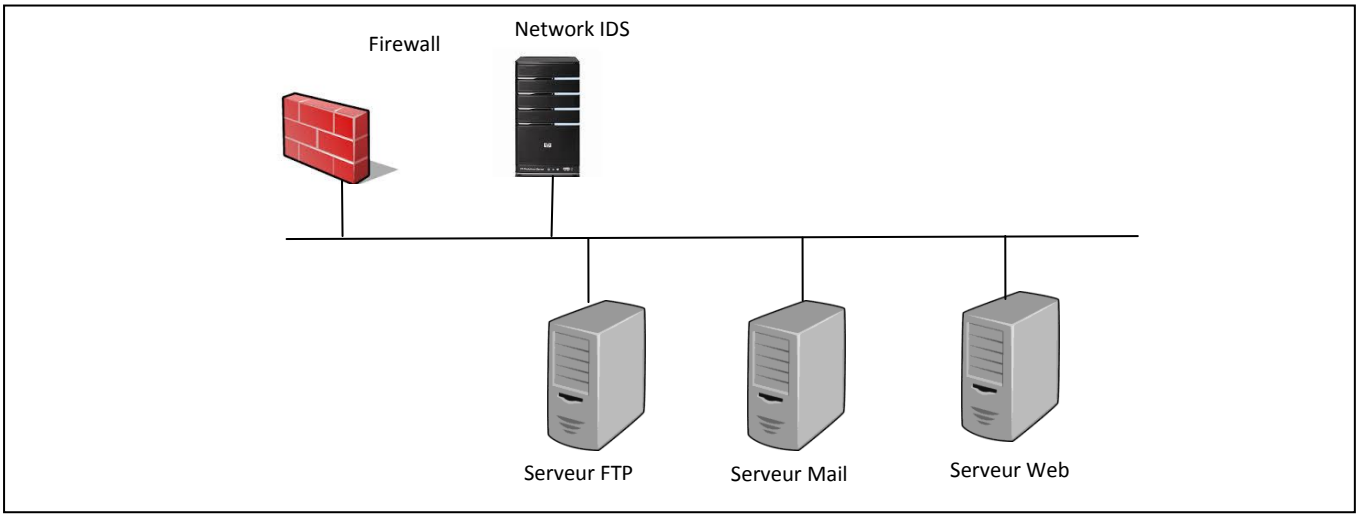


Figure 1.7 : Exemple d'un IDS dans un réseau (NIDS) [11]

Les avantages des NIDS :

- Le NIDS peut surveiller un grand réseau (un grand nombre d'hôte).
- Le déploiement de NIDS a peu d'impact sur un réseau existant. Les NIDS sont habituellement des dispositifs passifs qui écoutent sur un fil de réseau sans interférer l'opération normale de ce dernier. Ainsi, il est habituellement facile de monter en rattrapage un réseau pour inclure IDS avec l'effort minimal.
- NIDS peut être très sûr contre l'attaque et être même caché à beaucoup d'attaquants.

Les inconvénients des NIDS :

- 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.
- Plusieurs avantages de NIDS ne peuvent pas être appliqués pour les commutateurs modernes. La plupart des commutateurs ne fournissent pas des surveillances universelles des ports et limitent la gamme de surveillance de NIDS. Même lorsque les commutateurs fournissent de tels ports de surveillance, souvent le port simple ne peut pas refléter tout le trafic traversant le commutateur.
- NIDS ne peuvent pas analyser des informations chiffrées (dans le cas d'utilisation des VPN).
- La plupart de NIDS ne peuvent pas indiquer si une attaque est réussie ou non. Il reconnaît seulement qu'une attaque est initialisée. C'est-à-dire qu'après le NIDS détecte une attaque, l'administrateur doit examiner manuellement chaque host s'il a été en effet pénétré.
- Quelques NIDS provoquent des paquets en fragments. Ces paquets mal formés font devenir l'IDS instable.

5.3.2-Systèmes de détection d'intrusion sur hôte (HIDS) :

HIDS : Host Intrusion Détection System ou L'IDS Système : accomplir le travail de surveillance du trafic sur une machine locale par l'analyse des journaux, les appels systèmes, analyse de la base de registre et des logs en provenance de firewalls hétérogènes et vérifie l'intégrité des systèmes de fichiers. Le principe de fonctionnement des HIDS dépend du système sur lequel ils sont installés. L'intégrité des systèmes est alors vérifiée périodiquement et des alertes peuvent être levées par nature.

HIDS peut aussi observer les paquets réseaux de l'hôte (de la machine locale) pour la découverte des signaux d'intrusions (Déni de Services, Backdoors, chevaux de Troie, etc.).

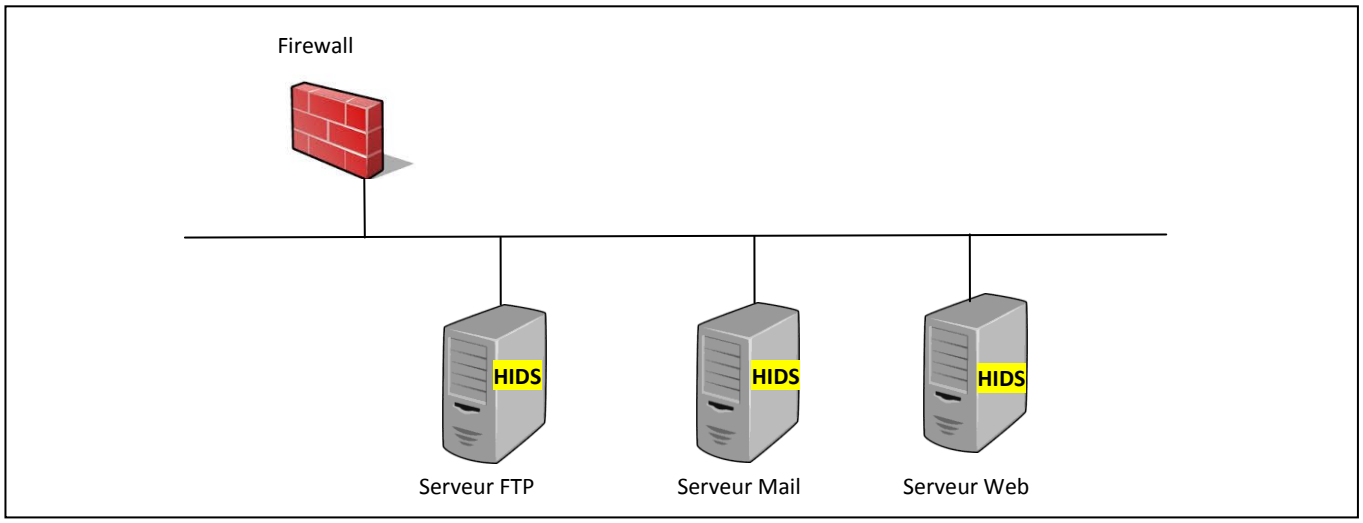


Figure 1.8 : Exemple d'un HIDS (L'IDS –Niveau Système) [11]

Les avantages des HIDS :

- Pouvoir surveiller des événements locaux jusqu'au host, détecter des attaques qui ne sont pas vues par NIDS.
- Analyse des flux cryptés (ce que ne peut réaliser un NIDS).
- lorsque les sources des informations de host-based sont générées avant l'encrypte des données ou après le décrypte des données au host de la destination.
- Les HIDS peuvent détecter le cheval de Troie ou les autres attaques relatives à la brèche intégrité de logiciel.

Les inconvénients des HIDS :

- HIDS est difficile à gérer, et des informations doivent être configurées et gérées pour chaque host surveillé.
- Puisque au moins des sources de l'information pour HIDS se résident sur l'host de la destination par les attaques, l'IDS peut être attaqué et neutralisé comme une partie de l'attaque.
- HIDS n'est pas bon pour la surveillance qui s'adresse au réseau entier parce que le HIDS ne voit que les paquets du réseau reçus par ses hosts.
- HIDS peut être neutralisé par certaine attaque de DoS.

5.4-Fréquence d'utilisation:

La dernière caractéristique des systèmes de détection d'intrusions est leur fréquence de surveillance: **périodique** ou **continue**.

5.4.1-Surveillance périodique : Dans ce cas les systèmes de détection d'intrusions analysent périodiquement les sources de données à la recherche d'une éventuelle intrusion ou anomalie passée. Cela peut être suffisant dans des contextes peu sensibles (on fera alors une analyse journalière, par exemple).

5.4.2-Surveillance continue : La plupart des systèmes de détection d'intrusions récents effectuent leur analyse des données sur la machine locale ou des paquets réseau de manière continue afin de proposer une détection en quasi temps-réel. Cela est nécessaire dans des contextes sensibles (confidentialité). C'est toutefois un processus coûteux en temps de calcul car il faut analyser à la volée tout ce qui se passe sur le système.

6- Les avantages d'utilisation des IDS :

- **Déjouer les attaques attendues sur le réseau :** Les IDS protègent les systèmes contre les attaques réseaux par : détection de porte dérobée, détection d'usurpation d'adresse IP, DoS, les vers, les chevaux de Troie, virus, Botnet, rootkit, Spyware, et autres menaces qui pourraient nuire au réseau, Les IDS actifs prennent des mesures automatiques contre les menaces de sécurité et les risques auxquels font face,
- **Avertis Administrateur réseau d'alerte pour les événements de sécurité potentiels :** la fonction de base des systèmes de détection d'intrusions est de générer des avertissements là où existent des menaces externes, internes ou de violations de la politique de sécurité réseau, et aussi de fournir à l'administrateur des informations détaillées sur le mouvement des données au sein du réseau.
- **Gagnez du temps:** L'utilisation des IDS fournit beaucoup de temps et d'effort pour connaitre de ce qui se passe dans le réseau, peut aussi tourner en permanence sans superviseur humain.
- **Contrôle des Programmes utilisés par les employés pour surveiller l'Internet :** IDS peut aider à découvrir les programmes qui traitent de l'internet, cela permet de mieux contrôler et de protéger le réseau,
- **Avoir la confiance des clients :** Les IDS aident les organisations de protéger les données de ses clients contre le vol et la violation de la sécurité, Cela permet d'avoir la confiance des clients et partenaires et garder une bonne réputation sur l'organisation.
- **Économisez de l'argent :** Grace aux IDS les organisations peuvent déterminer les mouvements suspects dans le réseau et signaler les responsables pour prendre des mesures proactives en protéger le réseau et gagner l'argent qui sera dépensé si la violation de la sécurité est arrivé dans le réseau ou si le vol de renseignements personnels a eu lieu.

7- Les limites actuelles de la détection d'intrusions:

- Les systèmes de détection d'intrusions actuels peuvent être mis en défaut, soit parce qu'ils sont incapables de détecter certains types d'attaques, soit parce qu'ils sont eux-mêmes attaquables.
- Les systèmes de détection d'intrusions actuels sont trop fermés, ce qui limite, les possibilités de comparaison de performances, et de coopération et de rendre difficile d'établir des standards d'interopérabilité entre outils pour résoudre ce problème.
- L'inadéquation entre les preuves exigées par les tribunaux et celles fournies par les outils de détection d'intrusions, cela nécessite des travaux supplémentaires pour extraire des preuves (des fichiers de logs du système, une partie du trafic réseau capturé durant l'attaque, des adresses IP incriminées et divers autres fichiers) lorsque des poursuites en justice sont envisagées, Le problème qui se pose est comment prouver que tout cela n'a pas été altéré.
- Les systèmes de détection d'intrusions génèrent trop de faux positifs.

8-Conclusion:

Dans ce chapitre nous avons abordé différentes notions de la sécurité informatique, on a expliqué les attaques informatiques, leurs classifications et les mécanismes utilisés dans la prévention contre ces attaques, parmi ces mécanisme on a détaillé les systèmes de détection des intrusions vu que c'est notre objectif dans ce mémoire, qui jouent un rôle complémentaire aux mécanismes de sécurité traditionnels.

Nous avons présenté aussi le principe de fonctionnement des IDS ainsi leurs classifications selon différents critères avec leurs avantages et inconvénients, parmi les critères de classification abordés la méthode de détection qui divise les IDS en deux types, les IDS comportementaux et à base de signatures, le principe de ce dernier consiste à rechercher dans l'activité de l'élément surveillé les empreintes (ou signatures) d'attaques connues d'une façon similaire à celle des antivirus, ce type d'IDS est simple d'implémenter mais il présente des difficultés comme la nécessité de mettre à jour d'une façon quotidienne la base de signature, en plus seuls les attaques contenues dans la base sont détectables, cela peut baissé l'efficacité des systèmes de détection des intrusions à base des signature surtout avec l'énorme développement des attaques réseaux d'aujourd'hui, et pour que la détection d'intrusions soit fiable nous avons basé dans notre projet(Détection d'intrusions à base de réseaux de neurones et algorithmes génétiques) sur l'approche comportementale qui rend la détection des intrusions inconnues possible et vise à classifier les enregistrements réseau en deux catégories(normal/attaque) en s'appuyant sur les réseaux de neurones et les algorithmes génétiques, qui sont les objets des prochaines chapitres.

Chapitre 2 :

Classification et réseaux de neurones

Introduction :

La classification est une technique très utilisé dans les travaux de fouille de données, la recherche d'information, reconnaissances des formes, le diagnostic médical, apprentissage automatique, l'aide à la décision et dans plusieurs domaines de recherche de l'intelligence artificielle.

Dans le domaine de la sécurité informatique, la classification est souvent utilisé pour classifier le trafic réseau ou les données système en deux catégories (légale ou illégale) pour le but d'aide à la décision, Cela est réalisé en pratique par des IDS qui utilisent des méthodes de classification pour modéliser d'une manière efficace le comportement des utilisateurs avec un nombre minimum de fausses alertes, en se basant sur différentes méthodes de l'intelligence artificielle telle que les réseaux de neurones artificiels.

Dans la première partie de ce chapitre nous nous focalisons sur la classification, les définitions et les concepts, leurs phases, types de classement (supervisée et non supervisée) et les méthodes de classement les plus utilisées de chaque types. Dans la deuxième partie, nous présentons le réseau de neurones, son historique, ses définitions, ses techniques d'apprentissage et ses architectures, nous passons ensuite à présenter en détaille le perceptron multicouches, et nous terminons par les domaines d'utilisation, les avantages et les inconvénients de ces réseaux.

I- Classification:

1-Concepts et Définitions:

La classification est le processus, qui permet de grouper des objets (observations ou individus) dans des classes (clusters) de manière à ce que les objets appartenant à la même classe soient plus similaires entre eux qu'aux objets appartenant aux autres classes. Le calcul de la proximité entre objets se fait sur une série de variables mesurées sur tous les objets. **[1]**

La classification connait une large utilisation dans plusieurs domaines notamment de l'intelligence artificielle comme l'analyse financière (prévision d'évolution de marchés), Marketing (établir un profil client, mailing), Banque (attribution de prêts), Médecine (aide au diagnostic), Télécom (détection de fraudes). Biométrie, Robotique, Reconnaissance de forme (OCR, Transcription de la parole, Compréhension/Dialogue), Recherche d'information (moteur internet, moteur multimédia) ou enfin la détection d'intrusions dans des réseaux informatiques qui est le sujet de notre projet de fin d'études.

2-L'architecture typique d'une application basée sur la classification :

Comme la classification est un problème central en reconnaissance des formes, l'application de cette dernière lors des travaux de développement d'un outil de classification dans n'importe quel domaine, doit être anticipé par d'autres phases d'extraction et d'analyse des informations à classifier ,ces

travaux se sont également attachés à étudier les phases montrés dans la figure (Figure2.1), dans notre travail (System de détection d'intrusions),notre objectif est de classifier l'ensemble des informations des connexions TCP/IP en deux sous-ensembles bien déterminés (**normal** ou **attaque**).

La Figure 2.1décrit le cursus parcouru par l'information à classifier.

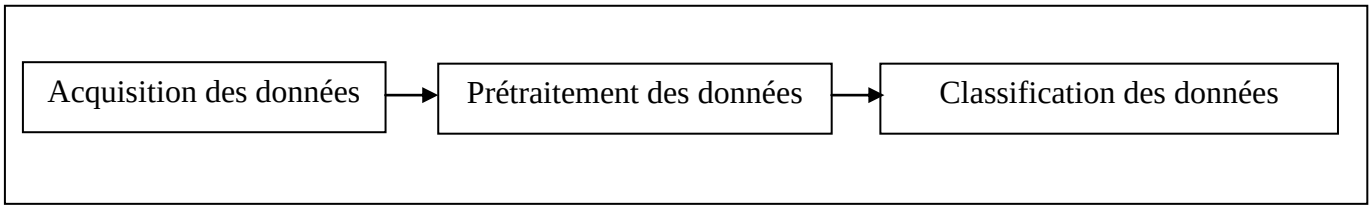


Figure 2.1 : Parcours de l'information à classifier [14]

- **Acquisition des données :** D'une manière générale, il s'agit à ce niveau de mettre en place l'ensemble d'instrumentation (capteurs, matériel d'acquisition, etc.) de façons à reproduire le phénomène observé le plus fidèlement possible. C'est l'opération de transformation de l'information à traiter en signaux numériques manipulables par ordinateurs ou bien la numérisation de l'information. Dans notre cas, il s'agit de placer des sondes (IDS) pour écouter le trafic réseau.
- **Prétraitement des données :** Cette phase correspond au filtrage des informations en ne conservant que ce qui est pertinent dans le contexte d'étude. Dans notre cas, il s'agit de l'enregistrement des connexions TCP/IP.
- **Classification des données :** Elle correspond à l'étape de décision et pour cela plusieurs méthodes se présentent pour la résolution. Pour la classification des connexions TCP/IP nous appliquerons un algorithme basé sur les réseaux de neurones.

3-Taxonomie de classification :

Un grand nombre de méthodes est établi pour résoudre à peu près tous les problèmes de classification, cependant du fait que certaines de ces approches partagent des caractéristiques communes, soit dans la façon d'appréhender le problème (apprentissage supervisé ou non), soit dans la nature de la sortie réalisée (groupes disjoints ou classification flou) , la Figure 2.2 montre le regroupement de ces méthodes sous la forme d'une hiérarchie (taxonomie). **[15]**

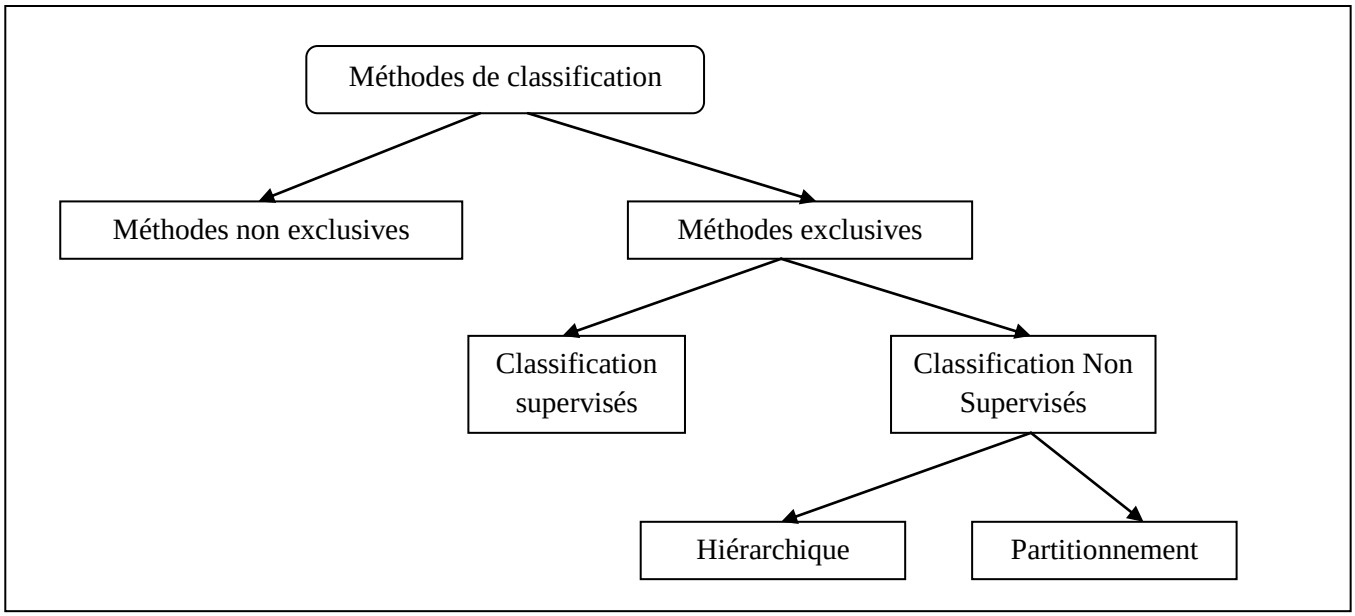


Figure 2.2 : Les méthodes de classification [15]

Nous donnons dans ce qui suit une rapide description des méthodes décrites dans la figure ci-dessus :

3-1-Classification exclusive : Dans la classification exclusive un objet ne peut être qu'à une seule classe dans la partition finale.

3-1-1Classification non supervisée : La classification non supervisée (clustering), consiste à affecter les individus considérés similaires au même groupe sans en connaître au préalable la structure. Les classes ne sont pas connues à l'avance, et les exemples disponibles sont non étiquetés. Le but est donc de constituer des classes, parmi les méthodes de classification non supervisé les plus utilisées, la méthode K-Means (centres mobiles) et la méthode hiérarchique. [**17**]

3-1-1-1- Classification ascendante hiérarchique (CAH) : [16]

La classification ascendante hiérarchique, a pour objectif de construire une suite de partitions emboitées des données en n classes, n-1 classes, ..., 1 classe. Que l'on visualise par exemple par un dendrogramme (Figure 2.3).

Principe :

- A l´étape initiale, les « n » individus constituent des classes à eux seuls.
- On calcule les distances deux à deux entre individus, et les deux individus les plus proches sont réunis en une classe.
- La distance entre cette nouvelle classe et les n-2 individus restants est ensuite calculée, et à nouveau les deux éléments (classes ou individus) les plus proches sont réunis.

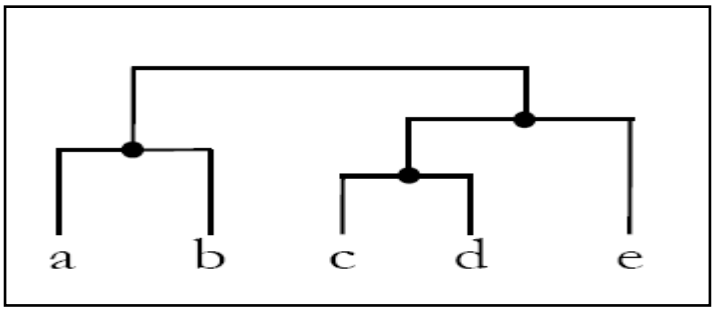


Figure2.3 : Exemple de dendrogramme [15]

3-1-1-2-Méthode des K-means (Centres mobiles) :

Cet algorithme fut longtemps utilisé sur les grands jeux de données en raison de sa rapidité. On s'intéresse tout d'abord à l'algorithme lui même, puis à ses propriétés.

Principe : On suppose qu'il existe K classes distinctes. On commence par designer K centres de classes μ_1, ..., μ_K parmi les individus. Ces centres peuvent être soit choisis par l'utilisateur pour leur "représentativité", soit désignés aléatoirement. On réalise ensuite itérativement les deux étapes suivantes :

– Pour chaque individu qui n'est pas un centre de classe, on regarde quel est le centre de classe le plus proche. On définit ainsi K classes C_1, ..., C_K, où C_i = {ensemble des points les plus proches du centre μ_i},

- Dans chaque nouvelle classe C_i, On définit le nouveau centre de classe μ_i comme étant le barycentre des points de C_i. **[16]**

L'algorithme s'arrête suivant un critère d'arrêt fixé par l'utilisateur qui peut être choisi parmi les suivants :

- Soit le nombre limite d'itérations est atteint, soit l'algorithme converge, c'est-a dire qu'entre deux itérations les classes formées restent les mêmes,
- Soit l'algorithme a "presque" converge, c'est-à-dire que l'inertie intra-classe ne s'améliore quasiment plus entre deux itérations.

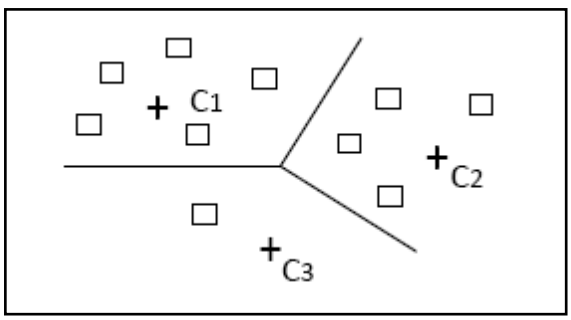


Figure 2. 4 : Exemple de partition obtenue par les centres mobiles [15]

3-1-2- Classification supervisée :

Dans ce type de classification, les classes sont connues à l'avance, et on dispose d'un ensemble d'objets déjà étiquetés (classés) servant l'ensemble d'apprentissage. Le problème est alors de générer un modèle capable d'associer un tout nouvel objet à sa classe la plus adaptée, en se servant des exemples déjà étiquetés (exemples d'apprentissages) , Parmi les méthodes de classification supervisée, nous citons la méthode de k plus proche voisins, la méthode d'arbre de décision.**[17]**

3-1-2-1- K plus proches voisins :

K plus proches voisins est la méthode de classification supervisée la plus simple. Son principe est résumé comme suit : pour un objet x à classer (prédire la classe d'un nouveau cas), on examine la distance de x à tous les échantillons (qui définissent toutes les classes), puis on sélectionne les K plus proches échantillons et on affecte x à la classe majoritaire parmi ces K échantillons.

3-1-2-2- Arbres de décision :

Les arbres de décision sont les méthodes les plus connues en classification. Le principe de cette méthode est de réaliser la classification d'un exemple par une suite de tests sur les attributs qui le décrivent. Concrètement, dans la représentation graphique d'un arbre où :

- Un nœud interne correspond à un test sur la valeur d'un attribut.
- Une branche part d'un nœud et correspond à une ou plusieurs valeurs de ce test.
- Une feuille est un nœud d'où ne part aucune branche et correspond à une classe.

Une règle de décision (de la forme si . . . alors . . .) est créée pour chaque chemin partant de la racine de l'arbre et parcourant les tests (en faisant des conjonctions) jusqu'à la feuille qui est l'étiquette de la classe

Il y a d'autre méthodes de la classification supervisée, tel que la méthode de **réseaux de neurones** et vu que nous nous basons sur ces réseaux pour réaliser notre modèle de détection d'intrusions, ces derniers vont être étudiés en détails dans les sections suivantes de ce chapitre. **[20]**

3-2-Classification non exclusive: Chaque objet est associé à une de densité de probabilité qui indique pour chacune des classes la probabilité que l'objet considéré y appartienne (la classification floue).

4-La classification et réseaux de neurones:

Vu que la majorité des problèmes de classification sont non-linéaires, Les modèles linéaires sont incapables de résoudre de manière satisfaisante un problème de classification, ainsi cet inconvénient des modèles linéaires est important. Le réseau de neurones artificiel, aussi appelé « perceptron multicouches », permet de corriger cette situation, ce réseau sera abordé en détail dans le chapitre suivant.

II-Réseaux de neurones :

Les réseaux de neurones artificiels nous a permis de simuler d'une façon formelle le travail du cerveau humain, les scientifiques ont découvert presque la façon dont le travail du cerveau humain en termes d'évolutivité et la portabilité de la mémoire d'apprentissage et la capacité de distinguer les objets et la capacité de prendre des décisions et comme nous le savons, le cerveau est constitué de milliards de neurones interconnectés entre eux d'une manière très complexe par les cellules neuronales, ce qui forme un énorme réseau de neurones associés les uns aux autres.

Cette corrélation entre les cellules nerveuses donne à ces dernières la capacité de stocker et fournir des informations, des images, audio et succession de signaux qui reçoivent à travers les différents neurones, les réseaux de neurones permettent également d'apprendre par la répétition et l'erreur.

1-Concepts et Définitions:

Historique :

L'histoire des réseaux de neurones artificiels revient au1943, où Mac Cullochet Pittsont proposé des neurones formels mimant les neurones biologiques et capables de mémoriser des fonctions booléennes simples.

Les réseaux de neurones artificiels réalisés à partir de ce type de neurones sont ainsi inspirés du système nerveux. Ils sont conçus pour reproduire certaines caractéristiques des mémoires biologiques par le fait qu'ils sont massivement parallèles, capables d'apprendre et de mémoriser l'information dans les connexions entres neurones, capables de traiter des informations incomplètes.

En 1949, Hebb a mis en évidence l'importance du couplage synaptique dans l'apprentissage par renforcement ou dégénérescence des liaisons inter-neuronales lors de l'interaction du cerveau avec le milieu extérieur.

Le premier modèle opérationnel est le perceptron simple inspiré du modèle visuel et capable d'apprentissage. Il a été proposé en 1958 par Rosenblatt.

Les limites du Perceptron monocouche du point de vue performance ont été montrées en 1969 par les mathématiciens Minsky et Papert.

Les travaux de Hopfield en 1982 ont montrés que des réseaux de neurones artificiels étaient capables de résoudre des problèmes d'optimisation et ceux de Kohonen (1982) ont montré qu'ils étaient capables des résoudre des tâches de classification et de reconnaissance. [**18]**

1-1. Neurone biologique :

Le neurone est une cellule composée d'un corps cellulaire et d'un noyau. Le corps cellulaire se ramifie pour former ce que l'on nomme les dendrites, celles-ci sont parfois assez nombreuses que l'on parle alors de chevelure dendritique ou d'arborisation dendritique. C'est par les dendrites que l'information est acheminée de l'extérieur vers le soma (corps du neurone).L'information traitée par le

neurone est propagée ensuite le long de l'axone (unique) pour être transmise aux autres neurones. La transmission entre deux neurones n'est pas directe. En fait, il existe un espace inter-cellulaire de quelques dizaines d'Angströms (10^{-9} m) entre l'axone du neurone afférent et les dendrites (on dit une dendrite) du neurone différent. La jonction entre deux neurones est appelée la synapse (Figure2.5) **[19]**

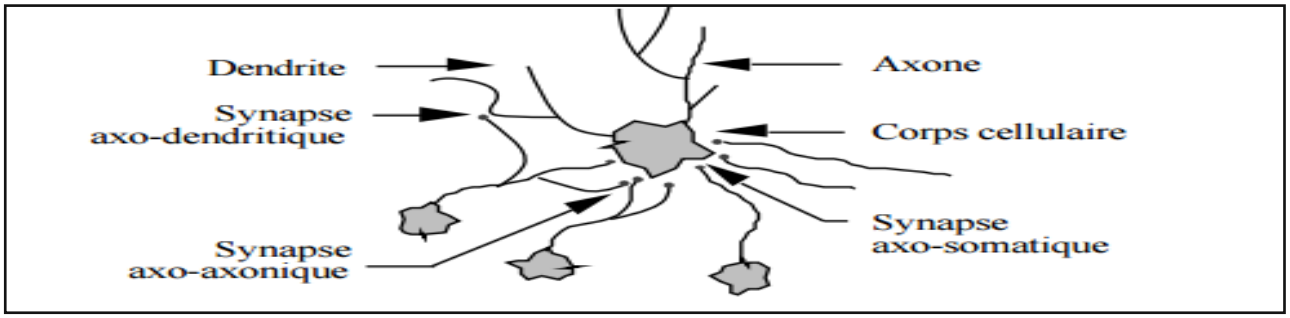


Figure 2.5 : Structure d'un neurone biologique [19]

1-2. Neurone artificiel :

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.

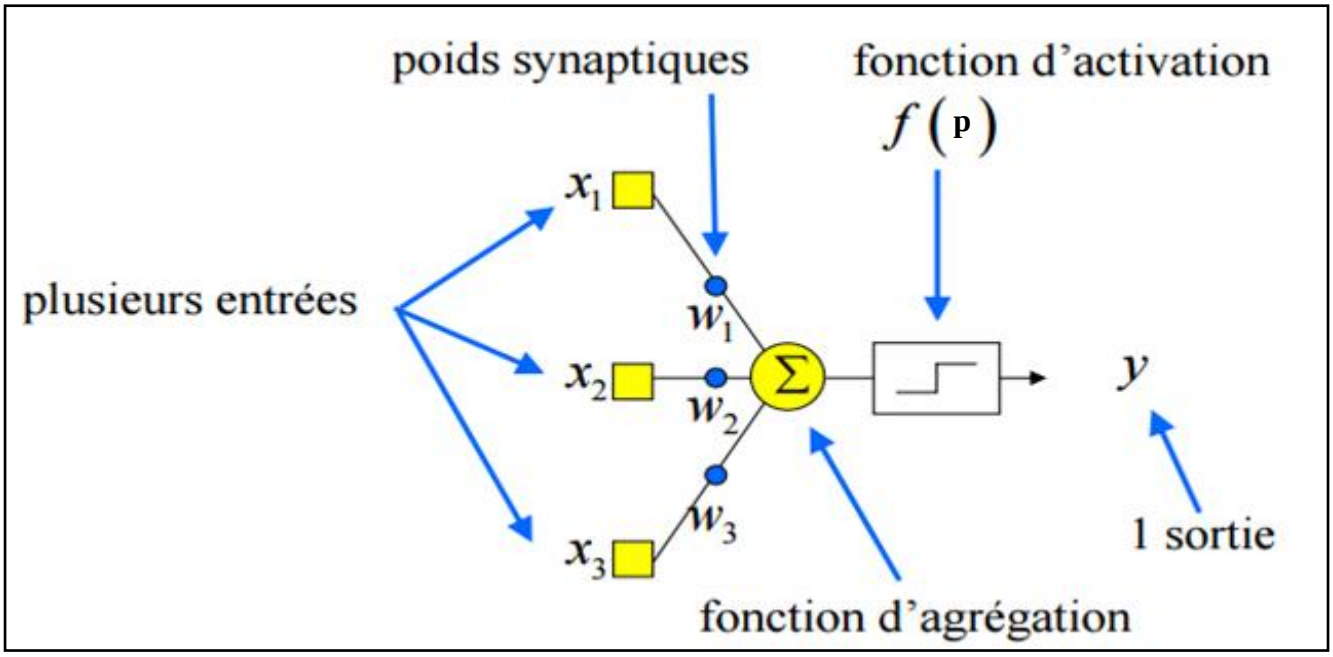


Figure 2.6 : Structure d'un neurone formel

1-3-Correspondance entre neurone biologique et neurone artificiel :

La structure d'un neurone artificiel est en fait inspirée de la structure des neurones biologiques. Les principales structures biologiques des neurones ont toutes leurs équivalents artificiels, ceci ayant pour but

de reproduire leur fonctionnement de la meilleure façon possible (d'une manière logique, simple et facilement représentable sur informatique).

Neurone biologique	Neurone artificiel
Synapses	Poids de connexions
Axones	Signal de sortie
Dentride	Signal d'entrée
Somma	Fonction d'activation

Tableau 2.1 : Transition entre le neurone biologique et le neurone formel

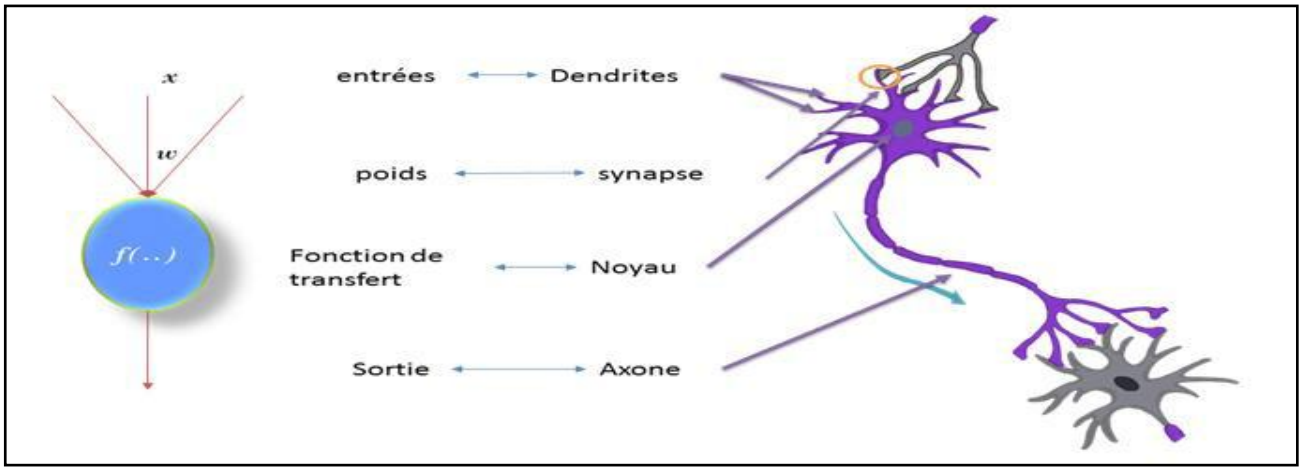


Figure 2.7 : Correspondance entre neurone artificiel et neurone biologique.

2- Comportement de neurone artificiel:

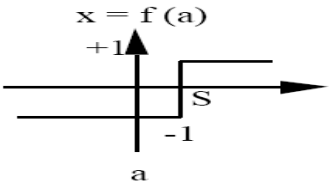
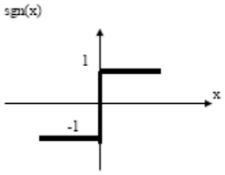
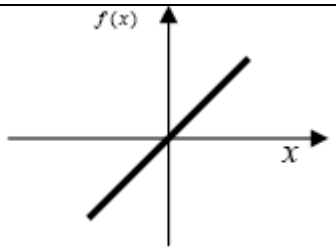
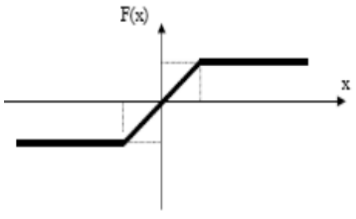
- **Entrées du neurone « X » :** Elles proviennent soit d'autres éléments ''processeurs'' (neurones), soit de l'environnement.
- **Le poids « W » (coefficient synaptique) :** Est une valeur numérique associée à une connexion entre deux unités (neurones) qui reflète la force de relation (connexion) entre ces deux unités *i* et *j*, et il est noté par W_i.
- **La fonction d'agrégation (combinaison) « P »:** Elle combine les entrées et les poids en calculant l'influence de chaque entrée en tenant en compte de son poids. Cette influence est calculée via la formule suivante: $P = \sum W_i X_i$, Où W_i est le poids de la connexion à l'entrée i, X_i: est le signal de l'entrée i.
- **La fonction de transfert (d'activation):**

La fonction d'activation (la fonction de transfert) joue un rôle très important dans le comportement du neurone. Elle retourne une valeur représentative de l'activation du neurone, cette fonction a comme

paramètre la somme pondérée des entrées ainsi que le seuil d'activation. Elle calcule la valeur de sortie à partir du résultat de la fonction de combinaison : S = F(P).

Où : **S** : est la valeur de sortie, **F** : est la fonction de transfert.

La nature de cette fonction diffère selon le réseau, les plus courantes sont présentées sur le Tableau 2.2, avec leurs équations mathématiques. Sachant que la différence avec les neurones biologiques est que l'état de ces derniers est binaire, par contre la plupart des fonctions de transfert sont continuées et offrant une infinité de valeurs comprises dans l'intervalle [0, +1] ou [-1, +1]. [**20**]

Catégorie	**La fonction**	**La Formule mathématique**	**Le graphe**
seuil	**Fonction de Heaviside**	**X=** $\begin{cases} \textbf{1 Si a}\geq\textbf{0} \\ \textbf{0 Si a}<\textbf{0} \end{cases}$	
	Fonction de signe	$S = \begin{cases} 1 \ si \ \ a \geq 0 \\ -1 \ si \ \ a < 0 \end{cases}$ Tel que : S = seuil	
Linéaire	**Identité**	f (x) = x	
	Fonction linéaire à seuil ou multi-seuil	$F(x) = \begin{cases} x & x \in [u, v] \\ v & si \ x \geq v \\ u & si \ x \leq u \end{cases}$	

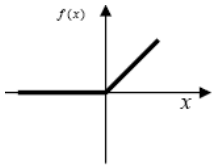
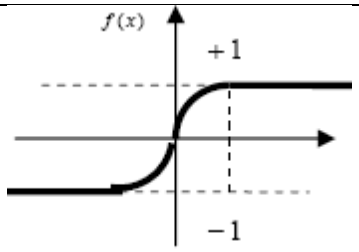
	Fonction Linéaire positif	$f(x)\begin{cases}0 & si\ x \prec 0\\ x & si\ x \geq 0\end{cases}$	
Non Linéaire	**Logistique (sigmoïde)**	$f(x)=\frac{1}{1+e^{-x}}$	

Tableau 2.2 : Les fonctions de transfert les plus utilisées.

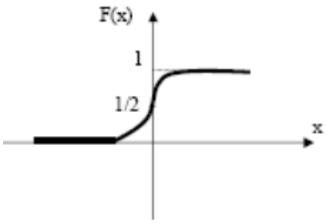
La fonction sigmoïde(dérivable):

Est la plus utilisée car elle introduit de la non-linéarité, mais c'est aussi une fonction continue, différentiable, C'est la fonction que nous avons utilisé pour mettre en œuvre notre modèle, Une fonction sigmoïde est définie par : $f_{sig}(x)=\frac{1}{1+e^{-x}}$

Possède les propriétés importantes évoquées précédemment (elle n'est pas polynomiale et est indéfiniment continument dérivable). En outre, une propriété simple permet d'accélérer le calcul de sa dérivée, ce qui réduit le temps de calcul nécessaire à l'apprentissage d'un réseau de neurones. On a en effet :

$$\frac{d}{dx}f_{sig}(x) = f_{sig}(x)\ (1 - f_{sig}(x))$$

On peut donc calculer la dérivée de cette fonction en un point de façon très efficace à partir de sa valeur en ce point. De plus, la fonction sigmoïde est à valeurs dans l'intervalle [0 ; 1], ce qui permet d'interpréter la sortie du neurone comme une probabilité.



3-Les types d'apprentissage des réseaux de neurones :

L'apprentissage est sans doute la propriété la plus intéressante des réseaux neuronaux, elle ne concerne cependant pas tous les modèles, mais les plus utilisés.

L'apprentissage est une phase du développement d'un réseau de neurones durant laquelle le comportement du réseau est modifié jusqu'à l'obtention du comportement désiré, "C'est-à-dire un changement dans la valeur des poids qui relient les neurones d'une couche à l'autre" [7]. Soit le poids w_{ij}

reliant le neurone i à son entrée j. Au temps τ, un changement $\triangle w_{ij}(\tau)$ de poids peut s'exprimer simplement de la façon suivante:

$$\triangle w_{ij}(\tau) = w_{ij}(\tau+1) - w_{ij}(\tau)$$

Par conséquent, $w_{ij}(\tau+1) = w_{ij}(\tau) + \triangle w_{ij}(\tau)$, avec $w_{ij}(\tau+1)$ et $w_{ij}(\tau)$ représentent respectivement les nouvelles valeurs et les anciennes aussi du poids w_{ij} .

On distingue plusieurs types d'apprentissages, parmi lesquels on cite les trois principaux: non supervisé, par renforcement et supervisé.**[20] [21]**

3-1- L'apprentissage non supervisé :

Le réseau doit détecter des points communs aux exemples présentés, par la modification des poids, afin de fournir la même sortie pour des entrées aux caractéristiques proches.

L'apprentissage non supervisé est bien adapté à la modélisation des données complexes (images, sons, etc.), généralement des données symboliques, où l'on possède des règles moins précises qui gouverne le comportement du système à modélisé par les réseaux de neurones.

3-2-Apprentissage par renforcement :

Dans ce cas, bien que les sorties idéales ne soient pas connues directement, il y a un moyen quelconque de connaître si les sorties du RNA s'approchent ou s'éloignent du but visé. Ainsi, les poids sont ajustés de façons plus ou moins aléatoire et la modification est conservée si l'impact est positif ou rejetée sinon.

3-3- L'apprentissage supervisé :

Comme nous l'avons vu précédemment, un réseau de neurones non bouclé réalise une fonction algébrique entre ses entrées et ses sorties. Donc on peut effectuer à un tel réseau la tâche qui consiste à réaliser une fonction algébrique non linéaire, on fournit à ce réseau un couple (entrée, sortie) et on modifie les poids en fonction de l'erreur entre la sortie désirée et la sortie obtenue, (Figure 2.8).

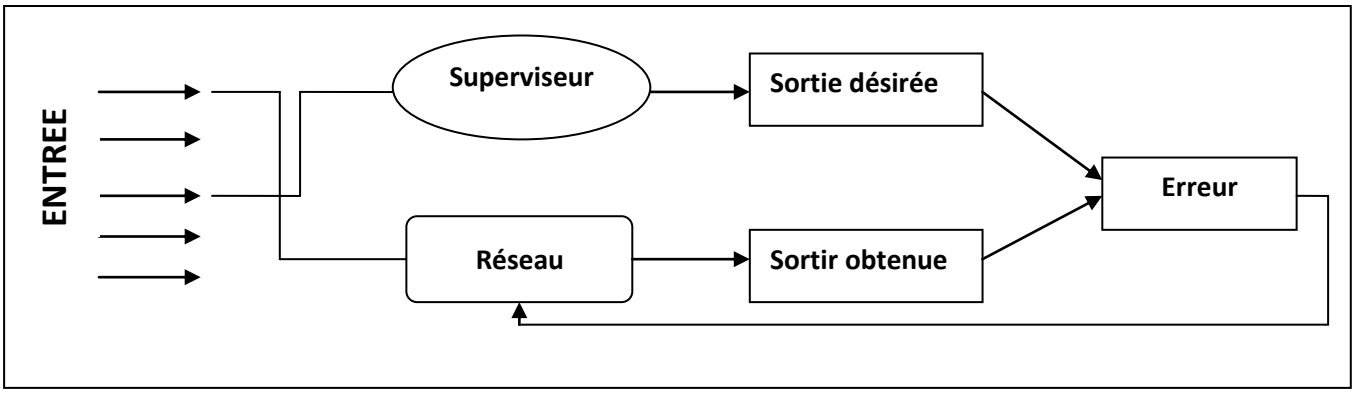


Figure 2.8: Apprentissage supervisé d'un réseau de neurones

On peut diviser la fonction algébrique réalisée par ce réseau en deux parties : fonction connue analytiquement, où le réseau réalise la tâche d'approximation et une fonction inconnue analytiquement, mais pour laquelle on dispose de valeurs, en nombre fini, si ces valeurs résultent de mesures effectuées sur un processus physique, chimique,... etc. Le réseau dans ce cas réalise une modification statique ou une régression**.**

Nous nous limitons, dans ce chapitre, à l'apprentissage supervisé et plus particulièrement à la modélisation statique. L'algorithme d'apprentissage utilisé dans notre travail : est la rétro propagation des erreurs car ce dernier est le mieux adapté à la modélisation statique pour le perceptron multicouche MLP.

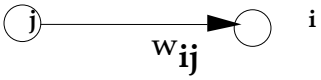
4-Règles d'apprentissage :

Il existe plusieurs types de règles d'apprentissage (modification de poids), Parmi lesquelles on trouve:

➢ **Correction d'erreur Loi de Widrow-Hoff (Règle de delta) :a_i** activation produite par le réseau, **d_i** réponse désirée par l'expert humain par exemple si la sortie est inférieure à la réponse désirée, il va falloir augmenter le poids de la connexion à condition bien sûr que l'unité j soit excitatrice (égale à 1). On est dans l'hypothèse d'unités booléennes {0,1}.

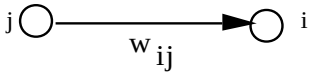
	ai = 0	ai = 1
di = 0	$\Delta W_{ij} = 0$	$\Delta W_{ij} = -R$
di = 1	$\Delta W_{ij} = R$	$\Delta W_{ij} = 0$

$$\Delta W_{ij}=R(d_i\_a_i)a_j$$



➢ **Loi de Grossberg:** On augmente les poids qui entrent sur l'unité gagnante **a_i** s'ils sont trop faibles, pour les rapprocher du vecteur d'entrée aj. C'est la règle d'apprentissage utilisée dans les cartes auto-organisatrices de Kohonen

$$\Delta W_{ij} = Ra_i\left(a_j - W_{ij}\right)$$



- **Apprentissage de Boltzmann:**

Les réseaux de Boltzmann sont des réseaux symétriques récurrents. Ils possèdent deux sous-groupes de cellules, le premier étant relié à l'environnement (cellules dites visibles) et le second ne l'étant pas (cellules dites cachées), les machines de Boltzmann opèrent en deux modes distincts :

Le mode figé («clamped»), dans ce cas les cellules visibles sont affectées par une valeur déterminée par l'environnement ;

Le mode libre évolution («free-running») dans lequel l'ensemble des cellules, qu'elles soient visibles ou cachées, peuvent changer d'état librement.

La règle d'apprentissage est de type stochastique, elle est dérivée de la théorie de l'information et des principes de la thermodynamique. L'objectif de cet apprentissage est d'ajuster les poids des connexions, de sorte que l'état des cellules visibles satisfasse une distribution probabiliste souhaitée. En accord avec la règle d'apprentissage de Boltzmann, les modifications se font par:$\Delta w_{ij}=h(\bar{r}_{ij}-r_{ij})$ où h est le taux d'apprentissage et$(\bar{r}_{ij})$, (resp. r_{ij}) sont les corrélations entre les états *i* et *j* lorsque le système est en mode figé (resp. libre-évolution), Les valeurs $\bar{r}_{ij}$ et r_{ij} sont classiquement estimées à l'aide de la méthode de Monte-Carlo[5] qui est extrêmement lente. **[17]**

- **Règle de Hebb :** ***«Si deux unités connectées sont simultanément actives, Alors le poids de la connexion qui les relie doit être renforcé »*****,** **[28]**, Selon la formule ci-dessous**, R** est une constante positive qui représente la force d'apprentissage (learning-rate).

	ai = -1	**ai = 1**
aj = -1	**Δ Wji = R**	**Δ Wji = -R**
aj = 1	**Δ Wji = -R**	**Δ Wji = R**

$\Delta W_{ji}=Ra_ia_j$

- **Apprentissage par compétition :**

A la différence de la règle de Hebb (dans laquelle plusieurs neurones peuvent être activés en sortie), cet apprentissage n'active qu'un seul neurone. Le principe de cet apprentissage est de regrouper les données en catégories. Les patrons similaires vont donc être rangés dans une même classe, en se basant sur les corrélations des données, et seront représentés par un seul neurone, on parle de «winner-take-all ».Dans un réseau à compétition simple, chaque neurone de sortie est connecté aux neurones de la couche

5La méthode de Monte-Carlo : C'est une méthode qui utilise des techniques probabilistes pour calculer des estimations. Le nom de cette méthode fait référence aux jeux de hasard pratiqués à Monte-Carlo, a été inventé en 1947.

d'entrée, aux autres cellules de la couche de sortie (connexions inhibitrices) et à elle-même (connexion excitatrice).

La sortie va donc dépendre de la compétition entre les connexions inhibitrices et excitatrices, Le résultat de la compétition est de choisir la cellule ayant la plus grande (ou la plus petite) entrée. A noter que seules les connexions du vainqueur sont mises à jour.

5- 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). **[22]**

5-1- Les réseaux de neurones non bouclés :

Un réseau de neurones non bouclé est un ensemble de neurones « connectés »entre eux, l'information circulant 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 (Figure 2.9).

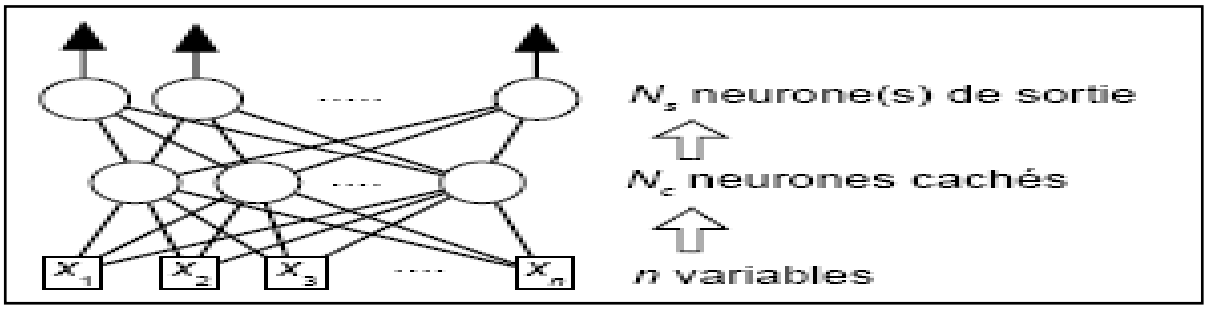


Figure 2.9: Architecture d'un RN non bouclé

5-2- 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 (Figure 2.10). **[22]**

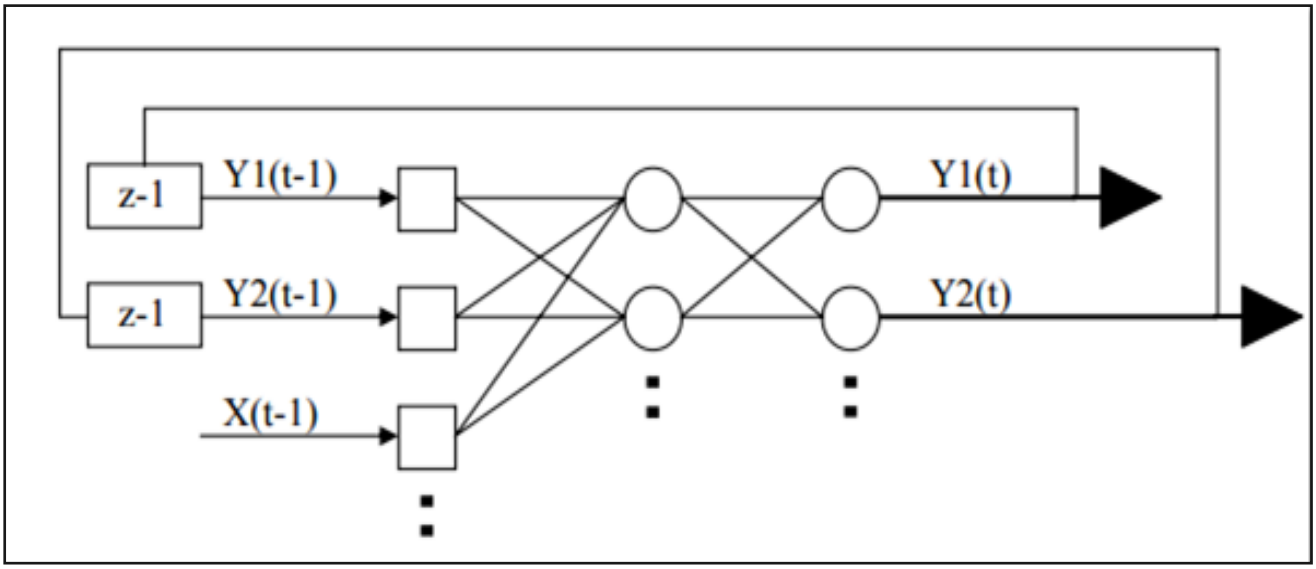


Figure 2.10 : Réseau de neurones bouclé

6-Quelques modèles de réseaux de neurones:

6-1-Perceptron simple:

Le perceptron est le premier modèle de réseau de neurones inventé en 1957 par Frank Rosenblatt[6]. Le but du perceptron est d'associer des formes en entrée à des réponses. Le perceptron se compose de deux couches : la couche d'entrée et la couche de sortie qui donne la réponse correspondant à la stimulation présente en entrée. Les cellules de la première couche répondent en oui/non. La réponse« oui » correspond à une valeur « 1 » et la réponse« non » correspond à une valeur « 0»à la sortie du neurone. Les cellules d'entrée sont reliées aux cellules de sortie grâce à des synapses d'intensité variable. L'apprentissage du perceptron s'effectue en modifiant l'intensité de ces synapses. Les cellules de sortie évaluent l'intensité de la stimulation en provenance des cellules de la couche d'entrée en effectuant la somme des intensités des cellules actives.

6**Frank Rosenblatt :** était un psychologue américain qui travaille sur l'intelligence artificielle

Le perceptron doit trouver l'ensemble des valeurs à donner aux synapses pour que les configurations d'entrée se traduisent par des réponses voulues. Pour cela, on utilise la règle d'apprentissage de Windrow-Hoff (Correction d'erreur).

Pour apprendre, le perceptron simple (Figure 2.11) doit savoir qu'il a commis une erreur, et doit connaitre la réponse qu'il aurait dû donner. De ce fait, on parle d'apprentissage supervise, La règle d'apprentissage est locale dans ce sens que chaque cellule de sortie apprend sans avoir besoin de connaitre la réponse des autres cellules. La cellule ne modifie l'intensité de ses synapses (apprend) que lorsqu'elle se trompe.

Marvin Lee Minsky[7] a montré qu'une forme toute simple (le XOR) ne peut être apprise par un neurone de type perceptron. Un neurone ne peut séparer que deux régions séparables par un hyper plan, avec plusieurs neurones, ça va déjà mieux mais il est vite clair qu'une seule couche de perceptron ne peut pas apprendre des figures complexes. **[23]**

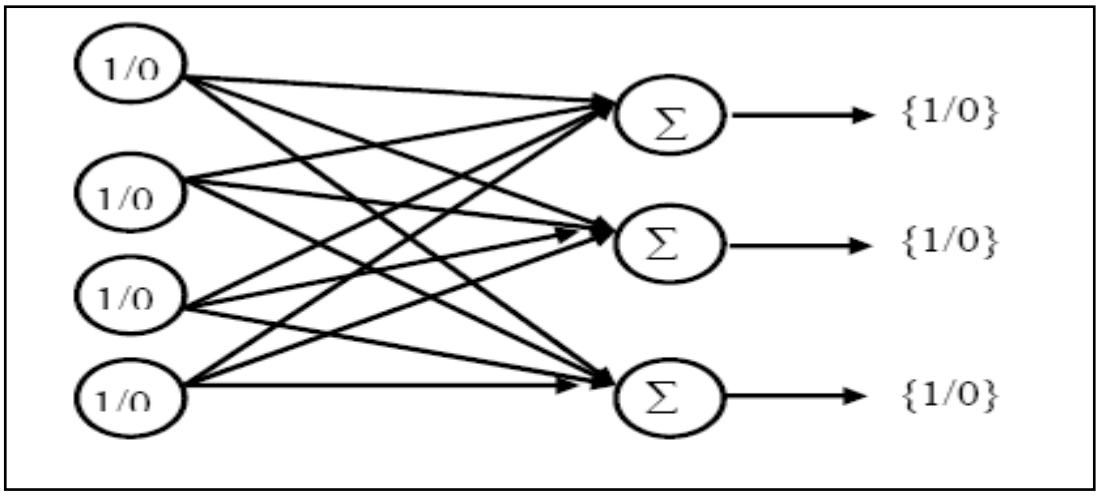


Figure 2.11 : Schéma général de perceptron simple [23]

6-2- Réseaux auto-organisateur (réseau de Kohonen)

Les cartes topologiques ou cartes auto organisatrices ont été introduites pour la première fois par T. Kohonen en 1981. Les premiers modèles cherchaient tout particulièrement à représenter des données multidimensionnelles. La particularité la plus importante des cartes auto-organisatrices est qu'elles rendent possible la comparaison des groupements qui ont réalisé directement à partir des données. Une observation est affectée à un groupe qui est projeté en un nœud de la carte. La comparaison des projections liées à deux observations distinctes permet d'apprécier la proximité des groupes dont elles sont issues. **[27]**

Dans la plus part des applications, les neurones d'une carte de Kohonen sont disposés sur une grille 2D (Figure 2.12). Chaque neurone i de la carte effectue un calcul de la distance euclidienne entre le

7Marvin Lee Minsky : est un scientifique américain, il a travaillé dans le domaine de l'intelligence artificielle, a montré en 1969 quelques limitations théoriques du perceptron simple.

vecteur d'entrée et le vecteur poids W_i, Dans les réseaux de Kohonen, la mise à jour des paramètres des neurones s'effectue sur tout un voisinage d'un neurone i, Un rayon de voisinage **r** représente donc la longueur du voisinage d'un neurone i en termes de nombre de neurones. On définit alors une fonction **h(i,k)** égale à 1 pour tous les neurones **k** voisins du neurone **i** compris dans le rayon **r** et égal à zéro pour tous les autres neurones.

L'algorithme d'apprentissage de la carte de Kohonen se présente comme suit : Après un temps très important de convergence, le réseau évolue de manière à représenter au mieux la topologie de l'espace de départ. Il faut noter que la notion de conservation de la topologie est en fait abusive puisqu'en général, la taille du vecteur d'entrée est bien supérieure à la dimension de la carte (souvent égale à 2) et il est donc impossible de conserver parfaitement la topologie.

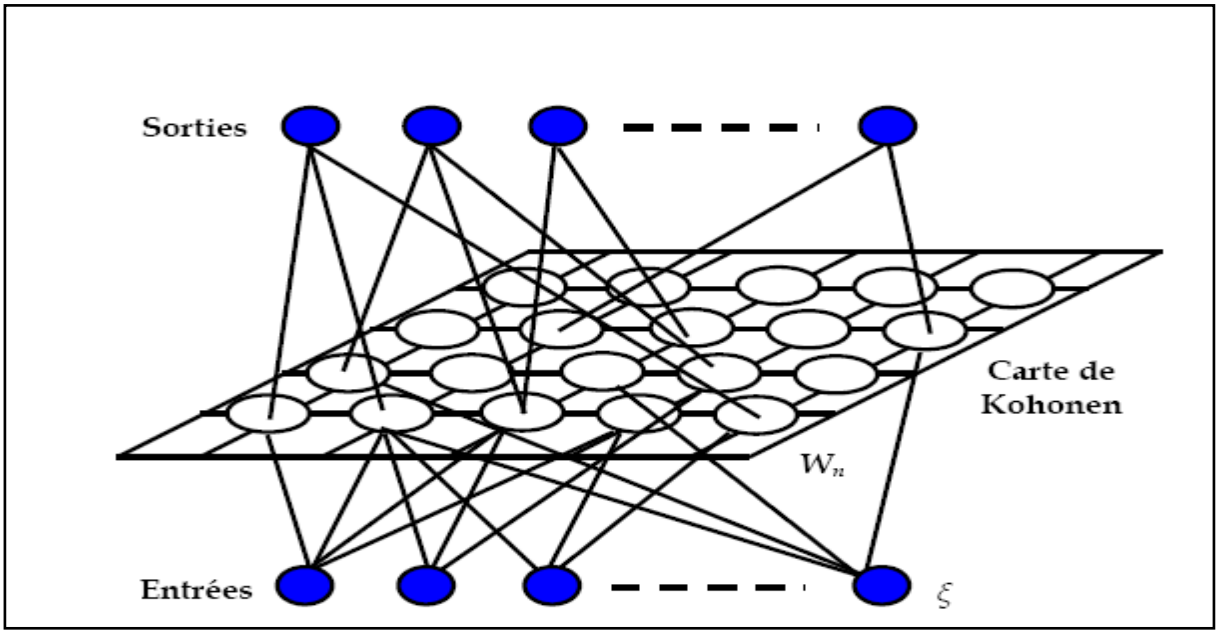


Figure 2.12: Carte topologique auto-adaptative de Kohonen [23]

Les avantages de la carte auto-organisatrice sont :

- L'espace de sortie est un espace de représentation donc on peut visualiser les sorties de la carte.
- Représentation des données de grande dimension.

L'un des inconvénients est le temps de convergence qui est très long. Il n'y a pas de preuve de convergence en multidimensionnel et d'unicité de la représentation

6-3-Perceptron Multicouches (MLP) :

Le perceptron Multicouches est un réseau orienté de neurones artificiels organisé en couches et où l'information voyage dans un seul sens, de la couche d'entrée vers la couche de sortie. Et vu que ce modèle de réseaux de neurones est celui que nous allons utiliser pour mettre en œuvre notre système de détection d'intrusion, nous allons l'aborder en détail dans ce qui suit de ce chapitre.

6-4-Le modèle de Hopefield:

Le modèle de Hopfield fut présenté en 1982. Ce modèle très simple est basé sur le principe des mémoires associatives. C'est d'ailleurs la raison pour laquelle ce type de réseau est dit associatif (par analogie avec le pointeur qui permet de récupérer le contenu d'une case mémoire).

Le modèle de Hopfield utilise l'architecture des réseaux complètement connectés et récurrents (dont les connexions sont non orientées et où chaque neurone n'agit pas sur lui-même).Les sorties sont en fonction des entrées et du dernier état pris par le réseau.

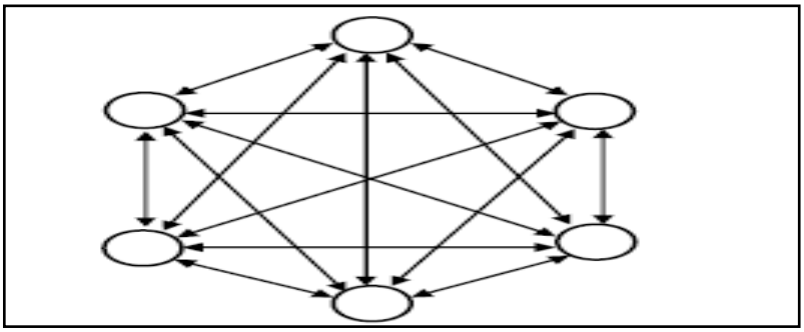


Figure 2.13: Le modèle de Hopfield [24]

7 -Le perceptron multicouches MLP : **(Multi Layer Perceptron)**

Le MLP avec son algorithme d'apprentissage, la rétro-propagation des erreurs est l'un des types des réseaux de neurones les plus utilisés pour la résolution de divers problèmes d'intelligence artificielle.

Le MLP ou PMC (perceptron multicouches) est un réseau orienté de neurones artificiels en couches, où l'information circule dans un seul sens, de la couche d'entrée vers la couche de sortie. La Figure 2.14 donne un exemple d'un réseau contenant une couche d'entrée, deux couches cachées et une couche de sortie. La couche d'entrée présente toujours une couche virtuelle associée aux entrées du système.

Dans l'exemple, il y a 3 neurones d'entrée, 4 neurones sur la première couche cachée, trois sur la deuxième couche cachée et 4 neurones sur la couche de sortie, Dans le cas général, un MLP peut posséder un nombre de couches quelconque et un nombre de neurones (ou d'entrées) par couche également quelconque. **[25]**

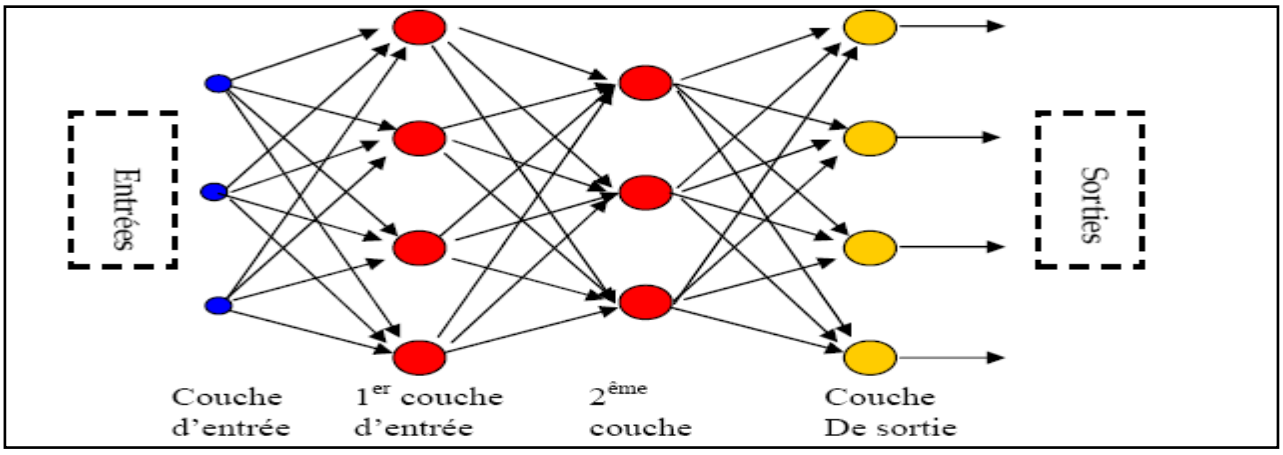


Figure 2.14: Exemple de réseau de type perceptron Multicouche [23]

7-1- Mise en œuvre du réseau de neurones MLP :

La mise en œuvre des réseaux de neurones comporte à la fois une partie conception, dont l'objectif est de permettre de choisir la meilleure architecture possible, et une partie de calcul numérique, pour réaliser l'apprentissage d'un réseau de neurones, mais en vue de perfectionner le fonctionnement du MLP d'un côté et réduire au maximum le temps de calcul d'autre part, on doit chercher une architecture optimale au point de vue nombre de couche , nombre de neurones par couche et nombre de sorties possibles.

A partir d'une architecture de réseau de neurones donné et des exemples disponibles (la base d'apprentissage), on détermine les poids optimaux, par l'algorithme de la rétro-propagation des erreurs, pour que la sortie du modèle s'approche le plus possible du fonctionnement désiré. **[20]**

7-2- L'apprentissage des réseaux MLP :

L'apprentissage d'un réseau de neurones multicouches se fait généralement par l'algorithme de la rétro-propagation« back-propagation » qui est l'exemple d'apprentissage supervisé le plus utilisé pour les MLP, Il utilise une méthode d'optimisation universelle consiste à trouver les coefficients du réseau (poids) minimisant une fonction d'erreur globale (fonction coût).

La technique de rétro-propagation du gradient est une méthode qui permet de calculer le gradient de l'erreur pour chaque neurone du réseau, de la dernière couche vers la première, le principe de la rétro-propagation peut être décrit en trois étapes fondamentales :

7-2-1-Propagation avant :

L'apprentissage neuronal basé sur cet algorithme fait appel à des exemples de comportement. Soit une base d'apprentissage constituée de N exemples d'apprentissage, chacun étant constitué d'un vecteur $\boldsymbol{x}(\boldsymbol{n})$ appliqué aux entrées du réseau, et du vecteur $\boldsymbol{d}(\boldsymbol{n})$ des valeurs désirées correspondantes pour les sorties, le vecteur $\boldsymbol{y}(\boldsymbol{n})$ correspond à la sortie du réseau pour l'entrée $\boldsymbol{x}(\boldsymbol{n})$. On suppose aussi que le réseau de neurones possède un nombre *r* de neurones de sortie.

Dans cette phase, après l'initialisation des poids (**w**) on calcule la valeur d'entrée de chaque neurone *j* du réseau qui est égale à la somme pondérée des valeurs de sortie des neurones de la couche précédente, la formule permettant de calculer cette valeur s'écrit :

$$VEj = \sum_{i \epsilon \text{ pred (j)}} \mathbf{Wij\ VAi}$$

la valeur de sortie (activation) de chaque neurone *j* (VA_j) *est* calculée ensuite à l'aide de fonction de transfert *f*, $VA_j = (VE_j)$

Si on utilise comme fonction de transfert ***f*** la fonction sigmoïde que nous avons utilisé dans notre travail, la valeur d'activation est donnée par :

$$VA_j = f(VE_j) = \frac{1}{1-e^{-VEJ}}$$

On répète cette opération jusqu'à le calcul des valeurs d'activation des neurones de sorties, La différence entre ces valeurs et les valeurs désirées correspondantes pour les sorties représente l'erreur d'apprentissage appelée delta :$(\Delta) = d(n) - y(n)$ qui doit être inférieure à un seuil fixé préalablement.

7-2-2- Rétro-propagation :

Après le calcul de l'erreur d'apprentissage qui est le résultat d'étape de la propagation vers l'avant des entrées, dans cette phase en va rétro-propager cette erreur à travers les couches du réseau en allant des sorties vers les entrées (vers l'arrière), cette erreur va donc être distribuée aux neurones de couches cachées afin de pouvoir ajuster dans la phase suivante les poids du réseau, le calcul de l'erreur (Δ) de chaque neurone j de la couche cachée se fait à l'aide de la formule suivante:

$$\Delta_j = VA_j\,(1 - VA_j) \sum_{k \epsilon \text{ succ (j)}} \mathbf{W_{jK}\,\Delta_k}$$

7-2-3- La mise à jour des poids : [27]

A la fin de l'étape précédente, l'erreur d'apprentissage a été distribuée (rétro-propager) à tous les neurones des couches cachées, et maintenant dans la phase actuelle on recalcule les poids du réseau qui sont au préalable initialisés avec des valeurs aléatoires en utilisant la règle suivante:

Nouveau poids = ancien poids + taux d'apprentissage *erreur de neurone actuel*sortie de la couche précédente, Cela est décrit par la formule :

$$W_{ij} = W_{ij} + \alpha * \Delta_j * VA_i$$

Où : αest le taux d'apprentissage, qui est généralement dans l'intervalle [0,1] (choisi par l'utilisateur).

Lors de l'apprentissage du réseau de neurones, ces trois phases (Propagation avant, rétro-propagation et mise à jour des poids) sont répétées autant de fois que le nombre d'exemples de la base d'apprentissage. Une fois terminé, la somme des carrées des erreurs d'apprentissage (l'erreur quadratique

moyenne (EQM)) est calculé, cette mesure est souvent applicable en classification binaire où les classes sont 0 et 1. C'est une mesure liée à des probabilités, elle est définie par la formule**:[25]**

$$\mathbf{EQM} = \sqrt{\frac{1}{\mathbf{n}} \sum_{k=1}^{n} (\mathbf{dk} - \mathbf{yk})^2}$$

C'est une étape de validation de réseau Où :

$\mathbf{d_k}$: la valeur désirée dans l'exemple d'apprentissage.

y_k: la valeur de neurone de la couche de sortie calculée par le MLP.

La valeur d'EQM obtenue doit être inférieure à un certain seuil pour lequel on peut dire que le modèle obtenu réagit bien aux exemples d'apprentissage, sinon la procédure sera répétée avec d'autres valeurs initiales des poids W_{ij} et du taux d'apprentissage α.

La phase d'apprentissage d'un réseau de neurone peut donc être résumée par l'algorithme suivant:

-Initialisation des poids avec des valeurs aléatoires comprises dans un intervalle choisi

-Lecture des exemples d'apprentissage.

-Normaliser les données d'entrainement;

Répéter

Pour chaque exemple d'apprentissage **Faire**

-Propagation de l'entrée vers l'avant

-Propagation de l'erreur vers l'arrière (rétro-propagation)

-Mise à jour des poids

Fin pour

-Calcul de l'erreur totale(EQM)

Tant que l'erreur quadratique moyenne (EQM) est supérieure au SEUIL OU le nombre maximum d'itérations atteint.

Figure 2.15 : Algorithme de rétro-propagation de gradient. [26]

L'efficacité de l'algorithme de la rétro-propagation dépend, en effet, d'un grand nombre de paramètres que l'utilisateur doit fixer : le pas du gradient, les paramètres des fonctions sigmoïdes, l'architecture du réseau (nombre de couches, nombre de neurones par couche), l'initialisation des poids.

8-Test et évaluation :

Une fois le réseau de neurones est entraîné (après l'apprentissage et la validation), il est nécessaire de le tester sur une autre base de données différente de celle utilisée pour l'apprentissage qui est appelée

généralement base de test**.** Ce test permet à la fois d'apprécier les performances du système neuronal en calculant les différentes métriques d'évaluation, telles que la précision, le rappel, le taux de réussite, etc. comme nous illustrons dans le prochain chapitre.

9-Avantages et limites des réseaux de neurones:

9-1 -Avantages :

- **Réutilisabilité:** Un réseau de neurones n'est pas programmé pour une application mais pour une classe de problèmes : après une phase d'apprentissage adéquate, il peut traiter de nombreuses tâches.
- **Robustesse :** Les couches cachées du réseau de neurone forment une représentation abstraite des données (concepts), qui permettent de savoir catégoriser des données non traitées lors de l'apprentissage (non prévues).
- **Parallélisme :** L'architecture des réseaux permet théoriquement à un grand nombre d'éléments de calcul simples de travailler d'une façon concurrente, ce qui facilite l'obtention des résultats très rapides et aide à l'implantation des applications ayant notamment des contraintes temps-réel.
- **Logique flou :** Les réseaux de neurone se sont inspirés du fonctionnement du cerveau humain, ils savent utiliser des notions imprécises**,** modélisent des systèmes dynamiques et non linéaires, le réseau établit lui-même ses connaissances, à partir d'exemples.

9-2- Limites:

- **Choix des attributs :** Pour permettre de travailler avec les réseaux de neurones, il est nécessaire de choisir soigneusement la représentation des données. Les attributs ne peuvent être que numériques.
- **Processus d'apprentissage**: Lorsque la durée d'apprentissage est très longue, la possibilité de perdre la capacité de généralisation par le RN augmente c.-à-d. apprentissage au détriment de la généralisation (Le problème du sur-apprentissage).
- **Architecture du réseau :** Le nombre de nœuds dans les couches d'entrée et de sortie sont généralement fixés par l'application, mais comment optimiser le nombre de niveaux cachés et le nombre de nœuds dans ces niveaux ? Il n'existe pas des règles claires dans ce sens.
- **Exploitabilité:** Il existe une grande difficulté pour expliquer les résultats obtenus par le réseau de neurones, car ce dernier fonctionne comme une boite noire et peut découvrir des règles et l'exploitent pour résoudre des problèmes, mais il ne permet pas la possibilité d'extraire des lois ou des formules depuis ces règles. **[27]**

10-Domaines d'applications des réseaux de neurones :

- Traitement d'image : compression d'images, reconnaissance de caractères et de signatures, reconnaissance de formes et de motifs, cryptage, classification, etc.
- Traitement du signal : traitement de la parole, identification de sources, filtrage, classification, etc.
- Traitement automatique des langues : segmentation en mots, représentation sémantique des mots, étiquetage morphosyntaxique, traduction automatique, etc.
- Contrôle : diagnostic de pannes, commande de processus, contrôle qualité, robotique, etc.
- Optimisation : allocation de ressources, planification, régulation de trafic, gestion, finance, etc.
- Simulation : simulation boîte noire, prévisions météorologiques.
- Classification d'espèces animales étant donnée une analyse ADN.
- Modélisation de l'apprentissage et perfectionnement des méthodes de l'enseignement.
- Approximation d'une fonction inconnue ou modélisation d'une fonction connue mais complexe à calculer avec précision

-Conclusion :

Dans ce deuxième chapitre, nous avons présenté en premier lieu une introduction au domaine de classification automatique des données, où nous avons abordé les notions de base de ce domaine, les différentes catégories de la classification ainsi que les différentes techniques de classification de chaque catégorie.

Dans la deuxième partie, nous avons introduit les définitions essentielles relatives aux réseaux de neurones. Nous avons notamment distingué entre les réseaux de neurones non bouclés (statiques), qui réalisent des fonctions non linéaires, et ceux bouclés (dynamiques) qui réalisent des équations aux différences non linéaires.

Nous avons aussi mis l'accent sur l'utilisation des réseaux de neurones comme outils de modélisation par apprentissage. Ces derniers permettent d'ajuster des fonctions non linéaires très générales à des ensembles de points. Comme toute méthode qui s'appuie sur des techniques statistiques, l'utilisation de réseaux de neurones nécessite que l'on dispose de données suffisamment nombreuses et représentatives, Enfin, nous avons décrit les applications des réseaux de neurones.

Chapitre3 :

Optimisation et algorithmes génétiques

- Introduction :

L'optimisation est un ensemble de techniques permettant de trouver les valeurs des variables qui rendent optimale une fonction de réponse, appelée aussi fonction objective. Sur le plan mathématique, cela correspond à la recherche des extrémums de fonctions à plusieurs variables. Dans le domaine des sciences appliquées, il s'agit en général de trouver l'optimum de la réponse d'opérations industrielles ou d'expériences de laboratoire ou d'une solution à un problème précis.

Dans ce chapitre nous allons aborder les techniques d'optimisation de manière générale, ensuite nous présentons l'algorithme génétique comme l'une des techniques d'optimisation choisi d'être utilisée dans notre projet conjointement avec le réseau de neurones multicouches (MLP) pour optimiser les valeurs des poids de ce dernier afin d'obtenir un meilleur résultat de classification du modèle de détection d'intrusion généré.

I -Les algorithmes d'optimisation :

Les algorithmes d'optimisation ont fait leur apparition afin d'avoir des solutions réalisables de bonne qualité pour des problèmes difficiles issus des domaines de la recherche opérationnelle ou de l'ingénierie dont nous ne connaissons pas des méthodes efficaces pour les traiter ou bien quand la résolution du problème nécessite un temps élevé ou une grande mémoire de stockage. Ces algorithmes peuvent être aussi utilisés conjointement avec les algorithmes de classification pour optimiser certains paramètres afin de converger vers les meilleures solutions avec moins de temps possible.

1. Définitions :

Un algorithme d'optimisation est une procédure mathématique qui permet de maximiser ou de minimiser une fonction réelle ***f,*** que l'on appelle fonction objective.

Les algorithmes d'optimisation sont des processus itératifs qui génèrent une séquence de valeurs à partir d'un point de départ. On dit qu'un algorithme est convergent quand pour n'importe quel point de départ, la séquence arrive à la solution (maximum ou minimum).Et la non convergence d'un algorithme n'implique pas la non existence de solution, il faut essayer avec un autre point de départ ou bien utiliser un autre type d'algorithme.

C'est très important de visualiser la fonction que l'on veut optimiser avant de choisir l'un ou l'autre des algorithmes et pour déterminer la valeur initiale à introduire pour notre analyse. Le même algorithme peut être utilisé pour résoudre les différents types de problèmes d'optimisation (Algorithmes méta heuristiques), qui représente l'une des familles de techniques d'optimisation. **[28]**

2- Les grandes familles de techniques d'optimisation :

Les techniques d'optimisation peuvent être classées en trois grandes familles :

- **les méthodes exactes:** Ces méthodes s'appliquent aux problèmes qui peuvent être résolus de façon optimale et rapide. Elles sont basées soit sur une résolution algorithmique, analytique ou sur une énumération exhaustive de toutes les solutions possibles, dans le but de trouver des solutions exactes à des problèmes difficiles**,** (exemples: méthode de branche and bound, algorithme de simplexe, etc).

- **les méthodes heuristiques** : une heuristique est un algorithme approché qui permet d'identifier au moins une solution réalisable rapide, utilisé pour calculer une solution spécifique approchée d'un problème bien déterminé et ainsi accélérer le processus de résolution exacte. Généralement une heuristique est conçue pour un problème particulier, sans offrir aucune garantie sur la qualité de la solution calculée.

- **les méta-heuristiques :** sont des méthodes approchées qui peuvent résoudre différents problèmes, elles sont souvent inspirées par des systèmes naturels, qu'ils soient pris en physique (cas du recuit simulé), en éthologie (cas des algorithmes de colonies de fourmis ou de l'optimisation par essaims particulaires) ou encore en biologie de l'évolution (cas des **algorithmes génétiques**).

Le principal avantage des méthodes **méta-heuristiques** est qu'elles peuvent s'appliquer à n'importe quelle classe de problèmes faciles ou très difficiles, bien ou mal formulés, avec ou sans contrainte, en particulier elles ne nécessitent pas une modélisation mathématique du problème.

Elles semblent être tout à fait adaptées à notre optimisation des poids de réseaux de neurones et donc notre choix a été fait sur l'algorithme génétique pour optimiser les poids de réseaux de neurones afin d'améliorer la performance de classification de ces derniers. **[29]**

II-Les algorithmes génétiques:

Les algorithmes génétiques (AG) sont des algorithmes d'optimisation s'appuyant sur des techniques dérivées de la génétique et de l'´évolution naturelle. Ils appartiennent à la famille des algorithmes évolutionnistes : un sous-ensemble des méta-heuristiques. Fondés sur les mécanismes de la sélection naturelle et de la génétique, ils ont été initialement développés et adaptés à l'optimisation par John Holland en 1975. Leurs champs d'application sont très vastes ; l'économie, l'optimisation de fonctions, finance, en théorie du contrôle optimal, théorie des jeux répétés et différentiels, etc.

1-Présentation des algorithmes génétiques:

Un algorithme génétique est défini par les caractéristiques suivant:

-Séquence/Chromosome/Individu (Codage binaire) : « ***Nous appelons une séquence (chromosome, individu) A de longueur l(A) une suite A = {a1,a2,...,al} avec*** $\forall i \in [1,l]$, $ai \in V = \{0,1\}$ **».[35]**

Un chromosome est donc une suite de bits en codage binaire, appelé aussi chaîne binaire. Dans le cas d'un codage non binaire, tel que le codage réel, la suite A ne contient qu'un point réel, nous avons

A = {a} avec a $\in \Re$.

- **Population** : un ensemble de chromosomes ou de points de l'espace de recherche.
- **Gène** : Un gène sera une partie d'une solution au problème, donc d'un individu.
- **Environnement** : l'espace de recherche.
- **Fonction de fitness** : la fonction - positive - que nous cherchons à maximiser.

Le codage des données :

La première étape est de définir et coder convenablement le problème. Le codage de chaque individu de la population est essentiel dans l'élaboration d'un algorithme génétique dont dépend notamment de l'implémentation des opérateurs de transformations. Ainsi, cette phase détermine la structure de données qui sera utilisée pour coder le génotype des individus de la population. Le codage doit donc être adapté au problème traité. Plusieurs types de codages sont utilisés dans la littérature, parmi lesquels le codage par une séquence binaire de longueur fixe et le codage réel. **[30]**

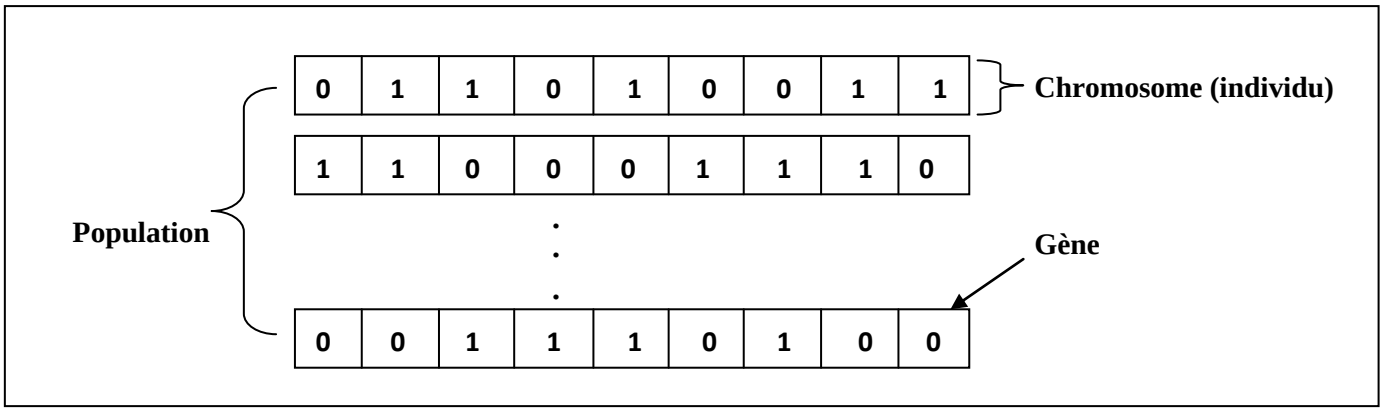


Figure 3.1 : Codage binaire des données pour l'algorithme génétique

2- Principe de fonctionnement :

Le fonctionnement des algorithmes génétiques est extrêmement simple, nous partons d'une population de solutions potentielles (chromosomes) initiales, choisies arbitrairement ou de n'importe quelle manière. Nous évaluons leur performance (Fitness) relative. Sur la base de ces performances nous générons une nouvelle population de solutions potentielles en utilisant des opérateurs évolutionnaires simples : la sélection, le croisement et la mutation. Quelques individus se reproduisent, d'autres disparaissent et seuls les individus les mieux adaptés sont supposés survivre. Nous recommençons ce cycle jusqu'à ce que nous trouvions une solution satisfaisante.

En effet, l'héritage génétique à travers les générations permet à la population d'être adaptée et donc répondre au critère d'optimisation, la figure 3.2 illustre les principales étapes d'un algorithme génétique. Un algorithme génétique recherche le ou les extrema d'une fonction définie sur un espace de données. Sa mise en œuvre nécessite :

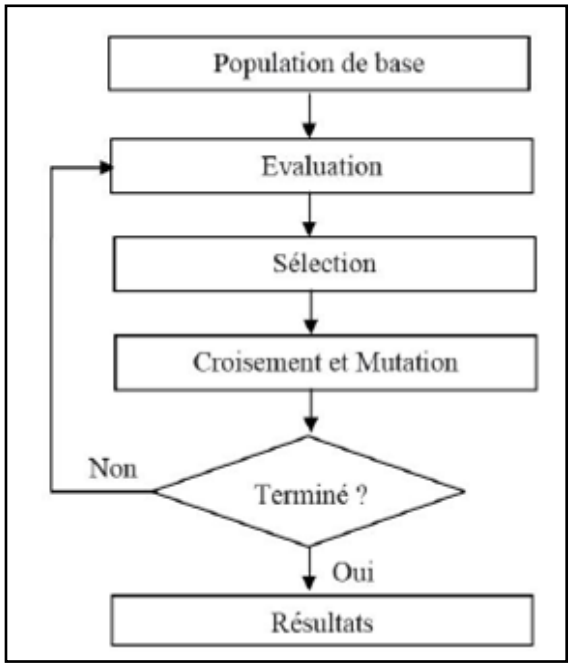


Figure 3.2 : Organigramme du fonctionnement d'un algorithme génétique.

- **L'évaluation (Fitness)**

L'évaluation de la Fitness est généralement l'étape où on mesure la performance de chaque individu pour pouvoir juger sa qualité et ainsi la comparée aux autres, il faut établir une mesure commune d'évaluation. Aucune règle n'existe pour définir cette fonction, son calcul peut ainsi être quelconque, que ce soit une simple équation ou une fonction affine. La manière la plus simple est de poser la fonction d'adaptation comme la formalisation du critère d'optimisation.

Dans notre cas, nous avons basé sur le principe que le taux des erreurs de classification soit très faible c.-à-d. : **FP+FN<μ**tel que 0<**μ**<N et **N :** Nombre total des enregistrements de la base KDDtest+, cela donne : **((FP+FN)/ N) < (μ /N)** alors : **((FP+FN)/ N) <seuil** tel que **: seuil= μ /N**

Donc le **seuil** appartient à l'intervalle **[0,1],** il est défini par l'utilisateur d'une façon qu'il soit au minimum le plus possible.

- **La sélection**

Permet d'identifier statistiquement les meilleurs individus d'une population et d'éliminer les mauvais pendant le passage d'une génération à une autre, ce processus est basé sur la performance (l'évaluation) de l'individu. Il existe beaucoup de façons de sélectionner une sous-population, parmi lesquelles :

Sélection uniforme: La sélection se fait aléatoirement, et sans intervention de la valeur d'adaptation. Chaque individu a donc une probabilité 1/P d'être sélectionné, où P est la taille de population.

Sélection par rang : Cette technique de sélection choisit toujours les individus possédant les meilleurs scores d'adaptation, le hasard n'entre donc pas dans ce mode de sélection, la sélection appliquée consiste à conserver les meilleurs individus (au sens de la fonction d'évaluation).

Roue de la fortune (roulette wheel) : Selon cette méthode, chaque chromosome sera dupliqué dans une nouvelle population proportionnellement à sa valeur d'adaptation. Nous effectuons, en quelque sorte, autant de tirages avec remises qu'il y a d'éléments dans la population., la fitness d'un chromosome particulier étant f (d(ci)), la probabilité avec laquelle il sera réintroduit dans la nouvelle population de taille N est : $\frac{f(d(Ci))}{\sum_{j=1}^{N} f(d(Cj))}$, Les individus ayant une grande fitness ont donc plus de chance d'être sélectionnés. On parle alors de sélection proportionnelle.

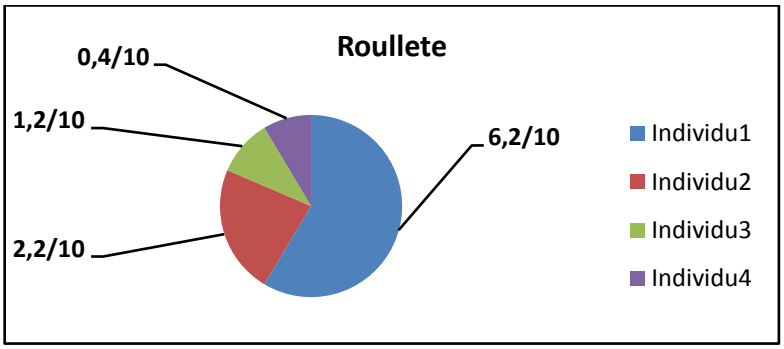


Figure 3.3 : Exemple d'une roulette de la fortune.

Nous avons utilisé, dans notre optimisation des poids de réseau de neurones avec l'algorithme génétique, en premier lieu la sélection des meilleurs individus pour construire la population initiale. Ensuite nous avons utilisé la roue de la fortune à chaque itération pour sélectionner les individus élus pour l'opération de croisement.

-Croisement (Crossover):

L'opérateur de croisement favorise l'exploration de l'espace de recherche et enrichit la diversité de la population en manipulant la structure des chromosomes, le croisement fait avec deux parents et génère deux enfants, en espérant qu'un des deux enfants au moins héritera de bons gènes des deux parents et sera mieux adapté qu'eux. Il existe plusieurs méthodes de croisement par exemple le croisement en un point (Figure 3.4), ou en multiples points.

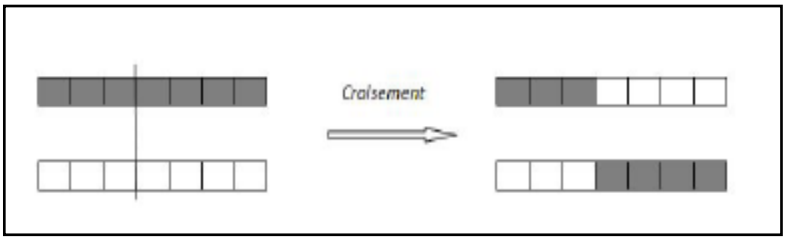


Figure 3.4 : Principe de croisement

-La mutation : La mutation est également identique à la mutation biologique, elle vise à modifier légèrement un gène à une position aléatoire dans le chromosome (Figure 3.5).

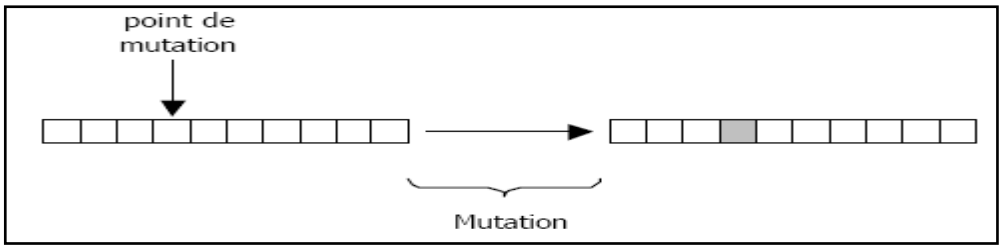


Figure 3.5 : Principe de mutation

3-Les paramètres de l'algorithme génétique :

L'évolution algorithmique a ceci d'intéressant par rapport à l'évolution biologique qu'il est aisé d'en contrôler tous les paramètres. Ainsi, on peut fixer la taille de la population, le nombre de générations, le génome de chaque individu ainsi que les taux et les méthodes de croisement et de mutation.

-La taille de la population : définit le nombre d'individus présents au départ de l'évolution. Les individus étant générés aléatoirement, alors plus la population est grande, plus grande sera la diversité des individus au départ.

-Le nombre de générations : définit pendant combien de cycles on va faire évoluer la population, un cycle représentant une itération évaluation-sélection-recombinaison (croisement des individus) de la population. Plus ce nombre est important, plus la population convergera vers la solution optimale. En général, ce nombre n'est pas fixé et l'évolution s'arrête quand un des individus de la population est jugé acceptable.

-Le taux de croisement : c'est le pourcentage de la population qui sera renouvelé par croisement. Un fort taux de croisement permet de rapidement élever la moyenne (en termes d'adaptation) des individus de la population mais peut également nuire à la diversité, qui est une clé essentielle du succès de l'algorithme génétique).

-Le taux de mutation : définit le pourcentage d'individus qui seront mutés à chaque génération.

4-L'utilisation des algorithmes génétiques conjointement aux réseaux de neurones :

Vu que les résultats obtenus par les réseaux de neurones pour la résolution d'un problème précis dépend d'un ensemble de paramètres tels que la taille du réseau, sa topologie, et la précision des poids de ses liaisons ; donc l'optimisation des valeurs de ces paramètres va nous permettre de trouver la bonne solution possible avec moindre de temps de calcul. Les algorithmes génétiques représentent l'une des méthodes d'optimisation de réseau de neurones et d'amélioration des poids de liens entres les neurones de ce réseau.

Notre approche consiste à utiliser l'algorithme génétique pour optimiser les poids des liaisons de réseau de neurones en partant des poids obtenus dans la dernière itération d'apprentissage d'algorithme de

rétro-propagation du gradient pour MLP. Ensuite la sélection de la population initiale, puis en effectuant les opérations de sélection avec la méthode de Roulettte wheel, croisement avec un taux (Pc) et la mutation avec un taux (Pm),

Pour le codage des données dans notre approche nous avons recopié les valeurs des poids des liens entres les neurones du MLP telles qu'elles (codage réel) dans une chaine qui représente un individu (chromosome) de la population pour implémenter l'algorithme génétique. Comme le montre la figure suivante:

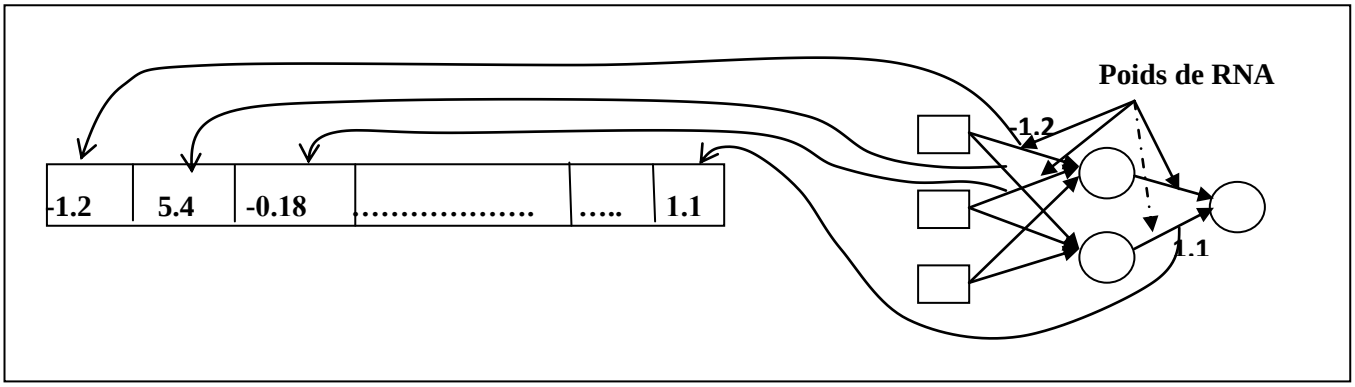


Figure 3.6: Création des individus d'algorithme génétique depuis le réseau de neurones

Nous répétons les étapes montrées dans l'algorithme décrit par l'organigramme ci-dessous (Figure 3.7) plusieurs itérations jusqu'à en arrive à une génération de population qui contienne un meilleur individu répondant à un critère défini préalablement et représentant la solution optimale (dans notre cas l'individu qui donne un bon taux de réussite).

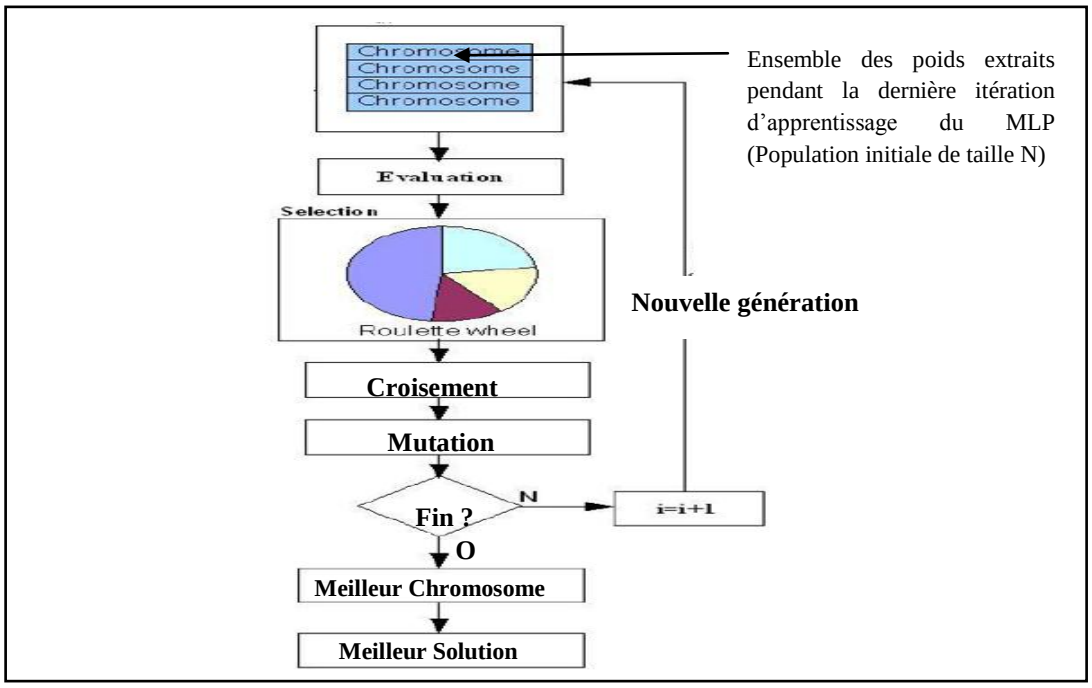


Figure 3.7 : Les étapes d'optimisation des poids de RNA par algorithme génétique.

-Conclusion :

Dans ce chapitre, Nous avons abordé dans sa première partie les techniques d'optimisation d'une manière générale ainsi que leurs classifications avec quelques exemples.

Aussi, on a montré dans la deuxième partie l'utilité d'algorithme génétique comme l'une des techniques d'optimisation qui présente un intérêt considérable, son principe de fonctionnement qui s'appuie essentiellement sur des opérateurs de sélection, croisement et mutation. Ces opérateurs permettent de dégager une solution optimale pour un problème précis.

Afin de montrer les avantages des algorithmes génétiques nous avons utilisé ces derniers dans notre travail pour améliorer les valeurs des poids qui relient les neurones d'une couche à l'autre dans le but d'augmenter la performance de classification du modèle MLP.

Chapitre4 :

Implémentation et discussion des résultats

Introduction:

Notre application se base sur le modèle MLP du réseau de neurones artificiels déjà décrit dans le chapitre précédent pour faire une classification supervisée des connexions TCP/IP de la base NSL_KDD afin d'établir un système de détection d'intrusions basé sur l'analyse du comportement de ces connexions et permet de les classifier en deux types (attaque et normale).

Pour cela nous avons montré dans ce chapitre les différentes phases du développement de notre application, en commençant par une description de la base de données NSL-KDD et les étapes de prétraitement que nous avons fait sur cette dernière, ensuite nous présentons notre modèle de classification suivi par la discussion des résultats obtenus.

I- Description de la base NSL-KDD et présentation du modèle de classification:

1-Description de la base NSL-KDD :

La base NSL-KDD (**N**etwork **S**ecurity **L**ayer-**K**nowledge **D**iscovery in **D**atabases) a été fondé sur l'ensemble de données KDD99, Cette dernière est une base de données qui contient des connexions TCP /IP extraites de l'ensemble de données d'évaluation des systèmes de détection d'intrusions. KDD99 été réalisées en 1998 par l'agence de L'armé américain DARPA (Défense Advanced Research Projects Agency) et AFRL (Laboratoire de recherche de l'armée de l'air), ensuite MIT Lincoln Labs[8] a collecté et distribué les ensembles de données pour l'évaluation du système de détection d'intrusions de réseau informatique.

La base NSL-KDD est un ensemble de données qui représente une version réduite de l'originale KDD 99,proposé en 2010 par les chercheurs dans le domaine de détection d'intrusions réseaux afin de résoudre certains problèmes qui ont apparu dans la base KDD 99. NSL-KDD considérée comme un ensemble de données de référence pour aider les chercheurs à comparer les différentes méthodes de détection d'intrusions. Le NSL-KDD présente les différences suivantes par rapport à l'originale KDD 99:

- Il n'inclut pas les enregistrements redondants dans les données d'apprentissage, ce qui améliore la performance de classification.
- Il n'y a pas d'enregistrements en double dans les ensembles de test proposés, ce qui aide à l'obtention de meilleurs taux de détection.
- Le nombre d'enregistrements dans les données d'apprentissage et les ensembles d'essais sont raisonnables, ce qui il est abordable d'exécuter les expériences sur l'ensemble complet sans la nécessité de choisissez au hasard une petite portion. Par conséquent, les résultats d'évaluation des travaux de recherche seront cohérents et comparables.

8 Le **MIT** : Massachusetts Institute of Technology :Lincoln Laboratory, situé à Lexington, dans le Massachusetts, est un centre de recherche et de développement du département de la Défense des États-Unis chargé aux technologies de problèmes de sécurité nationale,

Les données NSL-KDD contiennent des enregistrements de connexion TCP/IP, dont chaque enregistrement est constitué de 41 attributs caractérisant la connexion, et un attribut représente 5 classes qui sont : normales et 4 types d'attaques.

Les principaux types d'attaques de l'ensemble de données NSL-KDD sont (Probing, Dénis de Services, User to Root, Remote to User), Ces attaques ont été abordés en détail dans le premier chapitre de ce mémoire. **[31][32]**

1-1- Distribution des attaques de la base KDD99 :

Classes	Normal	Probe	DOS	R2L	U2L	Total
Nombre	97278	4107	391458	1126	52	494021
Pourcentage	19.69%	0.8313%	79.24%	0.2279%	0.0105%	100%

Tableau 4.1 : Répartition des attaques dans l'ensemble d'apprentissage KDD99

Classes	Normal	Probe	DOS	R2L	U2L	Total
Nombre	60593	4166	229853	16189	228	311029
Pourcentage	19.48%	1.34%	73.90%	5.20%	0.0 733%	100%

Tableau 4.2 : Répartition des attaques dans l'ensemble de Test KDD99

1-2- Le contenue de l'ensemble de données NSL-KDD :

KDDTrain + .ARFF: Ensemble complet NSL-KDD avec étiquettes binaires en format ARFF

KDDTrain + .TXT: Ensemble complet de trains NSL-KDD incluant les étiquettes d'attaque et le niveau de difficulté au format CSV

KDDTrain + _20Percent.ARFF: Un sous-ensemble de 20% du fichier KDDTrain + .arff

KDDTrain + _20Percent.TXT: Un sous-ensemble de 20% du fichier KDDTrain + .txt

KDDTest + .ARFF: Le test complet NSL-KDD avec des étiquettes binaires au format ARFF

KDDTest + .TXT: Ensemble de test complet NSL-KDD incluant les étiquettes d'attaque et le niveau de difficulté au format CSV

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

KDDTest-21.TXT: Sous-ensemble du fichier KDDTest+ .txt qui n'inclut pas les enregistrements ayant un niveau de difficulté de 21 sur 21.

1-3- Distribution des connexions réseau de NSL KDDTest+, et NSL KDDTrain_20%:

Catégorie	Nombre d'enregistrement en KDDTrain_20%		Nombre d'enregistrement en KDDTest+	
Normal	13499	53.39%	9711	43.08%
Dos	9234	36.65%	7458	33.08%
Probe	2289	9.09%	2421	10.74%
R2L	209	0.83%	2754	12.22%
U2R	11	0.04%	200	0.88%
Nombre total des enregistrements	25192		22544	

Tableau 4.3 : Distribution des connexions réseau de NSL KDDTest+, et NSL KDDTrain_20%

1-4- Attributs de la base NSL-KDD : Le tableau (1) de l'annexe récapitule les 41 attributs de la base NSL-KDD et leurs types de données. Ces attributs peuvent être classés en trois groupes :

- **Les attributs de base:** ces attributs décrivent les informations de base d'une connexion, telles que la durée, les hôtes source et destination, port et flag.
- **Les attributs du trafic:** ces attributs sont basés sur des statistiques, tels que le nombre de connexions vers la même machine.
- **Les caractéristiques du contenu:** ces attributs sont construits à partir de la charge utile (Data) des paquets du trafic tels que nombre d'échec de connexion et le nombre d'accès aux fichiers de contrôle.

2- Processus de génération du modèle de classification:

Car notre sujet traite la classification des connexions TCP/IP pour la détection d'intrusions pour cela et Comme pour tous modèles de classification, l'élaboration de notre modèle respecte les phases principales suivantes : le prétraitement, l'apprentissage et la phase de test. Comme l'indique le diagramme ci-dessous (Figure 4.1).

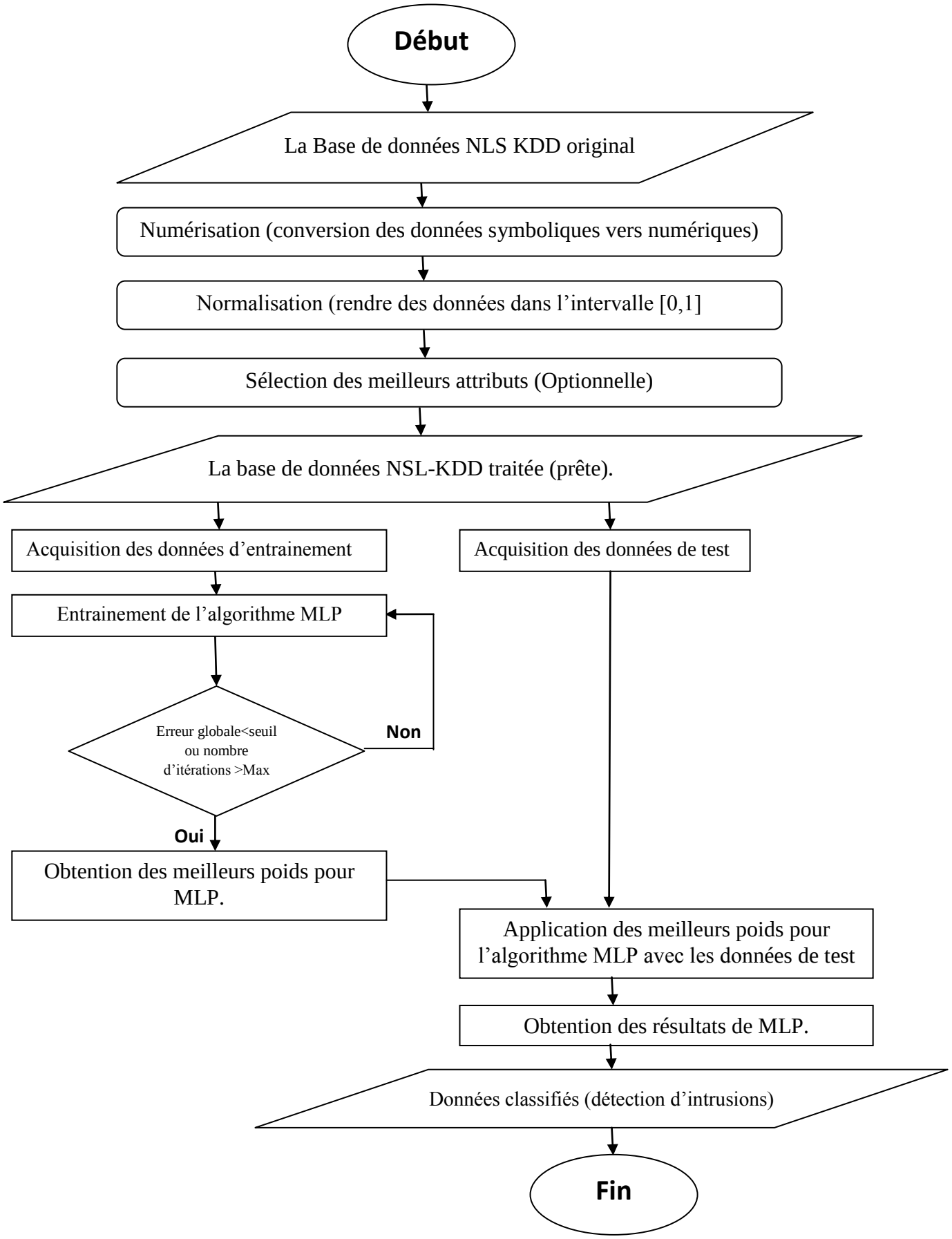


Figure 4.1 : Organigramme de fonctionnement de modèle de détection d'intrusion

2-1- Prétraitement de l'ensemble de données de la base NSL-KDD :

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

2-1-1-Numérisation (conversion des données symboliques vers numériques):

Comme nous l'avons dit précédemment la base de données de NSL-KDD contient 3attributs("protocol_type", "service" et "flag".) ont des données nominales. Sachant que les modèles de réseaux de neurones n'acceptent que des attributs numériques. Nous avons converti les trois attributs mentionnées ci-dessus vers des données numériques, Il existe plusieurs méthodes de conversion, parmi lesquelles la conversion alphabétique simple que nous avons choisi pour numériser les attributs de type nominal de la base de données NSL-KDD.

Conversion alphabétique simple: La conversion simple consiste à remplacer les valeurs des données catégoriques en ordre alphabétique par des nombres, Les données de l'attribut "type_protocole" par exemple, contiennent trois valeurs catégorielles distinctes : "tcp", "icmp" et "udp". Ces valeurs sont d'abord classées par ordre alphabétique puis on les attribuant un nombre pour chaque catégorie distinct de valeurs comme le montre (le tableau 4.4).

Avant conversion	Après conversion
Icmp	0
Tcp	1
Udp	2

Tableau 4.4 : La Conversion alphabétique simple des valeurs de l'attribut "protocol_type"

Pour l'attribut "service" qui contient 70 catégories distincts de valeurs, Après l'ordonnancement alphabétique de ces catégories, la valeur de donnée "aol" sera remplacé par "0", "auth" par "1" et "bgp" par "2" et on continue la conversion selon l'ordre alphabétique, comme le montre (letableau4.5).

Avant conversion	Après Conversion	Avant conversion	Après Conversion	Avant conversion	Après Conversion
aol	0	http_443	23	printer	46
auth	1	http_8001	24	private	47
bgp	2	imap4	25	red_i	48
courier	3	IRC	26	remote_job	49
csnet_ns	4	iso_tsap	27	rje	50
ctf	5	klogin	28	shell	51
daytime	6	kshell	29	smtp	52
discard	7	ldap	30	sql_net	53

domain	8	link	31	ssh	54
domain_u	9	login	32	sunrpc	55
echo	10	mtp	33	supdup	56
eco_i	11	name	34	Systat	57
ecr_i	12	netbios_dgm	35	Telnet	58
efs	13	netbios_ns	36	Tftp_u	59
exec	14	netbios_ssn	37	Tim_i	60
finger	15	netstat	38	Time	61
ftp	16	nnsp	39	urh_i	62
ftp_data	17	nntp	40	urp_i	63
gopher	18	ntp_u	41	uucp	64
harvest	19	other	42	uucp_path	65
hostnames	20	pm_dump	43	vmnet	66
http	21	pop_2	44	whois	67
http_2784	22	pop_3	45	X11	68
Z39_50			69		

Tableau 4.5 : La Conversion alphabétique simple des valeurs de l'attribut "service"

Le même principe est appliqué pour convertir les données de l'attribut Flag (tableau 4.6)

Avant conversion	Après Conversion	Avant conversion	Après Conversion
OTH	0	S1	6
REJ	1	S2	7
RSTO	2	S3	8
RSTOS0	3	SF	9
RSTR	4	SH	10
S0		5	

Tableau4.6 : La Conversion alphabétique simpledes valeurs de l'attribut"flag"

2-1-2-Normalisation:

Les valeurs obtenues après l'opération de la numérisation sont très variées et constituent un grand intervalle, Certains attributs prennent de grandes valeurs (src_bytes, dst_bytes, etc.), alors que d'autres ne prennent que des petites valeurs (serror_rate, same_srvrate, etc.), et cela peut nuire à la rentabilité du modèle de détection d'intrusions.

Afin d'éviter ce problème et garantir l'efficacité du modèle généré,les valeurs de la base données doivent être ajustées ou normalisées ; dans notre cas les données de la base NSL-KDD sont normalisés dans l'intervalle de [0, 1] en se basant sur une fonction de transfert. Nous avons utilisé la fonction Min-Max décrite par la formule suivante :

$$_{nouv}val=\left(\frac{\mathbf{ancval - Minanc}}{\mathbf{Maxanc - Minanc}}\right)\times (Max_{nouv} - Min_{nouv}) + Min_{nouv}$$

Où :

- val_{anc}: est la valeur à normaliser.
- val_{nouv} : est la valeur après la normalisation.
- Min_{anc}: est la limite inférieure de l'intervalle à qui val_{anc} appartient.

– Max_{anc}: est la limite supérieure de l'intervalle à que val_{anc} appartient.
– Min_{nouv}: est la limite inférieure de l'intervalle à que val_{nouv} va appartenir.
– Max_{nouv}: est la limite supérieure de l'intervalle à que val_{nouv} va appartenir.

2-1-3- La sélection d'attributs : [33]

Vu que la taille de la base de données NSL-KDD est très importante, dans notre cas (41 attributs et 25192 enregistrements pour l'entrainement et 22544 pour le test),donc le travail sur tous les attributs pour générer le modèle de classification sera très fastidieux et peut affecter considérablement les performances de l'algorithme d'apprentissage en matières de temps d'exécution et consommation des ressources systèmes, en plus les attributs de la base de données ne seront pas tous utilisés dans la classification des connexion TCP/IP effectuée par l'IDS pour détecter les attaques, certains étant plus pertinents que d'autres.

Pour ces raisons, et dans le cadre de notre projet, la sélection des attributs constitue une tache non obligatoire mais très importante pour extraire des sous-ensembles des attributs en préservant les plus significatives et pertinents et en excluant les attributs considérés comme source de bruits dont le but est de faire une classification supervisée des enregistrements de l'ensemble de données NSL-KDD en deux catégories (Normale et Attaque) et de réduire le coût et le temps nécessaire pour l'opération d'apprentissage.

Il existe différentes approches pour faire la sélection des attributs, dans le cadre de notre projet nous avons basé sur une approche de calcul de gain d'informations pour chaque attribut, puis nous avons choisi parmi eux ceux qui ayant des meilleurs gains on suivant les étapes ci-dessous :

a)-Calcul de l'entropie pour chaque attribut :

$$H(x_i) = -\sum_{j=1}^{n} p(x_j|c_1)\log_2 p(x_j|c_1) + p(x_j|c_2)\log_2 p(x_j|c_2)$$

Où :

– c_1, c_2: dénotent les deux classes de classification (normale, attaque).
– x_i: représente un attribut.
– x_j: représente une valeur particulière de l'attribut x.
– n: dénote le nombre de valeurs de l'attribut x.
– **p** : la probabilité.

b)-Calcul de gain pour chaque attribut :

$$Gain = Entropie_E - H(x_i)$$

Où :

$$Entropie_E = \frac{nb_{normal}}{nb_{connexion}}\log_2\left(\frac{nb_{normal}}{nb_{connexion}}\right) + \frac{nb_{attack}}{nb_{connexion}}\log_2\left(\frac{nb_{attack}}{nb_{connexion}}\right)$$

– nb_{normal} : présente le nombre des connexions qui sont classifiées comme normal

– nb_{attack}: présente le nombre des connexions qui sont classifiées comme une tentative d'attaque.

– $nb_{connexion}$: présente le nombre total des connexions dans la base d'apprentissage.

c)-Élimination des attributs ayant un gain inférieur à un seuil donné (0. 5 par exemple).

e)-Utilisation du sous ensemble d'attributs restant (les attributs ayant un gain supérieur ou égal au seuil donné) pour générer le système de classification.

2-2- Apprentissage et établissement du modèle de classification :

Nous avons établis un modèle de détection d'intrusions on se basant sur le perceptron multicouche (MLP) de réseaux de neurones artificiels qui contient un ensemble de neurones répartis en couches et qui utilise comme algorithme d'apprentissage la rétro-propagation du gradient, et lorsque notre classification est binaire où les classes sont 0 et 1, La couche de sortie est comporte un seul neurone prend deux valeurs possibles ;1 : classe normale et 0 : classe attaque qui décrite tous les types d'attaques : DOS, U2R, probe, R2L .

L'opération d'apprentissage commence par l'entrainement de l'algorithme MLP en utilisant, dans notre cas, le sous-ensemble de données KDDTrain +_20Percent comme base d'apprentissage qui contient 25192 individus. Cette opération se termine lorsqu'on arrive à une erreur quadratique moyenne inférieure à un seuil donné ou un nombre prédéfinit d'itérations est attient l'erreur quadratique moyenne obtenue et le nombre d'itérations effectués sont affichés par notre application à la fin de cette phase, ainsi que les poids générés aléatoirement au début de l'apprentissage et les poids optimaux après les mises à jour, ces derniers seront utilisés pour calculer la valeur de neurone de la couche de sortie pour chaque enregistrement dans la phase de test (classification).

2-3-Test et évaluation du modèle généré :

Pour mesurer la qualité de la performance du modèle de détection d'intrusions, le résultat de ce dernier sera comparé avec les données réelles (données marquées). Pour l'ensemble de données NSL-KDD de Test, où toutes les données ont été étiquetées, c'est-à-dire que la classe de chaque instance est connue. Chaque instance est qualifiée de normale ou d'anomalie. Le tableau 4.7, montre les résultats possibles de la nature du résultat du modèle proposé.

2-3-1-La matrice de confusion :

		Classe détectée (prédite)	
		Normale	**Attaque**
Classe réelle	**Normale**	Vrai négatif **TN**(True Negative)	Faux positif **FP** (False Positive
	Attaque	Faux négatif **FN** (False Negative)	Vrai positif **TP** (True Positive

Tableau 4.7 : La matrice de confusion

-Vrai positif (TP) : une attaque correctement détectée par le test .

-Faux positif (FP) : une activité normale détectée comme attaque par le test.

-Vrai négatif (TN) : une activité normale correctement détectée par le test.

-Faux négatif (FN) : une attaque détectée comme activité normale par le test,

2-3-2-Les mesures d'évaluation :

- **La précision :** cette métrique, également relative à chaque catégorie, renseigne sur la probabilité qu'une prédiction d'une catégorie donnée soit correcte.

$$Précision = \left(\frac{\mathbf{TP}}{\mathbf{TP + FP}}\right) \times 100\%$$

- **Le taux de détection (Rappel) :**C'est le rapport entre le nombre d'intrusions correctement détectées et le nombre total d'intrusions. Et décrit par la formule :

$$Rappel = \frac{TP}{FN+TP} \ \mathbf{X100\%}$$

- **Le taux de faux positif (FP) (Le taux des fausses alertes)** : est calculé comme le rapport entre les nombres de trafic normal qui sont incorrectement classés comme intrusions et le nombre total de trafic normal.

$$FP = \frac{FP}{TN+FP} \mathbf{X100\%}$$

- **le taux de réussite (*Accurac*)** : Indique la façon dont la technique de détection est correcte. C'est une métrique qui traduit également le rapport entre les détections correctes et les détections totales obtenues.

$$Accuracy = \frac{\mathbf{TP+TN}}{\mathbf{FP+FN+TP+TN}} \times \mathbf{100}\%$$

Mesure F(Moyenne harmonique): La moyenne harmonique F combine le rappel et la précision en un nombre compris entre 0 et 1.

$$Mesure\_F = \frac{\mathbf{2}}{\frac{\mathbf{1}}{\mathbf{R}} + \frac{\mathbf{1}}{\mathbf{P}}} \times 100\%$$

Où R et P sont respectivement le rappel et la précision.

II. Implémentation et analyse des résultats :

1- Environnement de programmation :

Nous avons choisi l'environnement de programmation IDE NetBeans8 pour Windows afin d'implémenté les deux modules d'apprentissage et de test de notre modèle de détection d'intrusions. Notre choix du langage java a été argumenté par les avantages offerts par la programmation orientée objet

en général. Le choix de Netbeans est fondamental puisqu'il est doté, en standard, de bibliothèques de classes très riches comprenant la gestion des interfaces graphiques (fenêtres, boites de dialogue, contrôles, menus, graphisme), la gestion des exceptions, la variété des types, la gestion des fichiers et au réseau, l'ajout automatique des déclarations d'importation, etc. L'utilisation de ces bibliothèques facilite grandement la tâche du programmeur lors de la construction d'applications complexes. De plus cet IDE est disponible, gratuit et extensible.

Pour l'implémentation nous avons utilisé un micro-ordinateur ayant les caractéristiques techniques représentées dans le tableau suivant:

N°	Matériel	Caractéristiques techniques
1	Processeur	Intel®Core™i3-370Mprocessor
2	Vitesse de processeur	2.40 GHz
3	Mémoire	4.00 Go
4	Système d'exploitation	Windows 7 64 bits

Tableau 4. 8 : Caractéristiques techniques de l'ordinateur utilisé pour l'implémentation

2- Description de l'application :

- **Fenêtre principale :** cette fenêtre est composée de deux menus : menu Classification et menu Exemple_d'Apprentissage.

Figure 4. 2 : Fenêtre principale de l'application

- **Menu Classification :** ce menu composé d'un seul sous menu (Apprentissage et test MLP) comme le montre la Figure 4.3

Figure 4. 3 : Le menu classification

- **Sous menu Apprentissage et test MLP :** Si on choisit le sous menu Apprentissage et test MLP, une fenêtre avec deux onglets s'affichera. Un onglet pour l'apprentissage et l'autre pour le test.

- **L'onglet Apprentissage MLP** : A partir de cette fenêtre nous pouvons entrer tous les paramètres d'apprentissage de réseau de neurones et démarrer la phase d'apprentissage sur une base de données d'entrainement (KDD-Train) pour générer les poids optimaux du réseau de neurone.

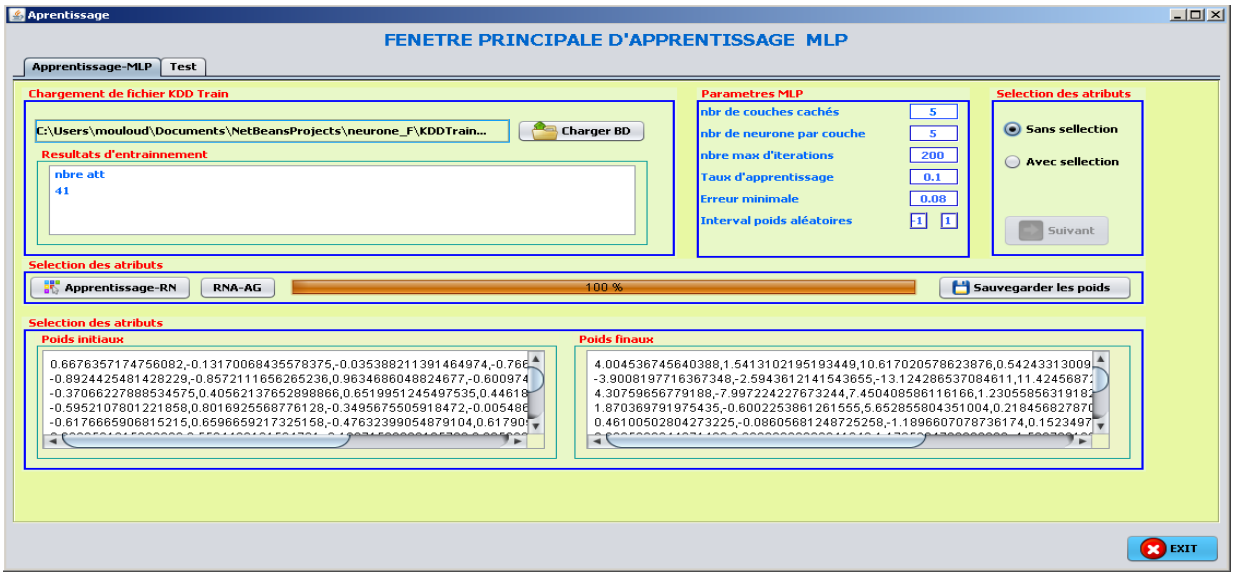


Figure 4. 4 : Onglet apprentissage de MLP

La phase d'apprentissage se termine par l'affichage d'un message de dialogue qui illustre l'erreur moyenne obtenue et le nombre d'itérations.

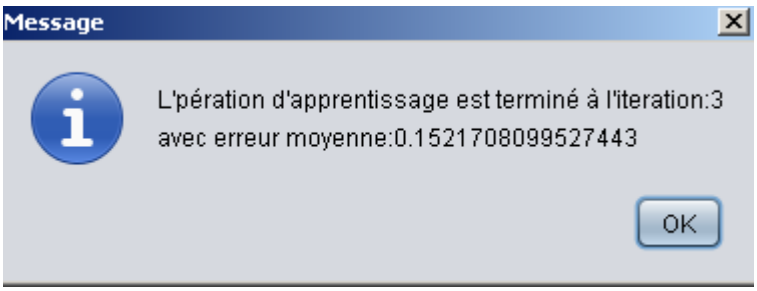


Figure 4. 5 : Message de fin de la phase d'apprentissage.

- **L'onglet Test MLP**: A partir de cette fenêtre on peut évaluer le résultat de la phase d'apprentissage en utilisant les poids générés pendant l'apprentissage et une base de données de test

(KDD-Test) et afficher les résultats d'évaluation (matrice de confusion, le taux de réussite, rappel, précision, etc.).

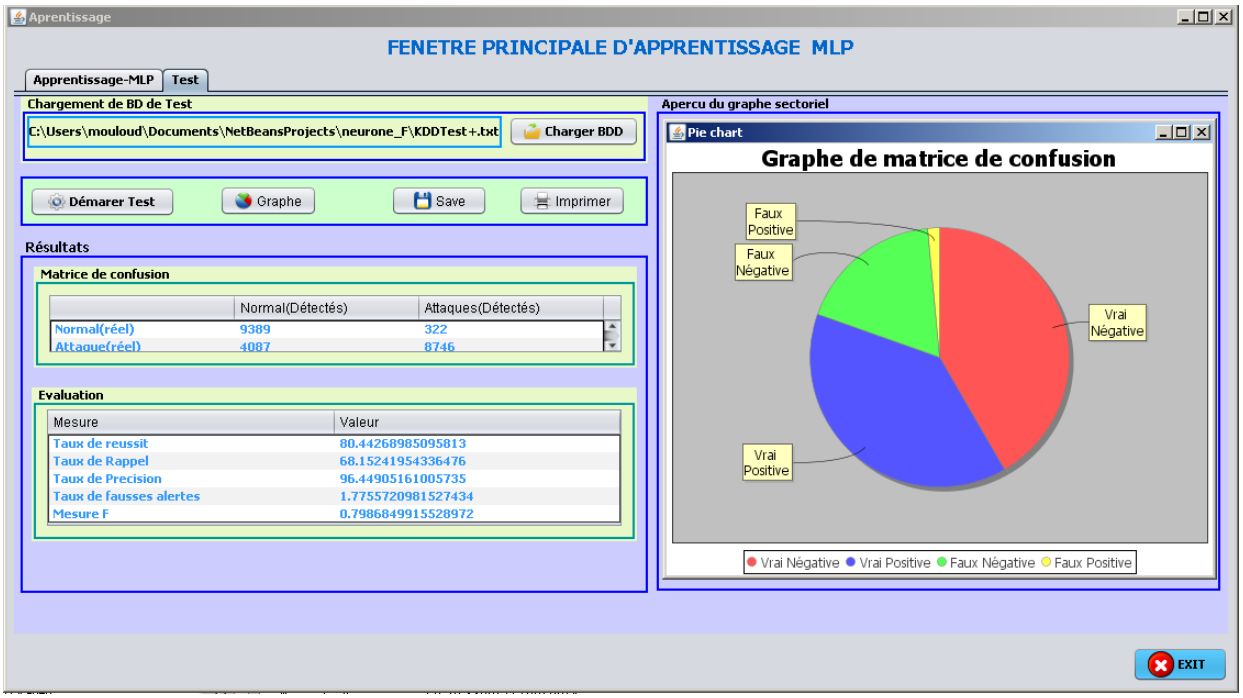


Figure 4. 6 : L'onglet Test MLP.

- **La fenêtre Sélection d'attributs**: l'opération de sélection d'attributs est optionnelle, si cette option est activé nous allons se dirigé vers une autre fenêtre (Figure 4.7) qui nous permet de :

– Calculer le gain d'information pour chaque attribut de la base de données.

– Sélectionner les meilleurs attributs (selon leur gain qui doit être supérieur ou égal à un seuil).

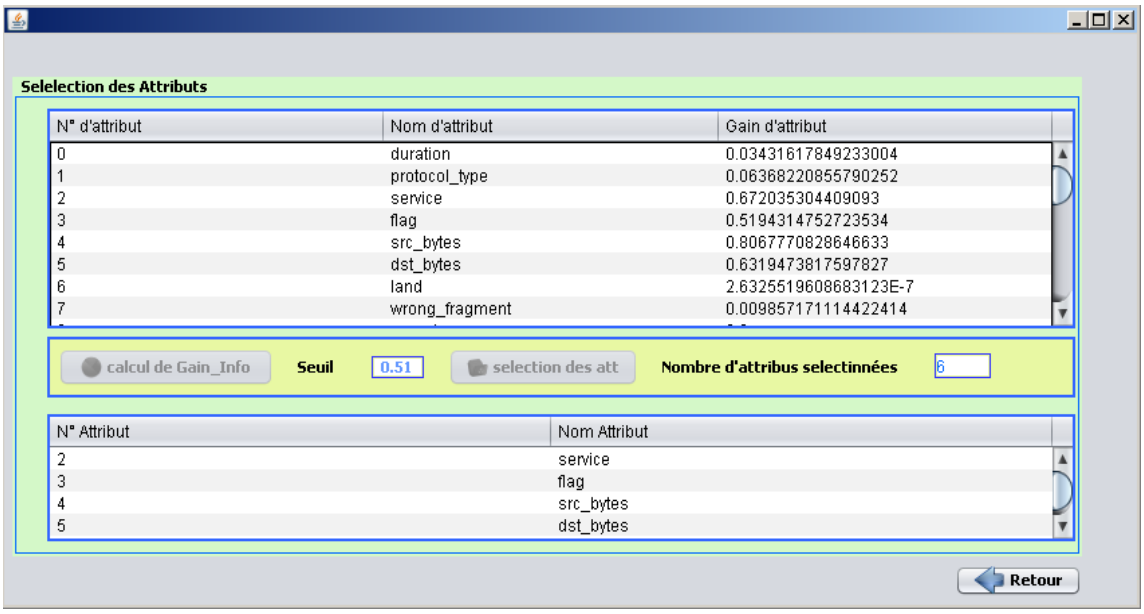


Figure 4. 7 : Fenêtre de sélection d'attributs.

- Une fois les meilleurs attributs sélectionnés, on peut passer directement à la phase d'apprentissage à partir du bouton « Retour » : qui nous dirige de nouveau vers le menu d'apprentissage et test MLP pour faire un apprentissage

- **Menu Exemple d'apprentissage**: Ce menu va nous diriger lorsque en clique sur le sous menu chargement des poids vers une fenêtre dans laquelle on fait l'apprentissage et le test en utilisant des fichiers de poids déjà enregistrés à partir d'une opération d'apprentissage précédente.

Figure 4. 8 : Le menu Exemple d'apprentissage

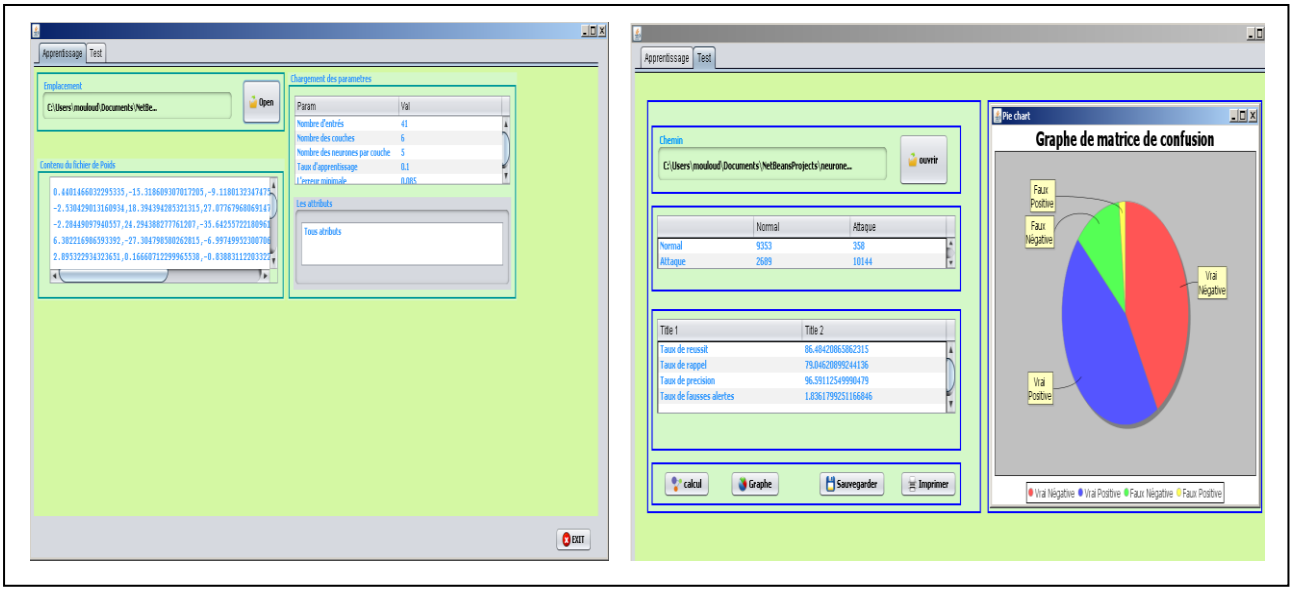


Figure 4. 9 : Chargement et test d'un exemple d'apprentissage déjà enregistré.

- **La fenêtre paramètres d'optimisation génétique:** cette fenêtre s'affiche lorsqu'on clique sur le bouton (Apprentissage Neuro-Génétique), elle sert pour le réglage des paramètres d'algorithme génétique, qui permet de modifier et d'améliorer les poids générés par l'apprentissage MLP, à la fin de cette opération qui peut prendre beaucoup de temps**,** les nouveaux poids obtenus seront utilisés pour le test MLP ou sauvegardés pour un futur test.

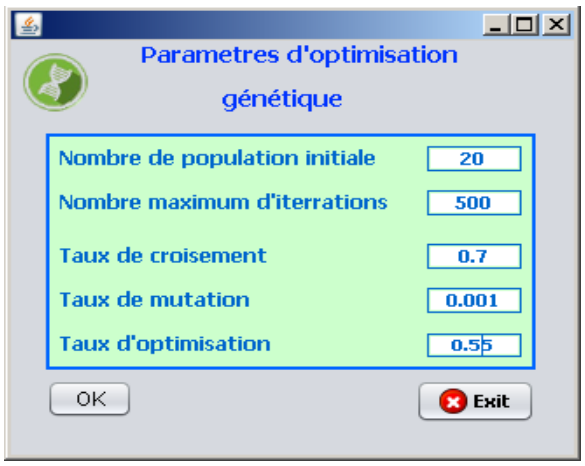


Figure 4. 10 : Fenêtre des paramètres d'optimisation génétique.

3- Test et résultats Expérimentaux :

3-1- Paramètres de test

Nous avons effectué plusieurs tests dans notre étude expérimentale en réglant à chaque fois les paramètres d'apprentissage de réseaux de neurones (taux d'apprentissage, nombres de couches cachés, neurones par couche, les bornes d'intervalle des poids initiaux, nombres d'itérations) dont l'objectif est de trouver les valeurs de ces paramètres qui donnent les meilleurs résultats en termes du taux de réussite (accuracy), de la précision, du taux des fausses alarmes, etc. Pour le nombre de couches cachées, nous avons réalisé le test sur différents valeurs, en commençant par2 couches, puis 3 couches cachées, 4, 5,6 et à chaque fois nous choisissons le nombre de neurones pour constituant la couche caché pour avoir une valeur qui donne le meilleur score.

Le choix des autres paramètres communs dans les différentes expérimentations pour le réseau de neurones et l'algorithme génétique sont indiqués dans le tableau ci-dessous.

Paramètres d'apprentissage		**Paramètres d'optimisation**	
Paramètres	**Valeur**	**Paramètres**	**Valeur**
Pas d'apprentissage	0. 1	Taille de population	20
Fonction de transfert	$1/(1+e^{-x})$	Taux de croisement	0.7
Le nombre d'itérations maximum	10000	Taux de mutation	0.001
Intervalle des poids initiaux	[-1,1]	Le nombre d'itérations maximum	500

Tableau 4. 9 : Paramètres d'apprentissage et d'optimisation

3-2 Résultats obtenus :

Afin de montrer l'importance de l'optimisation des poids de réseau de neurones par les algorithmes génétiques et son influence sur les performances du modèle généré, nous avons effectué, sur les mêmes paramètres d'apprentissage, deux tests séparés. Dans le premier test, nous avons utilisé uniquement les

réseaux de neurones pour générer les poids de ces derniers qui seront utilisés dans l'opération de classification par le modèle de détection d'intrusions. Alors que dans le deuxième test, nous avons introduit les poids résultants dans le premier test du processus d'apprentissage du MLP dans l'algorithme génétique qui permettra d'améliorer et d'optimiser ces poids. Ensuite on refait l'opération de classification tout en comparant les résultats obtenus.

Les résultats obtenus pour les deux tests sont montrés dans les tableaux et les graphes ci-dessous. Les meilleurs résultats obtenus sont représentés dans le tableau suivant :

- **Le premier test** : **avec réseau de neurones :**

Matrice de confusion :

	Nombre de neurones/ Couche	**Réels**	**calculés**	
			Normale	**Attaque**
MLP à 2 couches	4	Normale	9374	337
		Attaque	3400	9433
MLP à 3 couches	5	Normale	9455	256
		Attaque	3632	9201
MLP à 4 couches	5	Normale	9201	510
		Attaque	3635	9198
MLP à 5 couches	5	Normale	9339	372
		Attaque	3128	9705
MLP à 6 couches	5	Normale	9058	653
		Attaque	3370	9463

Tableau 4. 10 : Matrice de confusion de modèle MLP sans algorithme génétique

Mesures d'évaluation :

Nombre de Couches cachées	**Nombre d'attributs**	**Nombre de neurones/ couche**	**Erreur minimal**	**Nombre d'itérations**	**Taux de réussite %**	**Le Rappel %**	**La précision %**	**Les fausses Alarmes %**	**F_score %**
2	41	4	0.099	396	**83.42**	73.50	96.55	1.79	83.46
3		5	0.077	2000	**82.75**	71.69	97.29	1.37	82.55
4		5	0.087	469	**81.61**	71.67	94.74	2.77	81.60
5		5	0.083	1000	**84.47**	75.62	96.30	1.95	84.71
6		5	0.094	300	**82.15**	73.73	93.54	3.52	82.46

Tableau 4. 11 : Evaluation des résultats obtenus sans algorithme génétique.

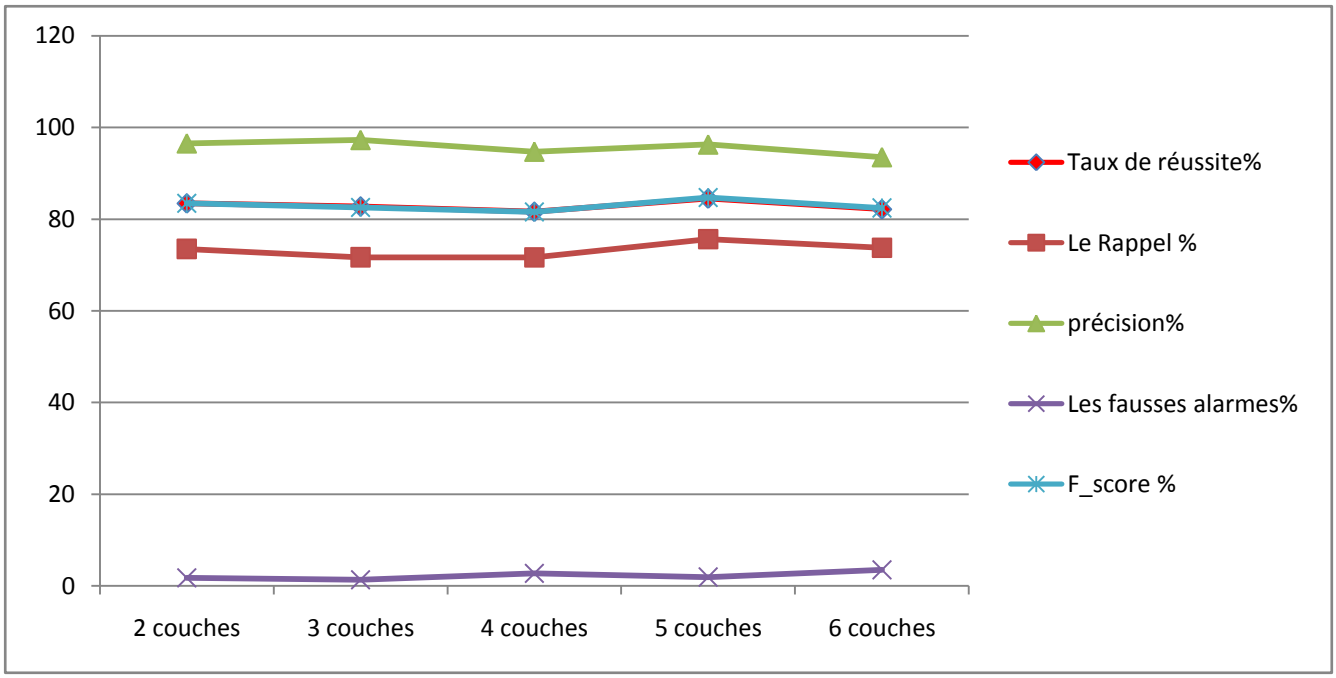


Figure 4.11 : Résultats obtenus sans algorithme génétique

- **Le deuxième test : avec Réseau de neurones et AG** :

Matrice de confusion :

	neurones/ couche		Normale	Attaque
MLP à 2 couches	4	Normale	9300	411
		Attaque	2984	9849
MLP à 3 couches	5	Normale	9330	381
		Attaque	3030	9803
MLP à 4 couches	5	Normale	9233	478
		Attaque	2776	10057
MLP à 5 couches	5	Normale	9072	639
		Attaque	2138	10695
MLP à 6 couches	5	Normale	9353	358
		Attaque	2689	10144

Tableau 4. 12 : Matrice de confusion de modèle MLP avec Réseau de neurones et AG

Mesures d'évaluation :

Nombre de Couches cachées	Nombre d'attributs	Erreur minimal	Nombre d'itérations	Taux de réussite%	Le Rappel %	La précision%	Les fausses Alarmes %	F_score %
2	41	0.086	1	**84.94**	76.74	95.99	2.10	85.29
3		0.077	360	**84.86**	76.38	96.25	1.99	85.18
4		0.087	269	**85.56**	78.36	95.46	2.47	86.06
5		0.098	200	**87.68**	83.33	94.36	3.23	88.50
6		0.094	17	**86.48**	79.04	96.59	1.83	86.94

Tableau 4. 13 : Evaluation des résultats obtenus avec Réseau de neurones et AG.

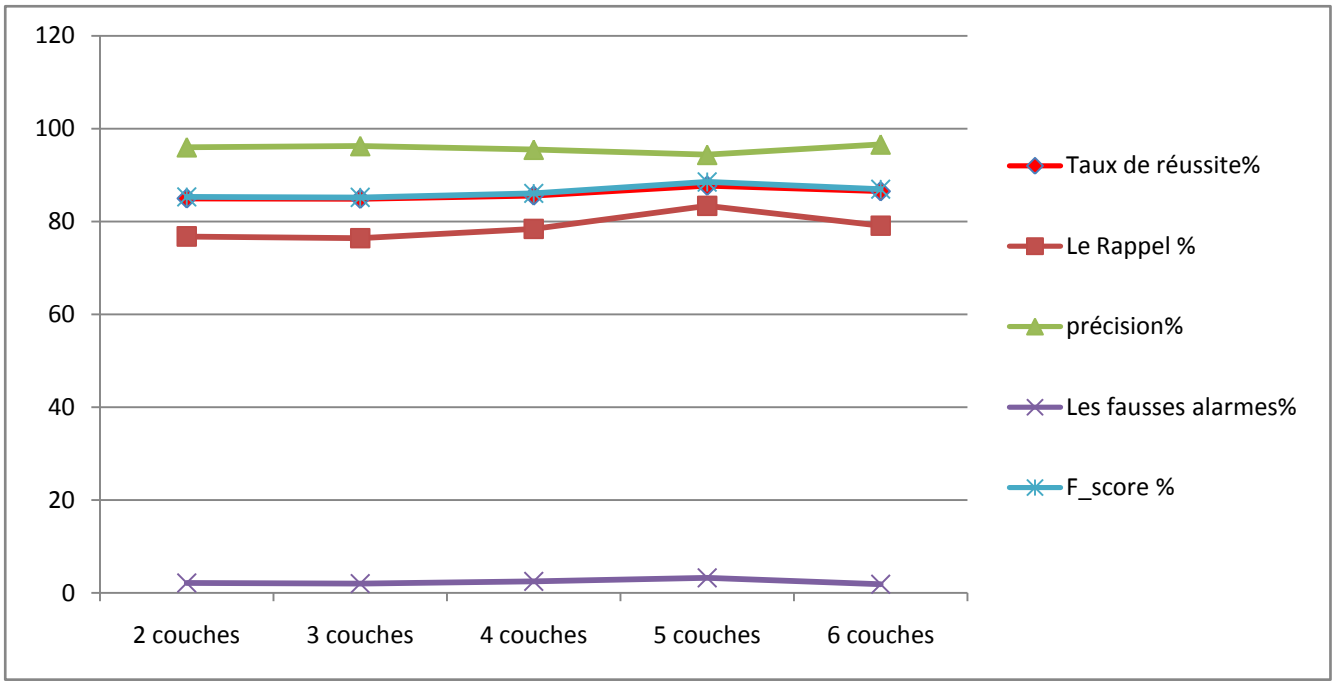


Figure 4.12 : Résultats obtenus avec RNA et algorithme génétique

2-2-3 Analyse et comparaison des résultats:

Les résultats présentés ci-dessus montrent que les performances de notre modèle de détection d'intrusions sont acceptables, il détecte les intrusions, y compris les nouvelles intrusions, avec un taux de réussite (respectivement rappel, précision) arrivant jusqu'à 87.68% (respectivement 83.33%, 97.39%) et un taux de fausses alarmes ne dépassant pas 3.5%. Les résultats montrent également l'importance de l'algorithme génétique comme technique d'optimisation des poids des réseaux de neurones. En appliquant cet algorithme, les performances de notre modèle se sont considérablement améliorées comme le montre le tableau suivant :

Nombre de couches cachés	taux de réussite (accuracy)			Le rappel			La précision		
	Sans A-G	Avec A-G	différence	Sans A-G	Avec A-G	différence	Sans A-G	Avec A-G	différence
2	83.42	84.94	**+1.52%**	73.50	76.74	**+3,24%**	96.55	95.99	**-0,56%**
3	82.75	84.86	**+ 02.9%**	71.69	76.38	**+4.69**	97.29	96.25	**-1.04%**
4	81.61	85.56	**+3.95%**	71.67	78.36	**+6,69%**	94.74	95.46	**+0,72%**
5	84.47	87.68	**+3.21%**	75.62	83.33	**+7,70%**	96.30	94.36	**-1,948%**
6	82.15	86.48	**+4.33%**	73.73	79.04	**+0.31%**	93.54	96.59	**+3.05%**

Tableau 4. 14 : Comparaison des mesures d'évaluation avec et sans A-G

En plus de tous ces diagnostiques et d'après les résultats obtenus, nous avons constaté aussi que le changement du nombre de couches cachées et le nombre des neurones par couche ont une influence sur

les performances du modèle généré. On peut, par exemple, remarquer que les meilleurs résultats sont donnés par le modèle MLP à 5 couches cachées avec 5 neurones par couche cachée.

Pour montrer clairement les améliorations apportées par l'algorithme génétique, nous présentons dans la figure 4.13 un histogramme comparant le taux de réussit du modèle généré avec et sans algorithme génétique.

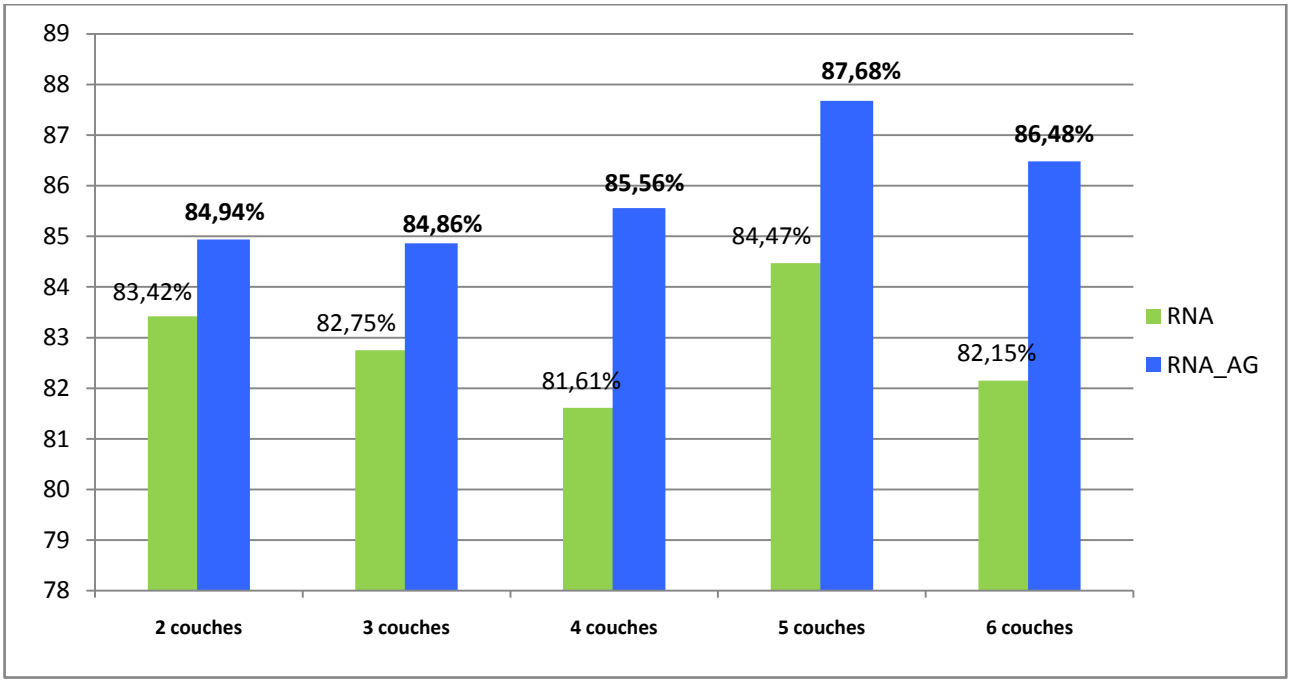


Figure 4.13 : Comparaison des résultats obtenus avec RNA et RNA avec AG

L'histogramme ci-dessus montre clairement l'efficacité de l'algorithme génétique dans l'amélioration du rendement des réseaux de neurones lorsqu'utilisé pour l'optimisation des valeurs des poids des liens entre neurones.

Conclusion :

Nous avons abordé dans ce dernier chapitre la base de données NSL-KDD qui a été utilisée pour l'apprentissage et test du modèle de détection d'intrusions généré basé sur les réseaux de neurones artificiels.

Nous avons aussi expliqué le fonctionnement de l'application implémentée pour mettre en œuvre ce modèle. Ensuite, nous présentons les résultats obtenus après plusieurs valeurs de paramètres du MLP, et les améliorations acquises après la combinaison des réseaux de neurones avec les algorithmes génétiques comme technique d'optimisation.

Les résultats ont prouvé l'efficacité des modèles à base de réseaux de neurones dans le domaine de la détection d'intrusions, plus particulièrement le perceptron multicouches, surtout lorsqu'il est combiné avec l'algorithme génétique. Les résultats ont également montré l'importance des algorithmes génétiques dans l'augmentation des performances de modèle généré.

Conclusion générale

Conclusion générale :

Les attaques informatiques sont en forte hausse ces dernières années et représentent aujourd'hui un risque réel qui menace les réseaux informatiques, les applicatifs et les systèmes d'information des entreprises. En d'autres termes, La découverte périodique et permanente de ces attaques, en temps opportun peut contribuer à les réduire en prenant les moyens de protection nécessaires, Cela nous a dirigés, dans ce mémoire, vers le développement d'un modèle du système de détection d'intrusions comme moyen d'identification des attaques, dans le but d'éviter leurs dommages. Pour réaliser ce modèle nous nous sommes basé sur les réseaux de neurones artificiels, Ces derniers sont utiles pour la simulation de n'importe quel problème difficile à décrire avec des modèles physiques et mathématiques en raison de la capacité des réseaux de neurones d'apprendre par des exemples.

Nous avons aussi utilisé la base de données du benchmark NSL-KDD comme source de données, et nous avons fait un modèle de classification binaire (deux classes : normale et attaque) pour classifier les connexions TCP/IP en deux classes : attaque ou normal, à l'aide de perceptron multicouches(MLP) qui est le modèle de réseaux de neurones le plus approprié pour la classification des données non linéairement séparables.

En effet, le réglage des paramètres du réseau de neurones a une grande importance pour obtenir de bons résultats. Pour cette raison, Nous avons effectué plusieurs expériences afin de choisir les meilleurs paramètres (poids initiaux, pas d'apprentissage, nombre de couches et nombre de neurones dans chaque couche) qui nous donnent les meilleurs résultats en termes de taux de réussite. Nous avons aussi implémenté dans notre application une fonction qui sélectionne les meilleurs attributs en fonction de leurs gains (sélection des attributs ayant un gain supérieur ou égal à un seuil défini par l'utilisateur).Puisque les résultats sont proches avec ou sans sélection des attributs, l'utilisation d'un sous ensemble d'attributs pendant l'apprentissage du MLP nous permet de gagner le temps de calcul.

Nous avons aussi utilisé l'algorithme génétique comme moyen d'optimisation des valeurs des poids générés par le réseau de neurones (MLP), Cela nous a conduits à des améliorations considérables particulièrement en termes de taux de réussite.

Pour conclure, la majorité des objectifs tracés dans de ce travail ont été atteints, mais il reste toujours des perspectives et des améliorations possibles qui peuvent encore être réalisé dans le future, telles que :

- La réalisation d'un modèle de classification multi-classes (Normal, DOS, R2L, U2R et Probe) au lieu de la classification binaire (Normal, Attaque) réalisée dans ce travail.

- En plus de l'emploi des algorithmes génétiques comme technique pour optimiser les poids des réseaux de neurones, élargir ces algorithmes pour la sélection des meilleurs paramètres d'apprentissage (nombres de couches cachés, nombre de neurones par couche, taux d'apprentissage, etc.).
- Aussi de penser à la création d'un modèle capable de capturer le trafic réseau en temps réel sans avoir besoin d'utiliser l'ensemble de données NSL KDD Test.

Annexes

Annexe 1: Les attributs de NSL KDD : [33]

Les détails des attributs sont répertoriés dans les tableaux suivants

N°	Nom de l'attribut	Type	Description
1	duration	Numérique	La durée de connexion
2	protocol_type	Nominal	Protocole utilisé dans la connexion (tcp, udp, icmp)
3	service	Nominal	Service réseau de destination,(http,telnet, ftp_data, etc.)
4	flag	Nominal	Statut de la connexion –Normal ou Erreur (SF, REJ, S0, S1, etc.)
5	src_bytes	Numérique	Nombre d'octets de donnée transférés de la source à la destination (491, etc.)
6	dst_bytes	Numérique	Nombre d'octets de données transférés de destination à la source (0, etc.)
7	land	Binaire	**Si** l'adresse IP de source et destination et le nombre de port sont les mêmes **alors**, *land*=1 **sinon** *land*=0
8	wrong_fragment	Numérique	Nombre total de fragments erronésdans cette connexion
9	urgent	Numérique	Nombre de paquets urgents
10	Hot	Numérique	Nombre d'indicateurs « Hot »
11	num_failed_logins	Numérique	Nombre de tentatives de connexion échouées
12	logged_in	Binaire	**Si** connecté avec succès **alors** *logged_in*=1 **sinon** *logged_in*=0
13	num_compromised	Numérique	Nombre de conditions compromises
14	root_shell	Binaire	1 si le root shell est obtenu, 0autrement
15	su_attempted	Binaire	1 **si** la commande "su root" a été tentée ou utilisée, **sinon** 0
16	num_root	Numérique	Nombre d'accès " root " ou nombre d'opérations effectuées comme racine dans la connexion
17	num_file_creations	Numérique	Nombre d'opérations de création de fichiers
18	num_shells	Numérique	Nombre d'invites du shell
19	num_access_files	Numérique	Nombre d'opérations sur les fichiers de contrôle d'accès
20	num_outbound_cmds	Numérique	Nombre de commandes sortantes dans une session FTP
21	is_host_login	Binaire	1 **si** la connexion appartient à la liste du « hot » (root ou admin);**sinon** 0
22	is_guest_login	Binaire	1 **si** le login est un login «guest »;**sinon** 0
23	count	Numérique	Nombre de connexions vers le même hôte de destination que la connexion en cours dans les deux dernières secondes
24	srv_count	Numérique	Nombre de connexions vers le même service (N° Port) que la connexion en cours dans les deux dernières secondes
25	serror_rate	Numérique	Le pourcentage de connexions qui ont activé le *flag* s0, s1, s2 ou s3,parmi les connexions agrégées dans *count*

26	srv_serror_rate	Numérique	Le pourcentage de connexions qui ont activé le *flag* s0, s1, s2 ou s3,parmi les connexions agrégées dans *srv_count*
27	rerror_rate	Numérique	Le pourcentage de connexions qui ont activé le *flag* REJ, parmi les connexions agrégées dans *count*
28	srv_rerror_rate	Numérique	Le pourcentage de connexions qui ont activé le *flag* REJ, parmi les connexions agrégées dans srv_*count*
29	same_srv_rate	Numérique	Le pourcentage de connexions qui sont au même service, parmi les connexions agrégées dans *count*
30	diff_srv_rate	Numérique	Le pourcentage de connexions qui sont aux différents services, parmi les connexions agrégées dans *count*
31	srv_diff_host_rate	Numérique	Le pourcentage de connexions qui sont à différentes machines de destination, parmi les connexions agrégées dans srv_*count*
32	dst_host_count	Numérique	Nombre de connexions ayant la même adresse IP de l'hôte de destination
33	dst_host_srv_count	Numérique	Nombre de connexions ayant le même numéro de port
34	dst_host_same_srv_rate	Numérique	Le pourcentage de connexions qui sont au même service, parmi les connexions agrégées dans *dst_host_count*
35	dst_host_diff_srv_rate	Numérique	Le pourcentage de connexions qui sont aux différents services, parmi les connexions agrégées dans *dst_host_count*
36	dst_host_same_src_port_rate	Numérique	Le pourcentage de connexions qui sont au même port de source, parmi les connexions agrégées dans*dst_host_srv_count*
37	dst_host_srv_diff_host_rate	Numérique	Le pourcentage de connexions qui sont à différentes machines de destination, parmi les connexions agrégées dans *dst_host_srv_count*
38	dst_host_serror_rate	Numérique	Le pourcentage de connexions qui ont activé le *flag* s0, s1, s2 ou s3,parmi les connexions agrégées dans *dst_host_count*
39	dst_host_srv_serror_rate	Numérique	Le pourcentage de connexions quiont activé le *flag* s0, s1, s2 ou s3, parmi les connexions agrégées dans *dst_host_srv_count*
40	dst_host_rerror_rate	Numérique	Le pourcentage de connexions qui ont activé le *flag* REJ, parmi les connexions agrégées dans *dst_host_count*
41	dst_host_srv_rerror_rate	Numérique	Le pourcentage de connexions qui ont activé le *flag* REJ, parmi les connexions agrégées dans *dst_host_srv_count*

Tableau1 Les 41 attributs de la base NSL-KDD

Liste des Acronymes

A-G Algorithme génétique

ARP Address Resolution Protocol

CPU Central Processing Unit

DNS Domain Name System

DARPA Defense Advanced Research Projects Agency

DOS Denial Of Service

HIDS Host based Intrusion Detection System

IDS Intrusion Detection System

IP Internet Protocol

KDD Knowledge Discovery in Databases

MAC Media Access Control

MLP MultiLayer Perceptron

MIT Massachusetts Institute of Technology

NIDS Network based Intrusion Detection System

NTIC Nouvelles Technologie de l'Information et de Communication

RNA Réseaux de Neurones Artificiels

TCP Transmission Control Protocol

R2LRemote to User

SE Système d'Exploitation

U2R User to Root

Bibliographie & web graphie

Webographie :

[w1] https://www.securiteinfo.com/conseils/choix_ids.shtml:le 27/02/2017

[w2] http://www-igm.univ-mlv.fr/~dr/XPOSE2004/IDS/IDSSnort.html: le 14/03/2017

[w3] http://www.windowsecurity.com/pages/article_p.asp?id=1147:le 17/03/2017

[w4] http://www.grappa.univ-lille3.fr/polys/apprentissage/sortie005.html: le 25/05/2017

[w5] http://magnin.plil.net/spip.php?article47 le 07/06/2017

[W6] https://fr.scribd.com/document/342463237/A-Study-on-NSL-KDD-Dataset-pdf Le 10/06/2017

Références bibliographique :

[1] Gunadiz Safia, Algorithmes d'intelligence artificielle pour la classification d'attaques réseaux a partir de données TCP, université de boumerdes,2011.

[2] M Thibaut Probst : évaluation et analyse des mécanismes de sécurité des réseaux dans les infrastructures virtuelles de cloud computing,2015.

[3] Mme LABED Ines : Proposition d'un système immunitaire artificiel pour la détection d'intrusions, universite de Constantine, 2006.

[4] :Aissaoui Sihem, Apprentissage automatique et sécurité des systèmes d'information :Application :un système de détection d'intrusion basé sur les (SVM),Université d'oran,2008.

[5] Rodrigue Mpyana Mise en place d'un système de sécurité basé sur l'authentification dans un réseau IP. Cas de Mecelco, 2011.

[6] David Powell et Robert Stroud : Conceptual Model and Architecture of MAFTIA. Technical Report Series-University of Newcastle Upon Tyne Computing Science, 2003.

[7]Abdelhalim Zaidi. Recherche et détection des patterns d'attaques dans les réseaux IP _a hauts débits. Réseaux et télécommunications [cs.NI]. Université d'Evry-Val d'Essonne, 2011.

[8] Nathalie Dagorn. D´etection et pr´evention d'intrusion : pr´esentation et limites. [Rapport de recherche], Université de Nancy1, France,2006

[9]SLIMANI Ahmed, Application des systèmes immunitaires artificiels à la détection d'intrusion, USTO-MB : 2011

[10]Cédric Michel, Langage de description d'attaques pour la détection d'intrusions par corrélation d'événements ou d'alertes en environnement réseau hétérogène, Université de Rennes 1,2003.

[11]Yousef Farhaoui, «Evaluation des systèmes de détection et de prévention des intrusions et la conception d'un BIDS », thèse de doctorat, Université Ibn Zohr, 2012.
[12]Philippe Biondi , Architecture expérimentale pour la détection d'intrusions dans un système informatique,2001.
[13] William Stallings, network security essentials: applications and standards fourth edition,2011.
[14]N.Labroche Modélisation du système de reconnaissance chimique des fourmis pour le problem de la classification non-supervisés: application à la mesure d'audience sur internet, 2003.
[15] Nicolas Monmarché ,Algorithmes de fourmis artificielles : applications `a la classification et `a l'optimisation, 2004.

[16] E. Lebarbier, T. Mary-Huard,ClassificationNonSupervisee-AgroParisTech
[17]YOUSSEF FATAICHA, RECHERCHE D'INFORMATION DANS LES IMAGES DE DOCUMENTS, MONTRÉAL, LE 27 DÉCEMBRE 2005.
[18] Patrice Wira, Réseaux de neurones artificiels : Architecture et applications

[19]Claude Touzet, LES RESEAUX DE NEURONES ARTIFICIELS, INTRODUCTION AU CONNEXIONNISME, 2016

[20]Ridha GHAYOULA, Contribution à l'Optimisation de la Synthèse des Antennes Intelligentes par les Réseaux de Neurones,2008]
[21]OTMANI Imene, L'ANALYSE DISCRIMINANTE ET LE PERCEPTRON MULTICOUCHE PRESENTE PAR:2011.

[22] G.Dreyfus, J.-M. Martinez, M. SamuelidesM. B. Gordon, F. Badran, S. Thiria, Apprentissage statistique», Eyrolles, Paris, 2004.
[23]MAHADOUI RAFIK, DIAGNOSTIC INDUSTRIEL PAR NEURO-FLOU-APPLICATION A UN SYSTEME DE PRODUCTION,2008]
[24] Lynda AMIMER , Modélisation et Commande des Systèmes Non Linéaires Fractionnaires par des Réseaux de Neurones Fractionnaires,2015
[25]Thanh Ha Dang, 2007, P103 : Mesures de discrimination et leurs applications en apprentissage inductif, l'Université Paris 6.
[26] Marc Parizeau, Le perceptron multicouche et son algorithme de retro-propagation des erreurs,2004,p1
[27]Philippe Jauffret, Introduction aux réseaux de neurones ,2002

[28]Maria Güell i Pons,Algorithmes d'optimisation,université de lausanne 2009.

[29] Khalaf Alahmed :Système de contrôle de qualité de production :Méthodologie de modélisation, de pilotage et d'optimisation des systèmes de production,université de paulverlaine-metz,2008.

[30] Thomas Vallée*]* et Murat Yıldızo˘glu : Présentation des algorithmes génétiques et de leurs applications en Economie, Université de Nantes, LEA-CIL :2001.

[31] Pierre Borneet all, Les réseaux de neuronesprésentation et applications, , Edition TECHNIP, France 2007.

[32] L.Dhanabal& Dr. S.P. Shantharajah, «A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms», International Journal of Advanced Research in Computer and Communication Engineering Vol. 4, Issue 6, June 2015.

[33] Rebecca Petersen,Data Mining for Network Intrusion Detection,Université de MID SWEDEN,2015

[34] Eric Maiwald, Network Security: A Beginner's Guide, McGraw Hill, 2001.

[35] Lerman, I. &Ngouenet, F. (1995), Algorithmes génétiques séquentiels et parallèles pour une représentation affine des proximités, Rapport de Recherche de l'INRIA Rennes - Projet REPCO 2570, INRIA.

Résumé

Le développement technologique, l'utilisation d'Internet à grande échelle et l'augmentation des moyens de stockage et de l'échange d'information sont contribué à un nombre élevé de cyber-attaques, qui exploitent les failles des systèmes d'information et les points faibles des systèmes de protection contre les intrusions, ce qui rend le processus de sécurisation de ces informations très important. Les spécialistes du domaine de la sécurité informatique s'occupent d'élaborer les outils nécessaires pour assurer la sécurité de l'information en développant de nouvelles technologies basées sur des moyens d'intelligence artificielle afin de détecter les pénétrations non découvertes par des dispositifs ordinaires. Dans ce contexte, nous visons à travers ce travail à réaliser un modèle de détection d'intrusions qui s'appuis sur l'apprentissage du fonctionnement " normal " des informations échangées et de signaler les divergences par rapport à ce fonctionnement de référence comme des attaques .en se basant sur les réseaux de neurones artificiels, l'algorithme génétique et la base de données de référence NSL-KDD pour générer et évaluer le modèle proposé.

***Mots clés** : intrusions, classification, réseaux de neurones, intelligence artificiel, algorithme génétique, NSLKDD.*

Abstract

The large-scale technological, development and use of the Internet and the increase in the means of storage and exchange of information have contributed to a high number of cyber attacks, which exploit the gaps in information systems and The weak points of protection systems against intrusions, which makes the process of securing this information very important. The security specialists are making the necessary tools to ensure information security by developing new technologies based on artificial intelligence to detect the penetrations undiscovered by ordinary devices. In this context, we aim, through this work, to realize an intrusion detection model based on the learning of the "normal" functioning of the exchanged information and to signal the the divergences from this reference functioning as attacks. Based on the artificial neural networks, the genetic algorithm and the NSL-KDD reference database to generate and evaluate the proposed model.

***Keywords:** intrusions, classification, neural networks, artificial intelligence, genetic algorithm, NSLKDD.*

ملخص

لقد أدى التطور التكنولوجي ، استعمال شبكة الانترنت على نطاق واسع و تعدد وسائل تخزين و تبادل المعلومات إلى ارتفاع عدد الهجمات الإلكترونية و عمليات قرصنة البيانات من خلال استغلال ثغرات الأنظمة المعلوماتية و نقاط ضعف أنظمة الحماية من الاختراقات، مما جعل عملية تامين المعلومات تكتسي أهمية كبيرة، ومن اجل مواكبة هذه التطورات عمل المختصين في مجال أمن المعلوماتية على توفير الأدوات و الوسائل اللازمة لتأمين المعلومات بوضع تقنيات جديدة تعتمد على الذكاء الإصطناعي لتحديد الإختراقات الغير مكتشفة بواسطة أجهزة تأمين المعلومات العادية، في هذا الإطار نهدف من خلال هذا العمل إلى انجاز نموذج لكشف الاختراقات يعتمد على فهم طبيعة البيانات العادية و تصنيف الغير معروفة كاختراقات و هذا باستعمال الشبكات العصبية الإصطناعية و الخوا رزمية الجينية ،كما اعتمدنا على قاعدة المعطيات DDK-LSN كمرجع لإنجاز و تقييم النموذج المقترح.

كلمات مفتاحية: اختراقات، تصنيف، شبكات عصبية، ذكاء اصطناعي، خوارزمية جينية، NSL-KDD.