N° d'ordre: 03/2014-D/INF

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE HOUARI BOUMEDIENE

FACULTE : ELECTRONIQUE ET INFORMATIQUE

Thèse

Présenté pour l'obtention du grade de : DOCTEUR

Spécialité : Intelligence artificielle

Par : Melle. MILOUD AOUIDATE Amal

SUJET

REDUCTION A L'AIDE D'UNE APPROCHE BASEE SUR LES KNN, DES ALERTES FAUSSES POSITIVES ISSUES D'UN DETECTEUR D'INTRUSIONS RESEAU

Soutenue publiquement, le 30-04-2014 Devant un jury composé de :

M. Yassine DJOUADI	Professeur, à L'USTHB	Président
M. Ahmed Riadh BABA-ALI	Maître de Conférence A, à L'USTHB	Directeur de thèse
Mme. Dalila BOUGHACI	Maître de Conférence A, à L'USTHB	Examinatrice
Mme. Karima BENATCHBA	Professeur à, L'ESI	Examinatrice
Mme. Fatiha BENBOUZIDE	Professeur à, L'ESI	Examinatrice
Mme. Narhimene BOUSTIA	Maître de Conférence A, à L'USDB	Examinatrice

Résumé

Nous avons proposé, au titre de cette thèse, de contribuer à apporter des éléments de réponse à la problématique de réduction des alertes fausses positives issues d'un système de détection d'intrusions en utilisant une approche basée sur la règle des K-plus proches voisins (KNN), moyennant la conception et l'évaluation de quatre nouveaux algorithmes, fondés sur les concepts de colonies de fourmis et de mémétique, et ce dans la perspective d'améliorer la détection d'intrusions tout en réduisant le taux d'alertes fausses positives.

Partant du constat qu'un système de détection d'intrusions basé sur la règle KNN utilise un ensemble de données d'apprentissage pour la classification dont la précision est subordonnée à la taille dudit ensemble et que cette dernière (la taille) détermine le temps de traitement en fonction de son volume, nous avons exploré l'applicabilité respectivement des concepts de colonies de fourmis et de mémétique suscités à travers deux approches de sélection d'instances.

La première approche consiste à combiner la technique de classification d'alertes KNN avec deux nouveaux algorithmes l'un abrévié Clust ANT-IS, qui est basé sur les principes d'optimisation par colonies de fourmis et le clustering DBSCAN, l'autre un algorithme de sélection d'instances, abrévié ALAIS qui améliore son précédent en étant basé sur l'apprentissage actif. Ces deux algorithmes sont construits sur la base de l'algorithme Ant-IS, qui est un algorithme de sélection d'instances basé sur des principes d'optimisation par colonies de fourmis. Cet algorithme améliore les performances du 1NN standards, en sélectionnant les instances importants permettant de créer un ensemble de formation efficace. Ant-IS a été prouvée efficace dans le domaine de la classification par apprentissage.

La deuxième approche consiste au jumelage d'un nouvel algorithme génétique amélioré abrévié GIS à un nouvel algorithme de recherche locale contrôlée, abrévié CLS. Ce jumelage a permis de créer un algorithme abrévié MCLS, qui permet de renforcer les capacités du classificateur standard KNN dans la détection d'intrusions. Ce qui est nouveau dans cette méthode, est le processus de recherche locale proposée qui guide à la fois la mutation et la sélection de la solution.

Les résultats obtenus, à la suite des tests sur le benchmark KDD'99, se sont avérés satisfaisants et démontrent que les deux approches permettent de réduire effectivement

le taux d'alertes fausses positives de 0.60% (ALAIS) et 0.53% (Clust Ant-IS), concernant l'approche basée sur les colonies de fourmis, et de 1.05% (GIS) et 0% (MCLS), concernant l'approche mémétique, avec des taux de détection respectifs de : 98.64% , 99.38%, 98.02% et 99.87%.

Mots clés *KNN, Optimisation par Colonies de Fourmis, Condensing, Sélection d'instances, Réduction d'ensemble, Algorithme Mémétique, Recherche Locale, Détection d'Intrusions, Sécurité des réseaux, Faux positifs, Apprentissage automatique.*

Abstract

We have proposed in the title of this thesis, to help solve the problem of reducing false positive alerts from a system of intrusion detection using an approach based on the K- nearest neighbors rule (KNN), through the development and evaluation of four new algorithms, based on the concepts of ant colonies and memetics, in the prospect of improving intrusion detection while reducing the rate of false alarms.

Noting that a system of intrusion detection based on KNN rule uses a training set of data for classification, whose accuracy is dependent on the size of the said set, and that this size determines the processing time, we explored the applicability of the concepts respectively of ant colony and memetic through two approaches of instance selection.

The first approach is an aggregation of KNN alerts classification with two new algorithms, one abbreviated Clust ANT-IS, which is based on the principles of ant colony optimization and DBSCAN clustering, the other, an instance selection algorithm, abbreviated ALAIS improves his previous and is based on active learning. These two algorithms are built on the basis of the Ant-IS algorithm, which is an instance selection algorithm, based on the principles of ant colony optimization. This algorithm improves the performance of standard 1NN, selecting instances to create effective training set. Ant-IS has been proven effective in the field of classification.

The second approach involves the pairing of an improved genetic algorithm abbreviated GIS to a new controlled local search algorithm, abbreviated CLS. This twinning has created the MCLS algorithm. The latter, improves the capabilities of standard KNN classifier in intrusion detection. What is new in this method is the proposed process of local search that guide both mutation and selection of the solution.

The results, after tests on KDD'99 benchmark , are very satisfactory and show that both approaches can effectively reduce false positive alert rates of 0.60% (ALAIS)and 0.53% (Clust Ant-IS) for the approach based on ant colonies, and 1.05% (GIS) and 0% (MCLS) for the evolutionary approaches, with respective rates of detection : 98.64%, 99.38%, 98.02% and 99.87%.

Keywords *KNN, Ant Colony Optimization, Condensing, Instance Selection, Set Reduction, Memetic Algorithm, Local Search, Intrusion Detection, Network Security, False positive, Machine Learning.*

ملخص

اقترحنا في هذه الرسالة ، المساعدة في حل مشكلة الحد من التنبيهات الإيجابية الكاذبة من نظام كشف التسلل باستخدام قاعدة تستند إلى نهج K- أقرب الجيران (KNN) ، من خلال تطوير وتقييم أربع خوارزميات جديدة ، استنادا إلى مفاهيم مستعمرات النمل و memetics ، و تحسين كشف التسلل في حين خفض معدل الإنذارات الكاذبة.

إن نظام كشف التسلل بناء على حكم KNN يستند على مجموعة بيانات تدريب للتصنيف الذي دقته تعتمد على حجم هذه المجموعة و هذا الحجم يحدد الوقت اللازم للتجهيز. استكشاف إمكانية تطبيق مفاهيم مستعمرات النمل و memetic أثيرت من خلال نهجين .

النهج الأول هو تجميع مصنف التنبيهات KNN مع خوارزميتين جديدتين ، يختصر احد Clust ANT- IS، ويقوم على مبادئ مستعمرات النمل و تكتلات DBSCAN ، والآخر خوارزمية تختصر ALAIS يحسن الخوارزمية السابقة و يقوم على التعلم النشط . هذه الخوارزميات تحسن أداء NN1 القياسية ، واختيار الحالات لخلق مجموعة تدريب فعالة .

النهج الثاني ينطوي على الاقتران لتحسين خوارزمية وراثية جديدة يختصر GIS و خوارزمية بحث محلي مرافقة جديدة مختصرة CLS . وقد خلقت هذا التوأمة الخوارزمية MCLS . هذه الأخيرة تحسن قدرات المصنف KNN القياسية في كشف التسلل . الجديد في هذا الأسلوب هو العملية المقترحة للبحث المحلي.

النتائج، بعد اختبارات على KDD'99 مرضية للغاية، و تبين أن كلا النهجين يمكن أن تقلل بشكل فعال معدلات التنبيه الكاذبة ب 0.60 ٪ (ALAIS) و 0.53 ٪ (Clust ANT- IS) للنهج المعتمد على مستعمرات النمل ، و 1.05 ٪ (GIS) و 0 ٪ (MCLS) للنهج التطوري ، مع معدلات الكشف: 98.64 ٪ ، 99.38 ٪ ، 98.02 ٪ و 99.87 ٪ .

كلمات البحث: KNN ، مستعمرات النمل، التكتيف ، اختيار المثال، تحديد مجموعة، خوارزميةMemetic ، البحث المحلي ، كشف التسلل ، أمن الشبكات ، كاذبة إيجابية ، التعلم الآلي .

Remerciements

En premier lieu je remercie ALLAH de m'avoir aidé et donné la force et la volonté de réaliser ce modeste travail.

Ensuite, mes remerciements s'adressent à M. Baba-Ali A. R., mon directeur de thèse, pour le soutien constant et les nombreux conseils qu'il m'a prodigués tout au long de ces trois années de thèse. Je lui suis particulièrement reconnaissante de m'avoir laissé une grande liberté scientifique, ce qui m'a permis d'acquérir un apprentissage idéal et privilégié de la recherche.

Je remercie très chaleureusement M. Abou Baker Hamdi-Cherif (Arabie Saoudite), M. José Ramon Cano (Espagne), M. Francisco Herrera (Espagne) et M. Fabrizio Angiulli (Italie), qui ont généreusement répondu à toutes mes sollicitations et qui m'ont fait profiter de leur savoir. Leur gentillesse et leurs remarques constructives ont grandement influencé mon travail.

Je tiens à exprimer ma reconnaissance envers les membres du jury pour l'intérêt qu'ils ont porté à ce mémoire en acceptant d'en être examinateurs. J'adresse mes vifs remerciements à M. Djouadi Yassine, Professeur à l'université des sciences et Technologie Houari Boumediene (USTHB) pour l'honneur qu'il me fait de présider ce jury. Je remercie également Mme. Benbouzide F. (ESI) et Mme. Boughaci D. (USTHB) pour avoir accepté la charge d'évaluer, en qualité d'expertes, les travaux réalisés.

Egalement, je tiens à remercier ma famille pour leurs encouragements, leur aide et leur grande patience avec moi.

Enfin, je dois à mes camarades en PG et à l'ensemble de mes collègues une reconnaissance que je veux à la hauteur du soutient et encouragements, qui n'ont jamais cessés de témoigner à mon égard.

Enfin, je tiens à remercier tous ceux qui ont contribué de près ou de loin à l'aboutissement de ce travail.

Table des matières

Table des figures

Chapitre 1

Introduction

Les failles de sécurité d'un système informatique peuvent avoir de graves conséquences sur une organisation, qui peuvent aller de la perte financière aux atteintes à la réputation de cette dernière jusqu'à la faillite.

Les attaques informatiques ne concernent pas uniquement les entreprises et les sociétés mais touchent de plus en plus les gouvernements et les structures sensibles des pays. A tel point que le Pentagone a clairement émit des menaces, en affirmant qu'il envisagera "toutes les options possibles" si les Etats-Unis étaient victimes d'une cyber-attaque [139].

Devant la gravité de la situation Joseph Henrotin [55] a même évoqué un "Pearl Harbor électronique" , car il pense qu'il est maintenant possible de tout frapper en même temps.

Par conséquent, les gouvernements et entreprises mettent en avant une ligne de défense surarmée, à première vue, pour protéger les systèmes des éventuelles attaques. Cette ligne comprend des systèmes d'authentification des utilisateurs, des scanners d'emails et de vulnérabilité, des antis-virus,des pare-feux, etc. Il est à noter que le dispositif qui represente la derniére barrière de protection reste bien le système de détection d'intrusions.

1.1 Motivation

De nos jours, la sécurité de l'information sur les réseaux informatiques devient un domaine de plus en plus sensible avec la croissance exponentielle du nombre de machines sur les réseaux. Pour préserver l'état de sécurité, il est essentiel d'être attentif au comportement des informations entrantes et en déduire si ces informations sont normales ou non. Autrement dit, il est crucial de surveiller le trafic entrant pour vérifier si ce dernier est normal ou malicieux.

Compte tenu du fait que les données intrusives sont nombreuses et ont des formes complexes, leur distinction de celles normales est assez difficile. En d'autres termes, les limites des comportements normaux et anormaux ne sont pas toujours précisément

définies.

Le problème de la détection d'intrusions a été largement étudié dans le domaine de la sécurité informatique [87] [58] [28] [32], et a béneficié de beaucoup d'attention.

Les systèmes de détection d'intrusions (IDS, de l'anglais Intrusion Detection System) [59] [176] [32] [50] sont des systèmes qui visent à détecter les intrusions. En d'autres termes, ce sont des mécanismes permettant de contrer l'ensemble de mesures qui tentent de compromettre l'intégrité, la confidentialité ou la disponibilité d'une ressource informatique [87]. Les IDS ont été initialement introduits par Anderson [150] et plus tard formalisés par Denning [53].

Les IDS sont considérés comme un élément essentiel de l'architecture de sécurité des informations. Etant la dernière ligne de défense, les IDS doivent examiner attentivement les données d'audit et déclencher des alarmes lorsque des activités suspectes sont détectées.

L'efficacité des IDS dépend de leur sensibilité. La sensibilité d'un IDs est liée au seuil de détection de ce dernier. Plus un IDS est sensible, plus haut est le niveau de sécurité qu'il peut offrir. D'autre part, le réglage de la sensibilité à un niveau trop bas rendrait l'IDS incapable de détecter certaines activités suspectes. Evidemment, puisque le premier devoir de l'IDS est de détecter les intrusions qui s'infiltrent à travers les contrôles d'accès, il n'est pas souhaitable pour l'IDS de manquer une quelconque attaque. Par conséquent, une sensibilité élevée est plus préférable qu'une sensibilité basse. Cependant, la haute sensibilité conduit à un problème auquel presque tous les IDS se heurtent [168] [38] [100], [132], qui est le problème d'***alertes fausses positives***. Comme la plupart des responsables de sécurité ont tendance à maintenir un taux de détection assez élevé pour les IDS, les IDS deviennent plus sensibles et donc moins tolérants aux anomalies. En conséquence, les IDS vont générer beaucoup d'alarmes même si les activités en cours d'examen sont en réalité légales. Ces alarmes constituent l'un des problèmes les plus importants rencontrés par la détection d'intrusions de nos jours [140].

Une alerte fausse positive survient quand un IDS détecte une activité suspecte, alors qu'une analyse détaillée montre que le trafic réseau est normal. Le problème principal créé par les alertes fausses positives, est que ces alertes peuvent facilement cacher les alertes réelles. En fait, il a été estimé que jusqu'à 99% des alertes signalées par les IDS sont de fausses alertes [168] [100][38]. Le défi consiste à réduire le nombre de faux positifs et d'améliorer la pertinence des alertes.

1.1.1 Alertes fausses positives

Organiser et traiter les logs enregistrés et les alertes générées par les capteurs de sécurité tels que les IDS n'est pas une tâche facile. La plupart des organisations considèrent la gestion de ces alertes comme un problème majeur. Étant donné le fait que

ces capteurs sont indépendants, ils vont générer chacun un nombre important d'alertes. Ces alertes générées sont envoyées à l'analyste afin de déterminer la nature des intrusions. L'analyse des alertes vise à réduire le taux de fausses alertes sans dégradation du taux de détection des attaques réelles.

Le terme faux positif décrit la situation où une activité normale est signalée comme malveillante par l'IDS [147]. L'IDS, dans ce cas, alerte l'utilisateur de la présence d'une activité suspecte, alors qu'une analyse approfondie du système prouve que le trafic réseau est normal.

Les raisons d'alertes fausses positives

La construction d'un IDS efficace qui génère seulement un nombre réduit de faux positifs est une tâche ardue. Les raisons de cela ont été citées dans [157] et sont :

- **Spécificité des signatures de détection :** D'apres [154] l'écriture de signatures d'attaques décrivant les modèles intrusifs pour les IDS basés-signature est une tâche très difficile. Dans certains cas, arriver à avoir un modèle de signature qui n'est ni trop spécifique (dans ce cas l'IDS ne sera pas en mesure de saisir toutes les attaques et leurs variantes) ni trop général (sinon l'IDS va classer les actions légitimes comme des intrusions), est une tache délicate.
- **Dépendance envers l'environnement :** Les actions qui sont normales dans certains milieux peuvent être malveillantes dans d'autres [35]. Par exemple, effectuer un balayage du réseau est malicieux, si c'est l'ordinateur d'un simple utilisateur qui a été autorisé à le faire. L'IDS déployé avec une sortie standard de configuration va probablement identifier de nombreuses activités normales comme malveillantes.
- **Limitations du temps d'exécution :** Les auteurs de [160]ont précisé que dans de nombreux cas une intrusion ne diffère que légèrement des activités normales, que parfois même, seulement le contexte dans lequel s'effectue l'activité détermine si elle est intrusive. Toutefois, en raison des exigences sévères en temps réel, les IDS ne peuvent pas analyser le contexte de toutes les activités.

Le problème causé par les alertes fausses positives

Les chercheurs et les praticiens ont signalé que le nombre d'alarmes générées par les IDS pouvait atteindre plusieurs centaines de milliers par jour, mais la plupart sont de fausses alarmes [102] [94]. Beaucoup d'IDS ont même été rendus inopérants par les alarmes déclenchées, parce que les machines ne sont pas assez puissantes pour traiter un aussi grand volume de données [147].

Le traitement des alarmes est une tâche très fastidieuse pour les analystes de sécurité. Demander à un agent de regarder chaque alarme entrante tous les jours peut être considéré comme une perte de temps. En outre, il est impossible pour le personnel

d'examiner toutes les alarmes attentivement, par conséquent certaines attaques peuvent être ignorées.

Outre la lourde charge de travail ajoutée aux responsables de sécurité, l'effet le plus dangereux lié au grand volume de fausses alarmes reste le fait que les responsables de sécurité sont tentés de négliger les réelles activités suspectes.

Lorsque les responsables sont avertis par l'IDS de la présence d'intrusions à chaque instant et que l'analyse de ces alarmes montre qu'elles sont fausses, ces responsables ont tendance à penser que les prochaines alarmes entrantes seront fausses et les ignorent trop facilement en négligeant les analyses approfondies. Finalement peu à peu les responsables prêtent de moins en moins attention aux nouvelles alarmes. Si tel est le cas, lorsque de vraies intrusions se produisent et que des alarmes réelles sont générées, les responsables peuvent les considérer comme fausses alarmes sans enquête détaillée. Ce qui fait que les IDS deviennent inutiles sans qu'ils aient perdu leurs fonctionnalités.

En conclusion, le problème principal que les alertes fausses positives créent, est qu'elles peuvent facilement noyer les alertes légitimes. Une règle simple causant des alertes fausses positives peut créer des milliers d'alertes dans un court laps de temps. Comme dit précédemment, dans un jour de travail, l'administrateur réseau ne peut analyser qu'un petit nombre d'alertes, donc la présence de fausses positives peut masquer les alertes légitimes.

1.2 Problèmatique

Le développement d'une technique de détection d'intrusion efficace qui génère un nombre minimum d'alertes fausses positives est une tâche complexe. Le principal problème créé par les alertes fausses positives, c'est qu'elles peuvent facilement cacher des alertes réelles. Ce problème survient, en fait, de la mauvaise identification d'un élément entrant. Une des solutions à cela, serait de faire appel à un classifieur d'alertes.

1.2.1 Limitation du problème

La règle de classification des K plus proches voisins (KNN) [44] est une méthode de classification puissante qui permet la classification d'une instance inconnue en utilisant un ensemble d'instances classées. Le but de cet algorithme est de classer une nouvelle instance selon les classes des instances de la base d'apprentissage. Ce classificateur se base sur le principe de voisinage afin de détecter la classe la plus fréquente parmi les k voisins de l'instance inconnue. **Le calcul d'un sous-ensemble cohérent de formation avec une cardinalité minimale pour la règle KNN s'avère être un problème NP-difficile [74]**, car un ensemble d'entrainement trop grand pourrait réduire significativement les performances du classifieur KNN en augmentant considérablement le temps de calcul de ce dernier. Par contre, un ensemble d'entrai-

nement trop petit pourrait détériorer les performances de classification du KNN en présentant une base d'apprentissage trop pauvre.

1.3 Résolution du problème

Le problème de la taille de l'ensemble d'entrainement utilisé par le KNN peut être résolu en utilisant des ensembles de données de taille réduite [186].

La réduction de la taille d'un ensemble de données peut se faire grâce au processus de sélection d'instances [180] [195], qui permet de sélectionner les données pertinentes à partir d'un grand ensemble de données, pour créer un ensemble réduit.

1.4 Hypothèses

Cette étude vise à prouver que la sélection d'instances est efficace pour réduire le nombre de fausses alarmes à travers la classification des alertes issues d'un IDS en utilisant le classifieur KNN. Nous avons choisi deux algorithmes évolutionnaires, sur lesquels nous nous sommes basés pour la création de nos propositions. La premiére technique choisie est ***l'optimisation par colonies de fourmis***, qui, à notre connaissance, n'avait jamais été utilisée jusqu'à ce jour, pour résoudre le problème de sélection d'instances. La seconde technique est ***le concept memetique***. Donc nos hypothèses sont que l'optimisation par colonies de fourmis et le concept memetique pourraient être utilisés pour améliorer le filtrage des alertes fausses positives issues d'un IDS basé sur l'algorithme KNN.

1.5 Objectifs

Bien que la fouille de données et l'apprentissage machine (machine learning) ont été utilisés pour la détection d'intrusions dans de nombreuses études, leur application pour filtrer les fausses alarmes de l'IDS est encore un domaine relativement nouveau. Nous avons mené cette recherche afin de proposer une méthode basée sur le classifieur KNN permettant de réduire le nombre d'alertes fausses positives issues d'un IDS.

L'approche proposée devra, par conséquent, remédier au probléme de réduction de la taille de l'ensemble d'apprentissage servant de base au KNN, tout en améliorant, ou au moins en maintenant, les pérformances de ce classifieur. Pour concrétiser cela nous avons délimité nos objectifs en les points suivants :

1. Etudier les travaux traitant de l'utilisation de la sélection d'instances pour la réduction d'ensembles ;
2. Etudier les travaux considérant la détection d'intrusions et la réduction de fausses alarmes basés sur les KNN ;

3. Suggérer une nouvelle approche de réduction de fausses alarmes qui peut améliorer le taux de réduction sans dégradation du taux de classification ;
4. Vérifier et tester l'approche proposée en utilisant des benchmarks standards ;

1.6 Contributions

Les idées principales couvrant nos contributions pour ce travail ont été publiées dans les revues et conférences suivantes :

- IJACSA : Amal Miloud-Aouidate, Ahmed Riadh Baba-Ali : Survey of Nearest Neighbor Condensing Techniques. International Journal of Advanced Computer Science and Applications. 2011. 02 :59-64.
- ICONIP : Amal Miloud-Aouidate, Ahmed Riadh Baba-Ali : A Hybrid KNN-Ant Colony Optimization Algorithm for Prototype Selection. ICONIP (3) 2012 : 307-314.
- CSE : Amal Miloud-Aouidate, Ahmed Riadh Baba-Ali : Ant Colony Prototype Reduction Algorithm for kNN Classification. CSE 2012 : 289-294.
- IJAMC : Amal Miloud-Aouidate, Ahmed Riadh Baba-Ali : An Efficient Ant Colony Instance Selection Algorithm for KNN Classification,has been accepted for publication in the International Journal of Applied Metaheuristic Computing 4(3),(Mai 2013).
- IJMHEUR : Amal Miloud-Aouidate, Ahmed Riadh Baba-Ali :IDS False Alarm Reduction Using a KNN-Memetic Algorithm. has been accepted for publication in the International Journal of Metaheuristics,(Juin 2013).
- IJAMC : Amal Miloud-Aouidate, Ahmed Riadh Baba-Ali : An improved Ant-IS algorithm for Intrusion Detection. Has been accepted for publication in the International Journal of Applied Metaheuristic Computing, 5(1), (September 2013).

Nos travaux se sont soldés par la proposition de quatres approches finales : Clust Ant-IS, ALAIS, GIS et MCLS, et cela suivant le parcourt détaillé ci-après :

Clust Ant-IS et ALAIS

- Nous avons développé Ant-KNN, un algorithme basé sur trois étapes permettant la réduction de l'ensemble d'apprentissage du classifieur 1NN. Cet algorithme fait appel à l'optimisation par colonies de fourmis pour le traitement des données, à une approche heuristique pour la réduction des données, et enfin, à une stratégie de filtrage de bruit pour finaliser la réduction.

  Nous avons étudié la complexité temporelle lors de l'exécution de ladite approche. Cette étude a montré que malgré les résultats satisfaisants présentés par Ant-KNN, son application à la réduction de l'ensemble Darpa'99 s'avérait inadaptée au vu de la taille de cette base de données.
- Nous avons ensuite proposé l'algorithme Ant-IS, un algorithme de sélection d'ins-

tances basé sur les principes d'optimisation par colonies de fourmis.
Cet algorithme propose une complexité temporelle meilleure que celle offerte par son précédent. Cependant, cette complexité reste insuffisante pour une exécution rapide sur l'ensemble darpa'99. Aussi, nous avons étudié les ensembles résultants de la réduction, et nous avons constaté que ces ensembles pouvaient encore être réduits.

– Après cela, nous avons proposé de fixer le nombre exact de fourmis nécessaires pour l'application du Ant-IS ainsi que leurs emplacements en nous inspirant des qualités de l'algorithme de clustering DBSCAN. Nous avons proposé de faire appel à une phase rapide de clustering, telle une étape de prétraitement, afin de désigner les centres et le nombre de clusters dans l'ensemble à réduire. Nous avons nommé l'algorithme issu de cette proposition Clust Ant-IS.
– Finalement, nous avons proposé l'algorithme ALAIS basé sur l'amélioration de l'algorithme Ant-IS à travers l'introduction de l'apprentissage actif (Active Learning), ainsi qu'une phase d'initialisation à travers la technique de clustering DBSCAN.

GIS et MCLS

– Nous avons fait appel un algorithme génétique pour la réduction de l'ensemble d'apprentissage du classifieur 1NN.
Nous avons analysé les résultats obtenus par cet algorithme, et nous avons constaté que ces résultats pouvaient encore être améliorés.
– Noua avons aussi développé une recherche locale guidée par le résultat.
– Finalement, nous avons développé MCLS, un algorithme memetique pour la réduction de l'ensemble d'apprentissage du classifieur 1NN. MCLS est basé sur les concepts génétiques ainsi que ladite recherche locale.

1.7 Organisation de la thèse

Cette thèse s'articule en deux parties. La première est un état de l'art des problèmes de détection d'intrusions et de sélection d'instances et des méthodes de résolution associées.

Dans le second chapitre, après un préambule historique sur l'évolution des systémes de détection d'intrusions et quelques définitions liées au domaine de la détection d'intrusions, nous introduisons les principales techniques de gestion d'alertes ainsi qu'une brève comparaison de ces dernières ;

Le chapitre 3 présente la règle KNN, ses atouts ainsi que le problème de sélection d'instances. Il expose aussi les différentes techniques de résolution appliquées au dit problème.

La seconde partie décrit les contributions de cette thèse, c'est-à-dire la conception de nouvelles méthodes de sélection d'instances permettant d'améliorer la détection

d'intrusions à travers KNN et la réduction d'alertes fausses positives, et est décomposée en trois chapitres :

Le chapitre 4 présente nos deux premières propositions, qui sont basées sur les concepts de colonies de fourmis. Ces dernières permettent de sélectionner les instances pertinentes pour la construction d'un ensemble d'apprentissage réduit efficace.

Le chapitre 5 décrit notre troisième proposition, qui est un algorithme génétique et son amélioration à travers un algorithme mémétique incluant une nouvelle recherche locale. Ces deux propositions se basent sur leurs fonctions objectifs afin de sélectionner les instances nécessaires à la construction d'un ensemble d'apprentissage réduit permettant d'atteindre les objectifs susmentionnés.

Première partie

Etat de l'art

Chapitre 2

Détection d'intrusions

2.1 Introduction

La nécessité de protéger les ressources informatiques se fait plus pressante de jour en jour. Afin d'arriver à assurer une protéction maximale, de nombreuses technologies plus performantes les unes que les autres ont été développées. Nous nous sommes interessés aux systèmes de détection d'intrusions (IDS).

Un système de détection d'intrusions se compose principalement de capteurs qui analysent les activités au sein du système d'information ou du réseau et qui génèrent des alarmes en cas de détection d'activités suspectes [113].

La recherche dans le domaine des système de détection d'intrusions a débuté en 1980 par les travaux de James Anderson [151] qui a souhaité améliorer les équipements d'audit et les capacités de surveillance des systèmes de détection d'intrusions. Le premier modèle de systèmes de détection d'intrusions générique a été proposé par le Dr Dorothy Denning en 1987 [53]. Ce modèle de détection d'intrusions est indépendant de tout système et environnement d'application [53].

Depuis, plusieurs travaux et recherches ont été effectués sur cet axe [32].

2.2 Vulnérabilité des infrastructures actuelles

Historiquement, les entreprises et les gouvernements ont été réticents à révéler des informations concernant les attaques sur leurs systèmes de peur de perdre la confiance publique, ou de peur que d'autres attaquants exploitent la même vulnérabilité ou une vulnérabilité similaire. Aussi, jusqu'à une époque très récente, les données spécifiques et détaillées sur les attaques n'étaient pas disponibles. Aujourd'hui encore les entreprises restent réticentes à dévoiler ce type de données, ce qui ne facilite pas le travail autour des IDS.

Par conséquent, obtenir des données précises au sujet des attaques est un réel défi à cause de cette dissimulation et de la difficulté de mesurer la gravité des alertes. Toutefois, des évaluations statistiques sur la situation des attaques ont été faites afin

de refléter la criticité de la situation.

L'une des attaques les plus spéctaculaires et les plus médiatisées est l'attaque DDoS contre le site web de Wikileaks. Cette attaque a été décrite comme etant le début de la guerre de données.

Les attaques visent de plus en plus les points de forces et les infrastructures sensibles, où près de 40 % des cyber-attaques sur des infrastructures sensibles aux USA ont porté sur des sites liées à l'énergie [187]. En 2012, l'industrie du gaz naturel, aux USA, a fait face à une série d'attaques sur ses pipelines, attaques qui ont contraint le département de la sécurité intérieure à émettre 3 alertes de niveau Amber, le second plus haut niveau d'alerte.

En Novembre 2010, la centrale de Bushehr et le centre de recherches de Natanz (Iran) ont subit des attaques virales visant à perturber leur programme nucléaire [117], ce virus, Stuxnet, a touché 30000 ordinateurs du gouvernement Iranien dans cette attaque.

Ces attaques ont pris une tournure géostratégique, où elles sont désormais considérées comme armes et moyens d'offensifs. Le meilleur exemple à cela serait l'investissement de 110 millions de dollars par la Maison Blanche pour les attaques cybernétiques contre l'Iran [93].

L'emploi offensif des technologies informatiques n'est pas nouveau. D'après les faits divulgués, un virus du nom de Morris avait attaqué en 1988 plus de 6000 ordinateurs du réseau internet naissant, arpanet, et, en 2003 le système de surveillance de la centrale nucléaire américaine de Davice-Besse (Ohio) s'est vu totalement paralysé par une attaque du virus Slammer, qui l'a mis hors service pendant un jour entier. D'autres attaques moins connues, mais réelles, ont également ciblé les systèmes de gestion du trafic ferroviaire et celui de distribution d'électricité et d'eau potable. Les bourses de Londres et de Moscou ainsi que le commissariat à l'énergie atomique Français n'ont pas été épargnés par ce fléau.

En plus d'être un moyen offensif, ces attaques representent aussi des moyens de protestation et de réplique

En réaction aux attaques lancées par les Etats Unis et Israel contre l'Iran, celui-ci s'est infiltré, dans les structures pétrolières et gazières de l'Arabie saoudite et celles du Qatar [149] [93]. En août 2012, la compagnie saoudienne, Aramco, et la compagnie qatarie, RasGaz, ont été attaquées par les hackers iraniens.

Le gouvernement israélien a été la cible d'une cyber-attaque massive marquée par des millions de tentatives de piratage de sites internet d'institutions publiques depuis le début de l'offensive sur Gaza [125]. Et cela afin de protester contre les raids israéliens contre les groupes armés de la bande de Gaza.

En 2007 alors qu'une vive tension diplomatique opposait l'Estonie à la Russie, les principaux sites du gouvernement Estonien se sont vus neutralisés ou défigurés

par des portraits d'Hitler, ainsi qu'un nombre important de banques et d'operateurs téléphoniques. Et lors de l'offensive russe sur la Géorgie en 2008, le site de la présidence Georgienne a été mis hors service pendant plusieurs semaines, ainsi que divers sites médiatiques de ce pays.

2.3 Detection d'intrusions

Le traitement de la sécurité informatique et la protection des données et des ressources informatiques sont souvent associés aux trois propriétés suivantes (communément appelées triade CID, CIA en anglais) [4] :

2.3.1 Confidentialité

La confidentialité concérne la prévention de toute divulgation, intentionnelle ou non intentionnelle, non autorisée de données. Rappelons, que généralement une telle violation est irréversible et ne peut se limiter facilement.

TABLE 2.1 – Confidentialité

Menaces	Contre mesures
Ecoute du réseau - Interne - Externe	Cryptographie (chiffrement) - Des données - Des communications
Vol de fichiers - Données - Mots de passe	Contrôle d'accès - Logique (mot de passe) - Physique (biométrie)
Espionnage	Classification des actifs
Ingénierie sociale	Formation du personnel

2.3.2 Intégrité

L'intégrité concérne la prévention des modifications non autorisées intentionnelles ou non intentionnelles de données. Notons que généralement l'intégrité peut être restaurée, par exemple, à partir d'autres sources telles que des copies de sauvegarde. Cependant, ce processus peut être coûteux, lent et n'est pas toujours complet.

TABLE 2.2 – Integrité

Menaces	Contre-mesures
Attaques malicieuses - Vers, virus - Bombes logiques	Cryptographie - Signature, authentification Systèmes de détection - Antivirus, systèmes de détection d'intrusion (IDS) Politique de sauvegarde
Désinformation	
Erreurs humaines	

2.3.3 Disponibilité

La disponibilité concérne la prévention de la retenue non autorisée des ressources informatiques à la source. Un exemple de disponibilité serait le déni de service (DoS), dans lequel l'attaquant bloque les ressources informatiques afin que les utilisateurs autorisés ne puissent pas les exploiter. En d'autres termes, la propriété de disponibilité permet d'assurer les fonctions offertes sans interruption, délai ou dégradation, au moment même où la sollicitation en est faite.

TABLE 2.3 – Disponibilité

Menaces	Contre-mesures
Attaques malicieuses - Denis de services - SPAM	Pare-feu Systèmes de détection d'intrusions Classification Formation des administrateurs Maintenance préventive, dispositifs de recouvrement
Attaques accidentelles - Le "Slashdot effect"	
Pannes	

2.4 Etude des systèmes de sécurité des réseaux

Un système de sécurité informatique est un système capable d'identifier et d'éradiquer, si besoin est, une attaque(ou intrusion) informatique. Les systèmes de sécurité ne sont pas fiables à 100% et ont besoin d'une continuelle mise à jour du fait de cette évolution technologique qui engendre une innovation au niveau malwares (logiciels malveillants).

Nous citons, ci-après, les différents systèmes de sécurité informatique existant tout en décrivant leur comportement face aux différentes attaques pour lesquels ils ont été conçus.

2.4.1 Authentification forte

Dans un réseau informatique, on a souvent besoin de s'assurer que les informations du réseau sont accessibles par la bonne personne et en cas d'intrusion, trouver le point d'origine. Une authentification forte permet de garantir cela. Une authentification forte est une procédure qui requiert l'utilisation d'au moins trois facteurs d'authentification qui sont : l'authentifieur (clé ou badges), le secret (mot de passe) et la biométrie (empreinte rétinienne ou digitale). Il existe à nos jours trois familles technologiques pour l'authentification forte : le One Time Password (OTP), le certificat numérique et la biométrie.

2.4.2 Chiffrement

Le chiffrement est un procédé rendant incompréhensible un document à toute personne qui n'a pas la clé de déchiffrement. Il existe deux systèmes de chiffrement : symétrique (quand il utilise la même clé pour chiffrer et déchiffrer) et asymétrique (quand il utilise des clés différentes).

2.4.3 Pare-feu

Un pare-feu est un système de sécurité informatique permettant de faire respecter la politique de sécurité du réseau informatique, celle-ci définissant quels sont les types de communication autorisés sur ce réseau. Il mesure la prévention des applications et des paquets. La première génération des pare-feu étaient dits " stateless " ou sans état. Il regarde chaque paquet indépendamment des autres et le compare à une liste de règles préconfigurées. La deuxième génération est le pare-feu à états ou " statefull ", il vérifie la conformité des paquets à une connexion en cours. La troisième génération est le pare-feu applicatif, il vérifie la complète conformité du paquet à un protocole attendu. Par exemple, il permet de vérifier que seul un paquet HTTP passe par le port TCP 80. La quatrième génération est le pare-feu identifiant, il réalise l'identification des connexions passant à travers le filtre IP. La cinquième génération de pare-feu est le pare-feu personnel. Les pare-feux personnels, généralement installés sur une machine de travail, agissent comme un pare-feu à états. Il s'agit en fait de nouveaux antivirus qui intègrent des pare-feux.

2.4.4 Système de détection d'intrusion(IDS)

Un système de détection d'intrusion est un mécanisme destiné à repérer des activités anormales ou suspectes sur la cible analysée (réseau ou hôte). Il permet d'avoir une connaissance sur les tentatives réussies comme celles échouées des intrusions. Ce système réputé pour être plus fiable qu'un pare-feu sera analysé en détail tout au long de ce chapitre, car il représente l'équipement permettant de présenter des réponses aux attaques ciblées.

En résumé, nous pouvons dire que trouver un système de sécurité infaillible est utopique mais le plus important est de savoir gérer ceux existants afin d'en assurer une évolutivité adéquate en fonction des menaces naissantes qui ne cessent de croître d'année en année.

2.5 Systèmes de détection d'intrusions

Nous trouvons dans la littérature plusieurs définitions pour les IDS, cependant nous allons en présenter celles qui nous ont semblées les plus claires :

2.5.1 Définition

Heady et al. [165] ont proposé une définition de l'IDS allant de l'intrusion au système de détection d'intrusions, et cela comme suit :

Intrusion : Une intrusion est n'importe quel ensemble d'actions essayant de compromettre l'intégrité, la confidentialité ou la disponibilité d'une ressource informatique.

Détection d'intrusions : C'est le problème de l'identification des actions qui essayent de compromettre l'intégrité, la confidentialité ou la disponibilité d'une ressource informatique.

Système de détection d'intrusions : C'est une combinaison de logiciel et de matériel qui essaie de réaliser la détection d'intrusions.

D'aprés Kim [95] un système de détection d'intrusions peut se définir comme un système automatisé dont le rôle est la détection des intrusions dans un système informatique tout en examinant les audits de sécurité fournis par le système d'exploitation ou bien les outils de contrôle du réseau. Son but principal est la détection des utilisations non autorisées, les mauvaises utilisations et les abus dans un système informatique par les utilisateurs internes et externes.

Le SANS Institute [167] définit un système de détection d'intrusions comme étant comparé à une alarme de cambriolage. Par exemple, le système de serrure dans une voiture protège la voiture contre le vol. Mais si quelqu'un casse le système de serrure et essaie de voler la voiture, c'est l'alarme anti cambriolage qui détecte que la serrure a été cassée et alerte le propriétaire en donnant une alarme. Le système de détection d'intrusions d'une manière similaire complète la sécurité du pare feux. Le pare feux protège un système contre des attaques malveillantes et le système de détection d'intrusions détecterait si quelqu'un tente de passer le pare feux et d'acceder au coté sûr du système, et alerte le gestionnaire au cas où il y ait infraction dans la sécurité.

2.5.2 Architecture classique d'un IDS

Un système de détection d'intrusions classique est constitué principalement de trois composants [50] : le capteur, l'analyseur et le gestionnaire.

La figure 2.1 illustre l'interaction de ces trois composants : Le capteur récolte et procure les informations sur l'évolution de l'état du système ; L'analyseur détermine la présence d'activités malveillantes dans les événements issus du capteur ; Et le gestionnaire est chargé de présenter à l'operateur de sécurité du système (SSO, de l'anglais System Security Officer) ou à l'analyste de détection d'intrusions (IDA, de l'anglais Intrusion Detection Analyst) les alarmes produites par le capteur.

Par conséquent, l'analyseur est le dispositif responsable de la classification de l'activité et de la distinction des anomalies par rapport au flux normal.

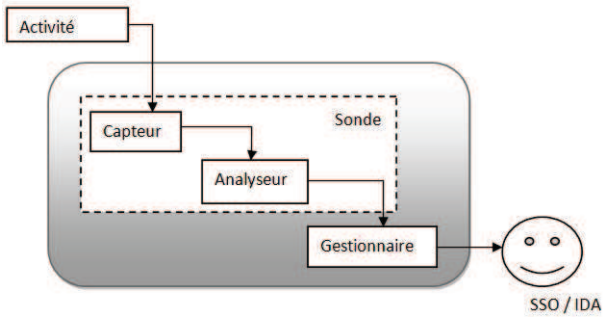

FIGURE 2.1 – Architecture standard d'un système de détection d'intrusions [152]

2.5.3 Fonctionnement d'un IDS

Le fonctionnement général d'un IDS peut être schématisé par la figure 2.2. L'IDS observe le système et collecte les informations d'audit qui sont généralement stockées avant leur traitement.

Le composant de traitement utilise des données de référence et de configuration afin de traiter les données d'audit et générer des alertes.

En faisant ce traitement, le composant de traitement peut stocker temporairement des données (contexte, informations de session, etc.) nécessaires à un traitement ultérieur.

Enfin, les alertes en sortie sont traitées soit par un analyste humain : responsable de sécurité du système (SSO) ou l'analyste de détection d'intrusions (IDA), ou un système de réponses automatiques.

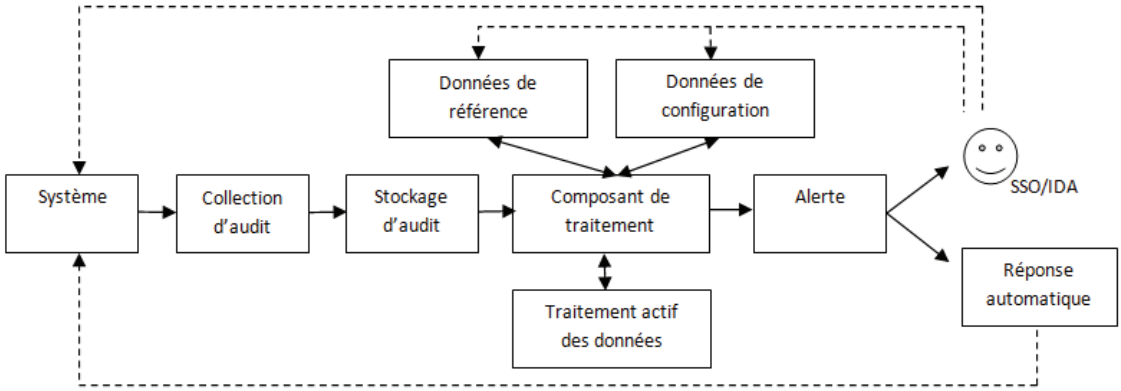

FIGURE 2.2 – Fonctionnement standard d'un système de détection d'intrusions [33]

2.5.4 Types d'IDS

Les IDS disposent de deux approches différentes afin de déceler les intrusions. Suivant ces approches nous pouvons classer les IDS en deux catégories :

1. les IDS à base de signature : Généralement, les IDS réseaux se basent sur un ensemble de signatures qui représentent chacune le profil d'une attaque. Une approche à base de signature consiste à rechercher dans un flux réseau les empreintes d'attaques connues, à l'instar des antivirus.

   Une signature est habituellement définie comme une séquence d'événements et de conditions relatant une tentative d'intrusion. La reconnaissance est alors basée sur le concept de "pattern matching".

   Si une attaque est détectée, une alarme peut être remontée si l'IDS est en mode actif, sinon, l'IDS se contente d'archiver cette attaque.

2. Les IDS comportementaux : Les IDS comportementaux ont pour principale fonction la détection d'anomalies. Leur déploiement nécessite une phase d'apprentissage pendant laquelle l'outil va apprendre le comportement "normal" des flux applicatifs présents sur son réseau. Ainsi, chaque flux et son comportement habituel doivent être déclarés. L'IDS se chargera d'émettre une alarme, si un flux anormal est détecté, et ne pourra bien entendu, spécifier la criticité de l'éventuelle attaque. Les IDS comportementaux sont apparus bien plus tard que les IDS à signature et ne bénéficient pas encore de leur maturité. Ainsi, l'utilisation de tels IDS peut s'avérer délicate dans le sens où les alarmes remontées pourraient contenir une quantité importante de fausses alertes.

2.5.5 Familles d'IDS

Un système de détection d'intrusion, comme cité précédemment, est un mécanisme destiné à repérer des activités suspectes sur un réseau ou un hôte.

Suivant leur localisation et leurs sources d'information, trois familles d'IDS sont usuellement distinguées :

- Les NIDS (NetworkBased Intrusion Detection System), qui surveillent l'état de la sécurité au niveau du réseau.
- Les HIDS (HostBased Intrusion Detection System), qui surveillent l'état de la sécurité au niveau des hôtes.
- Les IDS hybrides, qui utilisent les NIDS et les HIDS pour avoir des alertes plus pertinentes.

Les HIDS sont particulièrement efficaces pour déterminer si un hôte est contaminé et les NIDS permettent de surveiller l'ensemble d'un réseau.

A. Les IDS réseau

Il s'agit d'une plate-forme indépendante qui identifie les intrusions en examinant le trafic réseau et surveille plusieurs hôtes. Dans un NIDS, les capteurs sont situés aux points d'étranglement dans le réseau à surveiller, souvent dans la zone démilitarisée

(DMZ) ou aux frontières du réseau. Les capteurs captent tout le trafic réseau et analysent le contenu des paquets individuels pour une éventuelle détection du trafic malveillant.

Le rôle principal d'un IDS réseau consiste en l'analyse et l'interprétation des paquets circulants sur le réseau qu'il surveille. L'IDS réseau fait appel aux détecteurs afin d'analyser le trafic et si nécessaire envoyer une alerte.

Un IDS réseau agit sur les trames réseau à tous les niveaux. En disséquant les paquets et en inspectant les protocoles, cet IDS est capable de détecter les paquets malveillants conçus pour outrepasser un pare-feu et de chercher des signes d'attaque à différents endroits du réseau. Quelques exemples de NIDS : Net Range, NFR, Snort, DTK et ISS Real Secure.

Le NIDS surveille le trafic en des points choisis sur un réseau ou un ensemble interconnecté de réseaux. Le NIDS examine paquet par paquet le trafic en temps réel, pour tenter de détecter les modèles d'intrusions. Le NIDS peut examiner le protocole au niveau réseau, transport et/ou au niveau application. Une installation typique de NIDS comprend un certain nombre de capteurs pour la surveillance de trafic des paquets, un ou plusieurs serveurs pour les fonctions de gestion et une ou plusieurs consoles de gestion pour l'interface avec l'homme. L'analyse des modèles de trafic pour détecter les intrusions peut être faite soit au niveau du capteur, soit au niveau du serveur de gestion, ou dans une combinaison des deux.

Les IDS réseau présentent beaucoup d'avantages. Leurs capteurs sont bien sécurisés puisqu'ils se contentent d'observer le trafic et sont souvent furtifs (c.à.d. ils agissent de manière invisible). Les IDS réseau permettent, aussi, une détection facile pour certains types d'attaques (scans) grâce aux signatures. Cependant, les IDS réseau, et afin qu'ils soient efficaces, doivent être bien positionnés. Aussi, il est essentiel que ces IDS fonctionnent de manière cryptée. Enfin les IDS réseau ne permettent nullement de voir l'impact d'une attaque.

Types de capteurs réseau :

Les capteurs peuvent être déployés selon l'un des deux modes : actif ou passif.

Un capteur actif est inséré dans un segment du réseau de sorte que le trafic de ce segment doit passer par le capteur. On peut obtenir un capteur actif en combinant un capteur NIDS logique avec un autre périphérique réseau, tel qu'un pare-feu ou un commutateur. La raison principale de l'utilisation des capteurs actifs et de permettre le blocage d'une attaque lorsqu'elle est détectée. Dans ce cas, le dispositif effectue la détection d'intrusions et les fonctions de prévention.

Les capteurs passifs sont par contre les plus généralement utilisés.

Du point de vue circulation, le capteur passif est plus efficace que le capteur actif, car il n'ajoute pas l'étape supplémentaire de traitement qui retarde le paquet. L'interface réseau du capteur passif est une carte réseau qui ne possède pas d'adresse IP, ce qui lui

permet d'être invisible sur le réseau et donc de recueillir les informations sans aucune interaction avec le protocole réseau.

B. Les IDS basés hôte

Les systèmes de détection d'intrusions basés sur l'hôte analysent exclusivement l'information concernant cet hôte. Ces IDS sont généralement très précis sur les variétés d'attaques, puisqu'ils ne contrôlent que les activités concernant un hôte. Les IDS basés sur l'hôte utilisent deux types de sources pour fournir une information sur l'activité : les logs et les traces d'audit du système d'exploitation. Chacun a ses avantages : les traces d'audit sont plus précises, détaillées et fournissent une meilleure information ; les logs, qui ne fournissent que l'information essentielle, sont plus petits et peuvent être mieux analysés en raison de leur taille. Il n'existe pas de solution unique HIDS couvrant l'ensemble des besoins, mais les solutions existantes couvrent chacune un champ d'activité spécifique, comme l'analyse de logs système et applicatifs, la vérification de l'intégrité des systèmes de fichiers, l'analyse du trafic réseau en direction/provenance de l'hôte, le contrôle d'accès aux appels système, l'activité sur les ports réseau, etc.

Un HIDS se compose d'un agent sur un hôte qui identifie les intrusions par des analyses d'appels système , les journaux d'application, des modifications du système de fichiers et les etats d'activités de l'hôte. Dans un HIDS, les capteurs se composent généralement d'un agent logiciel. Un exemple d'un HIDS est OSSEC.

Les systèmes de détection d'intrusions basés sur l'hôte présentent certains atouts, comme la possibilité de constater l'impact d'une attaque et permettent, donc, une meilleure réaction. Ces IDS permettent, aussi, de détecter des attaques dans un trafic chiffré (impossible avec un IDS réseau), ainsi que d'observer avec précision les activités sur l'hôte. Néanmoins, les IDS basés sur l'hôte présentent certaines insuffisances, parmi lesquelles nous pouvons citer la difficulté de détection des scans ; leur vulnérabilité aux attaques de type DoS ; leur consommation importante de ressources CPU, et enfin la contrainte imposée par l'analyse des traces d'audit du système en raison de la taille de ces dernières.

C. Les IDS Hybrides

Malgré leurs dissimilitudes les HIDS et les NIDS se complétent l'un l'autre.

Un HIDS permet une détection plus facile des attaques de type cheval de Troie qu'un NIDS. Il permet aussi de déceler les attaques contenues dans le trafic crypté qui sont indétectables par un NIDS. Ce dernier par contre permet d'éviter l'arrivée de la majorité des attaques jusqu'à l'hôte.

Ainsi, nous pouvons remarquer que la combinaison des deux IDS permet d'éradiquer presque toutes les intrusions, cette alliance est ce qui est appelé un IDS Hybride.

Les IDS hybrides sont basés sur une architecture distribuée, où chaque composant

unifie son format d'envoi d'alertes (typiquement IDMEF "Intrusion Detection Message Exchange Format") permettant à des composants divers de communiquer et d'extraire des alertes plus pertinentes. Pour ces caractéristiques, les systèmes de détection d'intrusions hybrides atteignent des taux de détection élevés ainsi qu'un faible taux de faux positifs.

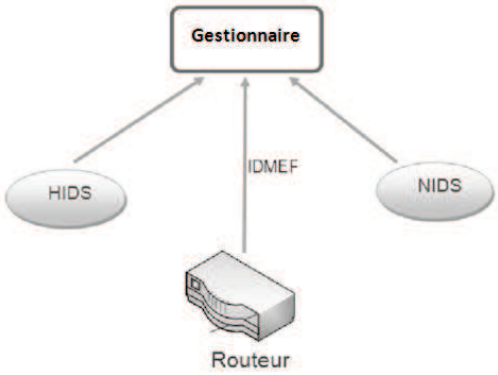

FIGURE 2.3 – Système de Détection d'Intrusions Hybride

2.5.6 Evolution des IDS

Les systèmes de détection d'intrusions ont beaucoup évolué depuis le modèle proposé par Denning. Plusieurs techniques ont été introduites afin d'améliorer leurs performances et rendre leur détection plus précise. Dans ce qui suit nous allons présenter un ensemble de systèmes des plus connus.

IDES

Comme cité précedemment, le premier modéle d'IDS "IDES" (Intrusion Detection Expert System) a été réalisé par Dorothy et Peter Denning. Ce modéle fait appel à des techniques statistiques afin de caractériser un comportement anormal, et se base sur un ensemble de règles pour détecter les violations. Dorothy a émi, plus tard [53], qu'il était possible de détecter des intrusions dans un système informatique indépendament de ce dernier, des applications installées et de sa vulnérabilité, et cela à travers un modéle reformulant les comportements des utilisateurs par les systèmes de détection d'intrusion.

Haystack

"Haystack" [63] représente une variante du modéle proposé par Dorothy Denning. Ce prototype [63] a été développé pour la détection des intrusions dans un environnement multi utilisateurs du Air Force Computer System (il s'agissait de la plate forme standard du Air force à l'époque). Pour détecter les intrusions le système utilise deux méthodes de détection : la détection d'anomalies et la détection à base de signature.

MIDAS

MIDAS [134] a été développé par le centre national de la sécurité (NCSC), en collaboration avec le laboratoire informatique international SRI, pour fournir une détection d'intrusions pour les réseaux mainframe. Les auteurs se sont inspirés des travaux antérieurs de Denning et Al. Ce modéle est construit autour de l'idée de détection d'intrusions heuristique et comporte une base de règles appelée P-Best écrite en Lisp.

Aprés MIDAS les recherches se sont dirigées vers l'utilisation des systèmes experts dans l'écriture des règles [81], ce qui permet une mise à jour dynamique de ces règles.

Discovery

L'IDS Discovery regroupe une approche hybride statistique et expert [196]. Cet IDS analyse les fichiers journaux d'une application. Discovery nécessite des méthodes statistiques pour la détection de profiles et se base sur un système expert pour la détection d'intrusions.

Wisdom & Sense

Wisdom & Sense ou WRS [184] est un système de détection d'anomalies, qui a été développé entre 1984 et 1989. Il est intéressant de noter que WRS à sa création n'a pas été destiné à être appliqué à la sécurité informatique, mais plutôt à un problème lié au contrôle de matières nucléaires [32]. Ce système est unique dans son approche de détection d'anomalies : il étudie l'historique des données d'audit et produit un arbre de règles décrivant le comportement normal, ces règles sont ensuite introduites dans un système expert qui évalue les données de vérification récentes et déclenche une alerte lorsque les règles indiquent un comportement anormal.

NSM

NSM [183], [37] a été le premier système à utiliser directement le trafic réseau en tant que source de données d'audit. Cet IDS écoute passivement tout le trafic qui passe par un réseau local de diffusion et en déduit le comportement intrusif. L'idée de cette approche découle de l'observation que plusieurs autres systèmes de détection d'intrusions essayaient d'atténuer les problèmes des différentes plateformes en suivant leurs pistes d'audit.

Hyperview

Hyperview [88] est un système de détection d'intrusions fondé sur deux composantes principales, la première consiste en un système expert qui surveille les pistes de vérification de signes d'intrusions connues de la communauté de sécurité. La seconde est une composante de réseau de neurones qui apprend le comportement d'un utilisateur de

manière adaptive et envoie une alarme quand la piste d'audit s'écarte du comportement appris auparavent.

DIDS

DIDS [177] est un système de détection d'intrusions distribué incorporant Haystack et NSM, vues précédemment. DIDS est constitué de trois composantes principales. Sur chaque hôte, un moniteur effectue la détection locale d'intrusion, résume les résultats et les communique au directeur DIDS. En outre, chaque segment broadcast du LAN posséde ses propres moniteurs qui surveillent le trafic dans ce LAN et rapportent au directeur du DIDS leurs observations. Finalement, le directeur DIDS, qui est un système constitué du responsable de communication et d'un système expert, analyse les observations des moniteurs et en communique les résultats au SSO.

USTAT

USTAT [92], [106] est un IDS basé sur l'approche d'analyse de transition d'état. Cette analyse estime que l'ordinateur au départ existe dans un état sûr et passe à un état compromis suite à un certain nombre d'intrusions (modélisées comme des transitions d'état) . USTAT lit les spécifications des transitions d'état nécessaires pour compléter une intrusion issue du SSO, et évalue ensuite une vérification de la piste à l'aide du cahier des charges. La piste d'audit produite par l'ordinateur est donc utilisée comme source d'informations sur les transitions d'état du système.

IDIOT

IDIOT [110], [111], [108], [133], [109] est un système qui a été développé à COAST (maintenant le Centre pour l'éducation et la recherche en sûreté de l'information et de la sécurité (CERIAS), http ://www.cerias.purdue.edu). Le principe de base derrière IDIOT est d'utiliser les réseaux de Pétri colorés pour une détection d'intrusions à base de signatures. Les auteurs suggèrent qu'une approche en couches doit être utilisée lors de l'application de techniques à base de signatures (pattern matching) pour résoudre le problème de détection d'intrusions. Les auteurs arguent, du fait que des nombreuses techniques de filtrage disponibles, les réseaux de Petri colorés (CP-nets) seraient la meilleure technique à appliquer, car elle ne souffre pas de certaines lacunes communes dans d'autres techniques, qui ne permettent pas l'assortiment conditionnel des modèles, et ne se prêtent pas à une représentation graphique.

NIDES

NIDES [46] [51] est le successeur du projet IDES. NIDES suit les mêmes principes généraux que les versions ultérieures de l'IDES : il a une base forte de détection d'anomalies, complétée par un composant de système expert à base de signatures. Ce dernier

est mis en œuvre en utilisant un système expert P-BEST. Le système NIDES est fortement modulaire, avec des interfaces bien définies entre les composants, construit sur une architecture client-serveur. Il est centralisé dans la mesure où l'analyse s'exécute sur un hôte spécifique et recueille des données de différents hôtes à travers un réseau informatique.

GrIDS

GrIDS [175] permet de construire des graphiques représentant l'activité du réseau pour faciliter la détection d'intrusions, particulièrement dans les grands réseaux. Les graphiques codifient les hôtes sur un réseau comme des nœuds, et les connexions entre les hôtes comme des raccords entre ces nœuds. Le choix du trafic fait pour représenter l'activité sous forme de bords est fait sur la base d'ensembles de règles écrits par l'utilisateur. Les événements du réseau sont représentés à travers un mode graphique qui permet à l'observateur de déterminer la présence d'une activité suspecte. Il serait contraignant de reporter toute l'activité réseau dans un même graphique, par conséquent, le système permet pour plusieurs ensembles de règles de définir un graphe pour chaque ensemble. Dans ce cas, toutes les données recueillies sont prises en considération pour définir l'inclusion dans tous les ensembles de règles, et donc deux ensembles de règles différents pourraient rendre ainsi les mêmes données d'audit que deux graphiques différents.

JiNao

JiNao [197] est un système de détection d'intrusions réseau qui vise à protéger l'infrastructure du réseau elle-même, plutôt que l'individu hôte sur ce réseau. Il est basé sur un modèle de menace qui suppose que certaines entités de routage dans un réseau peuvent être compromises, et oblige ces entités même à arrêter complètement le routage. Le prototype suppose que les routeurs communiquent via le protocole OSPF. Les auteurs indiquent que la détection d'intrusions dans JiNao est exploitée au moyen de trois paradigmes différents : de la détection basée sur l'abus, de la détection d'anomalies, et de la détection basée sur des protocoles (de détournement).

EMERALD

L'environnement EMERALD [158], [159] est une suite d'outils distribués et modulaires permettant de traquer les activités malicieuses dans un réseau à large échelle. EMERALD [158] contient ainsi un module de détection d'anomalies basé sur les travaux initiés pour NIDES. Alors que NIDES est utilisé pour définir un profil utilisateur, le module intégré dans EMERALD a été étendu afin de pouvoir définir un profil réseau. EMERALD est ainsi constitué d'un ensemble de moteurs de génération de profil (engine profiler) permettant une séparation totale entre la représentation du profil et les algorithmes mathématiques utilisés pour évaluer les nouvelles observations.

ADAM

Le projet ADAM (Audit Data Analysis and Mining) [34] est un IDS réseau comportemental basé-anomalies. Son objectif est d'apprendre de ces attaques et de les représenter sous forme d'un jeu de règles appelé " Profils ". Il est composé de trois modules, un moteur de prétraitement, un moteur de " Mining " et un moteur de classification. Il fonctionne en temps réel et utilise un mode de Mining incrémental. L'analyse se fait en deux phases. La première phase permet de créer un modèle de profil dit " normal ". Ce modèle permet d'en générer un modèle de référence à travers des règles. Une fois cette étape exécutée, la seconde phase a pour but de capturer les flux de données entrants, de les comparer avec les règles " normales " puis lors de la détection d'un comportement suspicieux, ce dernier est automatiquement notifié soit comme attaque ou comme fausse alarme.

MADAM ID

Le projet MADAM ID (Mining Audit Data for Automated Models for Intrusion Detection) [8] a été créé dans le but de montrer comment les techniques de data mining pouvaient être utilisées dans la construction d'IDS dans une approche plus systémique et de façon automatisée. L'approche utilisée est de définir un système de classification intelligent qui aura la capacité de distinguer les activités normales des intrusions. Malheureusement, les moteurs de classification voient leurs performances limitées car ils ne sont qu'un chainon final de l'analyse technique. C'est dans ce but que MADAM ID propose des règles d'associations permettant de définir des attributs plus prédictifs, ce qui offre aux moteurs de classification des mises à jour plus rapides pour leurs données. L'approche utilisée par MADAM ID est une détection par scénario. Son but principal est de recenser les données auditées et d'en définir des modèles d'intrusions ou d'activités normales.

2.5.7 Réponses d'IDS

Les réponses d'un IDS sont cet ensemble d'actions que le système prend une fois qu'il conclut que la source d'information est malveillante. C'est aussi la capacité de reconnaître une activité ou un événement comme une attaque et de prendre des mesures pour prévenir cette attaque.

L'ensemble de ces réponses se divise en deux grandes catégories : les réponses actives et les réponses passives. Ces deux types de réponses sont décrits comme suit :

A. Les réponses actives

Ce sont les actions entreprises par l'IDS pour répondre à une attaque. Ces réponses sont prises immédiatement et automatiquement par l'IDS. Les réponses les plus com-

munes sont la collecte d'informations supplémentaires, la modification de l'environnement ou la prise d'actions contre l'attaquant.

- la collecte d'informations supplémentaires : Afin d'identifier une attaque de manière précise, il est important de rassembler des informations additionnelles sur cette dernière. La collecte de ces informations se fait dans un environnement plutôt fermé où la communication avec d'autres IDS installés sur le réseau ainsi que l'augmentation du nombre et de la sensibilité des logs sont introduites ;
- modification de l'environnement : Typiquement, les IDS n'ont pas la capacité de bloquer l'accès à une personne spécifique, cependant ils peuvent bloquer l'accès à l'adresse IP de l'attaquant, rompre une connexion, ou bloquer certains paquets, et cela en stoppant l'alerte à travers les actions suivantes :
  - injecter un paquet TCP reset dans la connexion de l'attaquant pour l'arrêter ;
  - Et/ou reconfigurer le routeur et le firewall pour bloquer les paquets de l'attaquant selon son adresse IP ou selon son numéro de port, le protocole ou le service utilisé par cet attaquant.
- Actions contre l'attaquant : le premier point qui doit être soulevé à cet égard est la légalité des actions de contre-attaque, car plusieurs attaquants falsifient leurs adresses IP pendant l'attaque, et des mesures de contre-attaque peuvent entrainer l'endommagement des sites internet ou causer du tort à des utilisateurs innocents. Quoique ces actions peuvent ne pas être des contre-attaques sévères, mais plutôt l'essai d'obtention d'informations sur l'hôte ou l'emplacement de l'attaquant.

B. Les réponses passives

Elles sont normalement prises par l'administrateur pour répondre à une attaque. Ce processus se produit après la collecte et la corrélation des données de l'événement par l'administrateur. Les différents types de réponses passives sont les alarmes et les notifications ainsi que les traps SNMTP et les plugins, ces types de réponses sont englobés en deux familles et définis comme suit :

- **Les alarmes et les notifications :** Des alarmes et des notifications sont générées par l'IDS pour informer les utilisateurs lorsque des attaques sont détectées. La forme la plus commune d'une alarme est une alerte à l'écran ou une fenêtre popup. Ces attaques et alarmes peuvent être affichées sur la console IDS ou sur d'autres systèmes tel que spécifié par l'utilisateur pendant la configuration de l'IDS
- **Les traps SNMTP et Les plugins :** Les traps et les messages SNMP génèrent des alarmes et les signalent au système de gestion de réseau. Cela permet à toute l'infrastructure du réseau de répondre à l'attaque et offre la capacité d'utiliser les canaux communs de communication.

2.5.8 Limites des IDS

La capacité d'un IDS à détecter une attaque dépend de la présence au niveau de la source de donnée d'un capteur et de la capacité de l'analyseur à identifier l'activité comme étant intrusive. Cependant, le nombre d'événements remarquables détectables dépend aussi du fait que l'homme fasse partie du système, car en un jour de travail, l'administrateur du système enregistre et rapporte une quantité définie de détections, aussi, certains événements détectés ne sont pas classables ou pas assez documentés, et c'est les administrateurs qui doivent prendre des décisions à propos de ces événements, ces décisions peuvent être souvent erronées [182].

La détection d'intrusions vise, normalement, à fournir un faible taux de faux négatifs (vraies alertes négligées) et de faux positifs (fausses alarmes). Néanmoins, utilisés en milieu opérationnel, les IDS émettent un nombre important de faux positifs, et délaissent un nombre important d'attaques, créant ainsi un nombre conséquent d'alertes fausses positives et d'autres fausses négatives.

2.6 Gestion des alarmes IDS

Traiter les alarmes est l'un des axes de recherche dans le domaine de la détection d'intrusions, cependant, peu de travaux de recherche ont été réalisés dans ce domaine pour le moment. Les principaux objectifs des recherches faites peuvent se classer en deux catégories :

1. Réduire le nombre de fausses alarmes ;
2. Grouper les alarmes similaires et les présenter aux utilisateurs sous une forme simplifiée, afin que les administrateurs puissent mieux comprendre les rapports. Ce qui aide aussi à mieux comprendre l'origine des fausses alarmes.

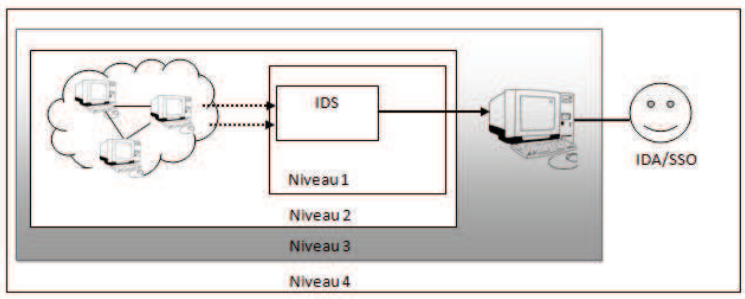

FIGURE 2.4 – Evolution du périmètre pour traiter les faux positifs [156]

Après avoir mentionné les deux principaux objectifs de la recherche dans les fausses alarmes issues d'IDS, nous pensons que le data mining pourrait représenter une solution adaptée pour résoudre ce problème.

Afin de réduire le nombre de fausses alarmes, il est necessaire de correctement classer ces dérnières ainsi que les vraies alertes.

Tadeusz Pietrazek [156] a regroupé les approches relevant de la gestion d'alertes en quatre niveaux comme illustré par la figure 2.4.

Nous allons dans ce qui suit présenter ces niveaux de manière ascendante.

Niveau 1 : Amélioration de l'IDS : Comme premières tentatives, la plupart des efforts se sont concentrés sur l'amélioration des capteurs et la création de capteurs permettant une meilleure détection d'intrusions ou produisant un faible taux de faux positifs. Les capteurs de détection d'intrusions ont évolué de simples moteurs de pattern-matching aux capteurs intelligents spécialisés pouvant comprendre les protocoles de transport sous-jacents, et même des techniques sophistiquées telle que la détection du niveau de fragmentation IP ou TCP. Suivant la croissance des vulnérabilités au niveau applicatif [9], l'intelligence des IDS a elle aussi évolué afin d'accompagner cette croissance. Les IDS ont été améliorés / développés afin de traiter divers protocoles de niveau applicatif (http, RPC). Aussi, les langages de signatures sont devenus de plus en plus puissants supportant ainsi les expressions régulières (tel que Snort) et même les interactions de protocoles complexes. Ces IDS spécialisés, contrairement aux IDS généraux, se concentrent sur des types particuliers d'intrusions permettant en même temps un faible taux de faux positifs.

Enfin, nous pouvons noter que ces IDS à usage particulier doivent impérativement être complétés par des IDS supplémentaires afin d'atteindre une couverture aussi complète que possible contre toutes les attaques, ce qui soulève la nécessité de déployer et de maintenir un réseau hétérogène d'IDS complémentaires.

Niveau 2 : Tirer parti de l'environnement : les IDS possèdent une vue limitée de l'environnement et ne peuvent donc pas distinguer, dans certains cas, avec certitude entre les attaques et les non-attaques. En utilisant des informations sur l'environnement, fournies par la vulnérabilité des scanners et des bases de données du système d'exploitation, les IDS en comparant leur environnement peuvent réduire de manière considérable le taux de faux positifs. Les principaux travaux traitant ce niveau se basent sur la corrélation d'alarmes.

Par exemple, Ptacek et Newsham [161] ont montré que, sans savoir comment les hôtes cibles gèrent certaines anomalies dans les paquets réseau, les intrus peuvent utiliser efficacement la fragmentation pour éviter d'être détectés par l'IDS réseau. Pour répondre à ces préoccupations, les approches d'active mapping [171] construisent des profils de l'environnement, qui peuvent ensuite être exploités par les IDS. Les approches basées sur les signatures de contexte [174], d'autre part, peuvent comprendre les interactions de protocole de niveau application et donc de déterminer l'impact d'une attaque. Un effet similaire peut être obtenu en mettant en corrélation des alertes de vulnérabilités [166] [72].

Niveau 3 : Post-traitement d'alarmes : le post-traitement d'alarmes utilise les alertes générées par un IDS en entrée et tente d'améliorer la qualité de ces alertes à travers leur traitement. Cela est obtenu en faisant appel au datamining et aux systèmes de corrélation d'alertes. Julisch [100] en se basant sur le data mining a montré que les techniques de data mining pouvaient se montrer très efficaces dans la découverte, l'éradication et la prévention des attaques fausses positives. En outre la corrélation d'alertes permet aussi de contrer le problème de faux positifs, mais aborde aussi le problème de redondance dans le flux d'alertes issues d'un IDS.

Niveau 4 : Implication de l'analyste : peu de systèmes traitant l'un des niveaux précédemment cités tiennent compte du fait que les alertes générées par les IDS sont transmises à l'analyste humain.

2.6.1 Post-traitement d'alarmes

Le post-traitement d'alarmes alloue l'identification de nouvelles attaques ou description d'événements suspicieux. Le moteur analyse et traite les données par rapport aux règles et politiques afin de déterminer si une donnée est malveillante ou bénigne. Le post-traitement implique la corrélation d'alarmes et la collecte d'informations provenant à partir d'autres moteurs d'analyse. Cette collecte d'informations est principalement assurée à travers des mécanismes d'extraction de connaissances. Enfin, l'unité d'intervention décide pour chaque alarme du type de réponse nécessaire : active / passive. Le post-traitement souffre néanmoins de la limitation du nombre de ressources pour l'investigation, car il ne dispose que d'un nombre limité d'alarmes pour le traitement.

A. Post traitement par corrélation d'alarmes

La corrélation d'alarmes permet de générer de nouvelles alarmes à partir de celles existantes. Elle consiste à combiner les informations fragmentées contenues dans la séquence d'alarmes et interpréter l'ensemble du flux d'alarmes. La corrélation offre principalement le filtrage des alarmes redondantes et le filtrage des alarmes de faible priorité, elle constitue une étape préalable à une contre-mesure efficace. Afin d'expliquer l'importance de la corrélation d'alarmes, nous allons prendre l'exemple d'une authentification échouée, cela génère une alerte de faible intensité. Mais s'il y a une série d'authentifications échouées avec des utilisateurs différents, on peut conclure à une attaque de force brute.

Plusieurs propositions de corrélation d'alarmes (par exemple [13], [22], [19], [14] et [23]) sont limitées à des règles prédéfinies pour les scénarios d'attaques et de contre-mesures. Une autre proposition pour trouver de nouveaux scénarios est donnée par Qin et al. [20]. Une approche différente est la corrélation probabiliste d'alerte adoptée par Valdes et al. [21], où un cadre mathématique est utilisé pour rassembler des alertes en fonction de leur similarité.

B. Post traitement par Datamining

L'extraction de connaissances, communément appelée datamining, est un domaine aujourd'hui très sollicité. Nous pouvons définir le datamining [5] comme étant un domaine qui s'occupe de résoudre les problèmes en analysant les données déjà présentes dans des bases de données [191]. Il s'intéresse aux techniques de recherche et de description des modèles structurels de données comme un outil d'aide à expliquer ces données et à faire des prévisions à partir de celui-ci. Les données prennent la forme d'un ensemble d'exemples, et la sortie est sous la forme de prédictions sur de nouveaux exemples.

Diverses méthodes d'apprentissage automatique et de datamining ont été proposées pour la détection d'intrusions, et ont eu un grand succès [115] [76] ; [130] ; [34]. Par exemple, les arbres de décision et les règles d'association floues ont été utilisées dans la détection d'intrusions [173] [126]. Les réseaux de neurones, eux, ont été utilisés pour améliorer les performances de détection d'intrusions [121]. Support Vector Machine (SVM) a été utilisé pour la détection d'anomalies non supervisée [60] et pour la détection d'intrusions supervisée [143]

C. Discussion

Table 2.4 – Comparaison des techniques de géstion d'alertes

Technique	Points forts	Faiblesses
Corrélation d'alarmes	- Donne un aperçu sur les relations entre les alarmes, leur cause ainsi qu'une présentation claire de ces derniéres	Ne permet pas de réduire le nombre d'alarmes fausses positives
Datamining	- Augmentation des performances des moteurs de recherche - Permet de créer des liens pertinents entre les données qui paraissent n'avoir aucune corrélation	-Le datamining est totalement dépendant de la base de données qu'il analyse et donc de sa taille

A partir du tableau 2.4 nous pouvons remarquer que la corrélation d'alarmes malgré qu'elle soit très intéressante pour la détection d'intrusions, ne constitue pas un choix judicieux pour la réduction d'alarmes fausses positives, dans le sens où cette technique ne permet qu'un simple filtrage de ce type d'alertes. Par contre, le datamining, de manière générale, offre des techniques permettant non seulement d'obtenir des taux de détection satisfaisants, mais aussi de distinguer et de traiter ce type d'alertes. Cependant, cette famille de techniques est fortement dépendante de la taille de la base de données, ce qui constitue un sérieux problème à résoudre lors de l'application du datamining dans

la réduction d'alarmes fausses positives.

2.6.2 Datamining : L'extraction de connaissances

Il existe diverses manières permettant de représenter les motifs qui peuvent être découverts par l'apprentissage automatique, et chacune dicte le type de techniques qui peuvent être utilisées pour en déduire cette structure de sortie de données [191]. Dans de nombreux exemples de datamining la sortie prend la forme d'arbres de décision ou de règles de classification, qui représentent les styles fondamentaux de connaissances que de nombreuses méthodes d'apprentissage automatique utilisent.

Il existe des variétés plus complexes de règles qui permettent de préciser les exceptions ou d'exprimer les relations entre les valeurs des attributs des différentes instances. La représentation à base d'instances se concentre sur les instances elles-mêmes plutôt que sur les règles qui régissent leurs valeurs d'attributs. Enfin, certaines méthodes d'apprentissage génèrent des clusters d'instances. Ces différentes méthodes de représentation de connaissances permettent de représenter et résoudre les différents types de problèmes d'apprentissage.

A. Arbres de décision :

Cette approche se base sur le concept : "diviser pour mieux régner" [191]. C'est une solution au problème d'apprentissage à partir d'un ensemble d'instances indépendantes à travers un style de représentation appelé arbre de décision. Les nœuds d'un arbre de décision impliquent le test d'un attribut particulier. Un Nœud feuille donne une classification qui s'applique à toutes les instances qui atteignent la feuille en question.

Pour classer une instance inconnue, elle est acheminée vers le bas de l'arbre en fonction des valeurs des attributs testés dans des Nœuds successifs, et lorsqu'une feuille est atteinte, l'instance est classée selon la classe attribuée à la feuille.

La technique la plus simple pour déterminer si un élément d'entrée correspond à une règle séquentielle est de le comparer à un seuil fixé par chaque élément de l'ensemble de règles. Une telle approche est utilisée par STAT [78] ou par SWATCH [10].

Cette technique est aussi utilisée par la version originale de Snort [7], sans doute l'outil de détection d'intrusions réseau à base de signatures le plus déployé.

Un autre système qui utilise les arbres de décision et les techniques de fouille de données pour extraire des caractéristiques à partir des données de vérification est présenté dans [189].

B. Règles de classification :

Les règles de classification sont une alternative populaire aux arbres de décision. L'antécédent ou la pré-condition d'une règle est une série de tests. Il est facile de lire un ensemble de règles directement sur un arbre de décision. Une règle est générée pour

chaque feuille. L'antécédent de la règle comporte une condition pour chaque Nœud sur le chemin de la racine à cette feuille, et le conséquent de la règle corréspond à la classe assignée par la feuille.

C. Règles d'association :

Les règles d'association ne sont pas vraiment différentes des règles de classification, sauf qu'elles peuvent prédire n'importe quel attribut, et pas seulement la classe, ce qui leur donne la liberté de prévoir des combinaisons d'attributs. Également, les règles associées ne sont pas destinées à être utilisées comme un ensemble, comme le sont les règles de classification. Différentes règles d'association expriment les différentes régularités qui sous-tendent l'ensemble de données, et prédisent généralement des choses différentes.

Le travail de Manganaris et al. [131] propose un environnement basé sur le data mining qui filtre les fausses alarmes IDS en utilisant des règles d'association pour modéliser le comportement normal de l'IDS. Dans leur approche, le comportement normal est représenté par un ensemble de fragments d'alarmes fréquentes et de règles de confiance élevée. Si les nouveaux fragments d'alarmes sont présents dans les motifs fréquents et correspondent aux règles de confiance élevée, les nouveaux fragments seraient inoffensifs. Manganaris et al. ont affirmé que la méthode de réduction permet non seulement de réduire les fausses alarmes, mais aussi de signaler certaines alarmes intéressantes qui étaient ignorées lors de la manipulation manuelle de l'alarme par le passé.

Wang Taihua et Guo Fan [17] ont proposé un algorithme non-itératif Apriori amélioré permettant de découvrir les alertes IDS. Ils ont utilisé l'intersection de deux lignes distinctes de la base de données de trafic réseau (DARPA 99) pour détecter les tendances récurrentes. Zhang Yanyan et Yao Yuan [18] ont présenté un algorithme d'association de règles basé sur les partitions pour la génération de règles IDS. Ming-Yang Su et al [15] ont proposé un algorithme incrémental de mining de règles d'association floues pour IDS basé réseau. Ratchadaporn Amornchewin et Worapoj Kreesuradej [16] ont proposé une probabilité incrémentale basée sur un algorithme de découverte de règles d'association.

D. Représentation basée-instances :

La forme la plus simple de l'apprentissage est la mémorisation pleine. Une fois que l'ensemble d'instances de formation a été mémorisé, en rencontrant une nouvelle instance la mémoire est parcourue en recherchant une ressemblance.

Tout d'abord, il faut noter que c'est une façon de représentation de connaissances qui est complètement différente : il suffit, pour cette technique, de stocker les instances elles-mêmes, son fonctionnement lorsqu'une instance dont la classe est inconnue doit être traitée se résume à trouver le rapport à celles déjà existantes dont les classes sont

connues. Au lieu d'essayer de créer des règles, cette méthode travaille directement à partir des exemples eux-mêmes. C'est ce qu'on appelle l'apprentissage à base d'instances. Dans l'apprentissage par instances, tout le travail réel est effectué lorsque vient le temps de classer une nouvelle instance plutôt que lorsque l'ensemble d'apprentissage est traité.

Il peut ne pas être nécessaire ou souhaitable de stocker toutes les instances de formation, car cela peut rendre le calcul de ressemblances insupportablement lent.

Un exemple simple d'un algorithme d'apprentissage basé sur les instances serait l'algorithme des k plus proches voisins (KNN).

En 2005, Alharby et Imai [24] ont proposé une méthode de classification d'alarmes en utilisant des séquences continues et discontinues. Ils ont recueilli de fausses alarmes IDS à partir des réseaux et les ont partitionnées en différentes séquences. Puis tous les motifs séquentiels possibles ont été extraits pour chaque séquence. Deux sortes de motifs séquentiels ont été extraites dans ce travail : les modèles continus et les modèles discontinus. L'ensemble de tous les modèles continus et discontinus représente les caractéristiques des fausses alarmes. Après avoir récupéré les schémas de fausses alarmes, les auteurs ont effectué les mêmes procédures pour les alarmes nouvellement entrantes et compté le nombre de similitudes entre les nouvelles alarmes et de fausses alarmes. Si la plupart des motifs séquentiels de nouvelles alarmes sont identiques à ceux des fausses alarmes, les nouvelles alarmes sont considérées comme fausses. Dans leurs expériences en utilisant la DARPA 1999 comme ensemble de données, un taux de réduction d'alarmes de 93% a été signalé. Cependant, l'applicabilité de cette méthode dans un environnement réel est discutable, car l'extraction de tous les motifs possibles continus et discontinus constitue une tâche de calcul très intensif.

Les principaux travaux traitant la détection d'intrusions et la réduction d'alertes fausses positives basés sur le KNN seront étudiés de manière détaillée dans le chapitre suivant.

E. Clustering :

Lorsque c'est les clusters plutôt qu'un classificateur qui sont entrainés, la sortie prend la forme d'un diagramme qui montre comment les instances arrivent dans les clusters. Le cas le plus simple consiste notamment à rattacher à chaque instance un numéro de cluster, ce qui peut être décrit en posant les instances sur deux dimensions et en partitionnant l'espace pour montrer chaque cluster.

Récemment, les algorithmes de clustering ont été proposés pour les IDS afin de surmonter ces problèmes. Plusieurs techniques ont été créées telles que STING [188], Apriori [27], FP-growth [84], WaveCluster [172], CLIQUE [26] et pMAFIA [146]. Toutefois, le nombre de nouveaux types d'intrusion est rapidement augmenté et le volume de l'information est trop grand.

Les méthodes de clustering sont généralement bonnes pour filtrer les valeurs aberrantes et de découvrir des groupes de formes arbitraires. Quelques exemples de méthodes basées sur la densité sont DBSCAN [64] et OPTIQUES [30].

Cependant, la méthode la plus connue reste celle proposée par Julisch, qui a utilisé des techniques de clustering pour analyser les alarmes IDS et déterminer leurs causes profondes [101][99]. Il a défini les causes profondes des alarmes comme les raisons pour lesquelles ces dernières se produisent. Il a observé que certaines causes profondes sont responsables de plus de 90% des alarmes, et que les alarmes IDS correspondant à la même cause sont très similaires. Par conséquent, après l'application du clustering les alarmes similaires vont être regroupées afin de former un cluster, et la caractéristique commune des alarmes dans un cluster représenterait leur cause.

F. Discussion

TABLE 2.5 – Comparaison des téchniques de Datamining

Technique	Points forts	Faiblesses
Arbres de décision	Pas de problèmes de combinaisions de variables -Sélection de variables intégrée -Génération de règles logiques simples de classification -Représentation graphique des règles	Les critères de sélection ne tiennent pas compte des densités de points dans l'espace des données
Règles d'association	Aide au réarrangement des capteurs IDS de manière statique	Ne convient pas pour la réduction temps réel de fausses alarmes
Clustering	Bonne réduction d'alarmes fausses positives (82%) Haute disponibilité	Nécessite une configuration manuelle Coût important Exige un temps de calcul important
Apprentissage basé sur les instances	Très bon taux de réduction d'alarmes fausses positives (93%)	Exige une puissance de calcul très élevée

Le tableau 2.5 montre que les arbres de décision, malgré leurs avantages intéressants, ne constituent pas la meilleure option à explorer pour la réduction d'alarmes fausses positives.

Par contre, le clustering et l'apprentissage basé sur les instances offrent les meilleures capacités de réduction. Cependant, ces techniques souffrent d'un défaut majeur qui est le temps de calcul nécessaire au traitement d'une grande quantité d'alarmes, ce qui est considéré comme un obstacle à leur utilisation en temps-réel. Ceci-dit, le clustering

nécessite dans la majorité des cas une configuration manuelle pour permettre l'adaptation de l'approche au cas traité, pour cela nous avons jugé l'apprentissage basé sur les instances plus adéquat pour le traitement des alertes fausses positives.

Enfin, les techniques basées sur les règles d'association ne considèrent guère une réduction en temps réel.

2.7 Conclusion

Pour réduire le nombre de fausses alarmes, la méthode de détection doit être suffisamment efficace pour filtrer un certain nombre de fausses alarmes. En outre, la nature de l'algorithme de filtrage d'alarmes doit permettre la classification en temps réel, car la détection d'intrusions est un processus continu qui exige une réponse rapide. Par ailleurs, pour réduire la charge des administrateurs réseau, le processus de détection devrait impliquer un minimum de procédures manuelles.

D'une autre part, le type de traitement le plus adéquat, pour les alarmes, selon nos études est la classification. Toutefois ces algorithmes souffrent, en général, d'une complexité de calcul importante.

L'un des classifieurs les plus efficaces est le KNN, ce classifieur est trés prometteur dans le domaine de la classification des alertes.

Dans le chapitre suivant nous allons présenter ce classifieur en détail.

Chapitre 3

Sélection d'instances pour la méthode KNN

L'apprentissage supervisé est la tâche la plus fondamentale de l'apprentissage automatique. L'apprentissage supervisé comprend deux types d'exemples, ceux dits de formation et ceux dits de test.

Un exemple de formation est un couple (x_i, y_i) où x est une instance et y est une étiquette ou une classe.

Un exemple de test est une instance x avec une étiquette inconnue.

L'objectif de ce type apprentissage est de permettre la prédiction des étiquettes relatives aux exemples de test.

Parmi les différentes méthodes d'apprentissage supervisé, la règle des plus proches voisins (KNN) réalise des performances élevées, sans à priori sur les distributions de formation à partir desquelles les exemples sont tirés.

La méthode des plus proches voisins est peut-être la méthode la plus simple de tous les algorithmes de classification, sa phase de formation est triviale : il suffit de stocker tous les exemples de formation, avec leurs étiquettes.

3.1 Présentation

La règle de classification des K-plus proches voisins (KNN) est une méthode de classification puissante permettant la classification d'une instance inconnue en utilisant un ensemble d'instances de formation classées.

L'algorithme des k-plus proches voisins (k-Nearest Neighbours) permet la classification d'instances à partir des instances de formation les plus proches dans l'espace caractéristique.

Il s'agit d'un type d'apprentissage basé sur les instances, appelé aussi memory-based ou lazy-learning, car il n'y a pas d'apprentissage réel, les exemples d'apprentissage sont juste stockés en mémoire et réutilisés lors de la classification.

L'idée principale de l'algorithme est de prédire pour chaque nouvelle observation les

k observations lui étant les plus similaires dans l'ensemble de données d'apprentissage.

Lorsqu'on parle de voisin cela implique la notion de similarité ou de distance. La distance la plus souvent utilisée est la distance euclidienne définie par la fonction 3.1 où x_i et u_i représentent des instances distinctes de p attributs chacune.

$$D((x_1, ..., x_p), (u_1, ..., u_p)) = \sqrt{(x_1 - u_1)^2 + ...(x_p - u_p)^2} \tag{3.1}$$

Le cas le plus simple de cet algorithme est lorsque k est égale à 1. 1-NN se base sur la classe du voisin le plus proche afin de classer l'instance inconnue.

Pour étendre 1-NN à k-NN il suffit de chercher les k-1 autres plus proches instances, puis d'utiliser une règle de décision à la majorité pour classer l'observation.

KNN se caractérise par la possibilité de faire une classification sans émettre d'hypothèses sur la fonction reliant la variable dépendante (classe) aux variables indépendantes (instances), mais aussi, par l'influence de la valeur de k qui peut être choisie dans une échelle allant de quelques unités à quelques milliers, où les grandes valeurs de k produisent un lissage qui réduit le sur-apprentissage dû au bruit, ce qui est un avantage.

Un exemple de classification KNN est illustré par la figure 3.1 où le point inconnu (étoile) appartient soit à la première classe (carré) ou à la deuxième classe (triangle). Si K = 3, le point inconnu est classé en deuxième classe parce qu'il y a deux triangles et un seul carré parmi les trois plus proches exemples à l'intérieur du cercle. Si K = 5, il est classé dans la première classe.

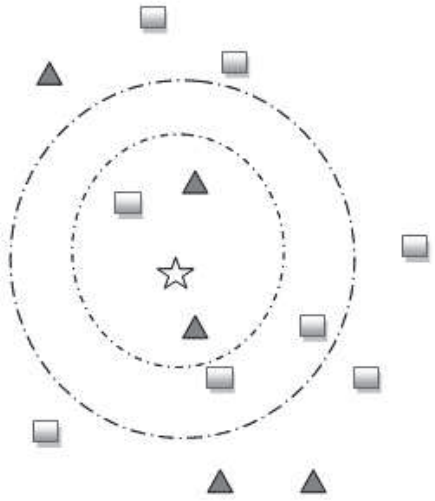

FIGURE 3.1 – Exemple de classification KNN (K=3 et K=5)

3.2 L'algorithme de base

L'algorithme KNN est un algorithme d'apprentissage supervisé où le résultat est le classement d'une nouvelle instance selon la majorité des k-catégories de ses instances

voisines les plus proches. Chaque instance I_i de l'ensemble d'entrainement est sous forme d'un couple, composé de plusieures entrées et d'une sortie, où $I_i =< a_{1i}, a_{2i}, ..., a_{ji}, c_i >$. Les entrées représentent les attributs $(a_{1i}, a_{2i}, ..., a_{ji})$ de l'instance I_i, et la sortie sa classe respective (c_i).

L'idée de l'algorithme est de prédire pour une instance non classée les k instances classées lui étant les plus similaires, dans les données d'apprentissage, et d'utiliser ces classements pour affilier l'instance à une classe.

La plus simple présentation de cet algorithme est exposée par l'algorithme 1 :

1 Algorithme KNN

Arguments :
X : L'instance non classée
S : L'ensemble de toutes les instances (l'espace d'etude)
Y_i : Les classes des instances de S
Y_{ij} : Les instances de S appartenant à la classe Y_i
d : La métrique utilisée (distance)

Entrées : X, S, d, k
Sorties : Y= classe de X

Debut :

1. Pour chaque Y_{ij} dans S faire
   - Calculer d (X, Y_{ij})
   - Fin pour ;
2. Classer les distances par ordre croissant
3. Compter le nombre d'occurrences de chaque classe Y_i parmi les k plus proches selon l'ordre
4. Y= la classe la plus fréquente
5. Retourner Y

Fin

D'autres algorithmes plus compliqués pour KNN ont été proposés. Chaque algorithme diffère des autres selon son application, mais le principe reste le même que celui de l'algorithme de base présenté ci-dessus.

3.2.1 Mesures de similarité

Le but d'une mesure de similarité est de comparer deux listes de valeurs (vecteurs par exemple), et de calculer une valeur unique qui évalue leur similarité. La plupart des mesures ont été élaborées dans le cadre de la comparaison des paires de variables à travers des cas. En d'autres termes, l'objectif est de déterminer dans quelle mesure deux variables co-varient, c'est-à-dire, quand est ce qu'elles peuvent avoir les mêmes valeurs pour les mêmes cas.

Le principe général est que la mesure de similarité doit être invariante sous les transformations de données admissibles.

A. Métrique et distance de Minkowski

La métrique de Minkowski est une mesure généralisée qui inclut les autres mesures comme des cas particuliers de la forme généralisée. Bien que théoriquement un nombre infini de mesures peuvent exister en faisant varier l'ordre de l'équation, seulement trois mesures ont bénéficié de beaucoup d'attention : la distance de Minkowski, la distance Euclidienne et la distance Manhattan.

La distance de Minkowski est souvent utilisée lorsque les variables sont mesurées sur des échelles de rapport avec une valeur zéro absolue. Les variables avec un éventail plus large peuvent dominer le résultat. Même les quelques valeurs aberrantes avec valeurs élevées biaisent le résultat et ne tiennent pas compte de la ressemblance offerte par un couple de variables proches.

Les métriques de Minkowski sont définies par :

$$D_r(X_i, X_k) = (\sum_{(j=1)}^{d} |x_{ij} - x_{kj}|^r)^{(1/r)} \tag{3.2}$$

Où :
- X : Ensemble de données multi variées
- r : Exposant de la métrique
- d : Dimension de données (le nombre d'attributs d'un ensemble de données)
- x_i : Scalaire représentant une mesure d'un vecteur de données

B. Distance Euclidienne

La plus simple et la plus populaire des mesures de similarité entre des données multi variées est la distance Euclidienne, qui est un cas particulier de la famille des métriques de Minkowski (quand l'exposant de la métrique r=2).

La distance Euclidienne est la racine carrée de la somme des carrés des différences entre les éléments correspondants des deux vecteurs. Cette distance est le plus souvent utilisée pour comparer les profils de variables. Elle peut être définie comme suit :

$$D_e(X_i, X_k) = \sqrt{(\sum_{(j=1)}^{d} (x_{ij} - x_{kj})^2)} \tag{3.3}$$

C. Distance Manhattan

La distance de Manhattan permet de calculer la distance qui serait parcourue pour se rendre d'un point de données à l'autre si un chemin de type grille est suivie. La

distance de Manhattan entre deux éléments est la somme des différences de leurs composantes correspondantes.

La distance de Manhattan est aussi un cas spécial des métriques de Minkowski tel que r=1. Elle est plus appropriée pour mesurer la similarité entre des données multi variées.

La formule de cette distance entre deux points est décrite par :

$$D_m(X_i, X_k) = |x_{ij} - x_{kj}| \tag{3.4}$$

3.3 Etat de l'art des approches de détection d'intrusions basées sur l'algorithme KNN

Yihua Liao et Vemuri [119] ont utilisé une nouvelle approche basée sur le classificateur KNN pour classer le comportement des programmes en normal et intrusif. L'apprentissage des profiles d'un programme nécessite du temps de calcul et demande des phases d'entrainement et des tests. En utilisant le classificateur KNN, les fréquences des appels systèmes ont été utilisées pour décrire le comportement du programme. Les techniques de catégorisation de texte ont été adoptées par les auteurs afin de convertir chaque exécution d'un programme en un vecteur et calculer la similarité entre deux activités du programme. Ceci implique qu'il n'y avait plus besoin de générer des profiles individuels pour les programmes, et donc les calculs nécessaires ont pu être largement réduits. De courtes séquences d'appels système ont été utilisées afin de caractériser le comportement normal d'un programme. Cependant, diverses bases de données de courtes séquences d'appels système devaient être construites pour différents programmes. Les auteurs ont traité les appels systèmes différemment : au lieu de voir l'ordre local des appels système, la méthode proposée utilise les fréquences de ces derniers pour catégoriser le comportement d'un programme. Ce stratagème a permis le traitement de beaucoup d'appels système d'affilé comme si c'était une seule unité, en écartant le besoin de construire et de maintenir des bases de données séparées pour chaque programme. En utilisant la métaphore de traitement de texte, chaque système est traité tel un " mot " dans un long document, et l'ensemble d'appels système généré par un processus est traité comme le " document ". Cette analogie permet d'extraire le meilleur des méthodes de traitement de texte pour traiter le problème de détection d'intrusions. La méthode introduite dans ce travail est la méthode des k−plus proches voisins (KNN). Afin de classer un vecteur du document d'une classe inconnue, le classificateur KNN classe les voisins du document et utilise le label de la classe des K plus similaires voisins pour prédire la classe du nouveau document. Les classes de ces documents sont pondérées en utilisant la similarité de chaque voisin par rapport au document de test, où la similarité a été mesurée en utilisant la distance Euclidienne

ou la valeur cosine entre deux vecteurs documents. Comparée à d'autres méthodes de catégorisation de texte, KNN est efficace du point de vue calculs. Le principal calcul est le stockage des documents d'entrainement afin de trouver les K plus proches voisins pour le document de test.

La méthode proposée par Law et Kwok [114] consiste à déterminer si les séquences d'alarmes entrantes sont différentes de la situation normale, si oui, une alerte peut être signe d'attaque et une investigation plus approfondie est nécessaire, sinon le risque d'être attaqué est faible. Etant donné un grand nombre d'alarmes générées par un IDS dans un environnement sans attaques avec N types distincts d'alarmes il y a un ensemble distinct de points avec N attributs dans un espace représentant ce nombre d'alarmes de types différents dans une fenêtre temporelle. Pour détecter les modèles d'alertes anormales nouvellement arrivées, de nouveaux points de données sont crées pour ces nouvelles alarmes. La distance de ces points par rapport aux points normaux indique leur déviation de la situation normale. Le classificateur KNN a été utilisé pour classer les données et savoir si un point est normal ou non. KNN a d'abord été utilisé dans la zone de détection d'intrusions pour la détection d'anomalies d'apprentissage et la découverte du comportement du programme d'intrusion à partir des données d'audit. Les auteurs ont proposé de l'étendre à l'utilisation pour la détection de fausses alarmes en se basant sur la distance Euclidienne, qui représente la similitude entre deux points. Plus cette distance est petite plus les points sont similaires. Le processus de détection d'alarmes a utilisé les alarmes normales (alarmes relevées par l'IDS en cas d'aucune attaque) pour construire le modèle de fausses alarmes. Les alarmes entrantes sont filtrées en permanence par une procédure de filtrage utilisant ces modèles comme patrons. La modélisation proposée par les auteurs ainsi que les procèdures de filtrage ont été indépendantes du processus de détection d'intrusions. Par conséquent, ce modèle peut être appliqué à la plupart des IDS commerciaux sans rien changer dans leur configuration.

Yang Li et Li Guo [118] ont proposé une nouvelle méthode de détection d'intrusions basée sur l'algorithme d'apprentissage automatique TCM-KNN, ainsi qu'une méthode de sélection de données d'apprentissage basée sur l'apprentissage actif.

Cette approche a été la première qui inclut le TCM-KNN dans la détection d'intrusions. Pour cette utilisation les auteurs ont optimisé le TCM-KNN sur deux aspects. La première amélioration est l'introduction d'une méthode d'apprentissage actif afin de sélectionner un petit nombre de données mais de bonne qualité pour l'apprentissage, ce qui permet d'allèger la quantité de données étiquetées et de réduire la taille de l'ensemble d'apprentissage, et donc de réduire aussi le coût de calcul du TCM-KNN. Le second aspect a été la proposition d'une méthode de sélection d'attributs les plus importants et nécessaires pour le TCM-KNN. Les auteurs ont attribué à chaque point une mesure appelée mesure d'étrangeté individuelle. Cette mesure définie l'étrangeté

du point par rapport au reste des points. La mesure d'étrangeté utilisée a été le rapport de la somme des k distances les plus proches de la même classe à la somme des k distances les plus proches de toutes les autres classes. Cette mesure croît lorsque la distance entre les points de la même classe augmente ou lorsque la distance entre les autres classes devient plus petite. L'algorithme comprend deux phases importantes : la phase de formation et la phase de détection. Dans la première phase trois actions principales doivent être considérées : la collecte des données pour la modélisation, la sélection d'attributs et enfin la modélisation par l'algorithme TCM-KNN et donc la construction du classificateur de détection d'intrusions. Pour la phase de détection toutes les données recueillies en temps réel du réseau doivent être prétraitées en étant vectorisées suivant les attributs sélectionnés.

Sur la base de leur précédente proposition (TCM-KNN), les mêmes auteurs ont présenté un mécanisme de sélection d'instance pour TCM-KNN basée sur EFCM (Extended CMeans Fuzzy) pour la détection d'anomalies. Ils ont introduit un algorithme de clustering [194], visant à limiter la taille des données de formation, réduisant ainsi le coût de calcul du TCM-KNN et améliorant le rendement de sa détection. Tout d'abord, les auteurs ont proposé de classer l'ensemble de données d'entraînement normal en introduisant trois classes : les données notables, des données obscures et les données redondantes. Les données notables ont été définies comme celles appartenant à une série de clusters. Ces données représentent ces clusters et y sont les plus centrées. Les données obscures désignent celles qui appartiennent à un cluster avec de très petits grades d'appartenance calculés par EFCM. Les données redondantes comprennent, quand à elles, les données restantes qui n'appartiennent à aucune des deux classes précédentes, et qui sont essentiellement les données qui se trouvent le long des limites des clusters. Ils ont ensuite proposé de former l'algorithme TCM-KNN avec les données notables et obscures, afin que ce dernier puisse offrir le meilleur usage possible en permettant de distinguer les anomalies des cas normaux avec un taux de détection élevé et un faible taux de faux positifs, ainsi qu'un coût de calcul réduit.

Liwei Kuang and Mohammad Zulkernine [107] ont proposé une méthode de détection d'anomalies : CSI-KNN en combinant l'étrangeté et la mesure d'isolation. L'algorithme de détection d'intrusions analyse différentes caractéristiques des données du réseau en employant deux mesures : l'étrangeté et l'isolation.

La métrique d'étrangeté mesure si le comportement inconnu est plus semblable aux comportements normaux ou aux comportements anormaux. Elle effectue une détection d'anomalies en analysant la distribution d'étrangeté des données normales mais qui traite également les attributs d'anomalies en employant des données d'attaque. La métrique d'isolation identifie la similarité par rapport aux classes normales et peut détecter les attaques anormales. C'est une méthode pure de détection d'anomalies qui n'utilise que les données normales

Basée sur ces mesures, une unité de corrélation soulève les alertes d'intrusions et les estimations de confiance associées. Multiples classificateurs CSI-KNN travaillent en parallèle afin de traiter différents types de services réseau. Le composant de corrélation agrège les résultats de l'étrangeté et les modèles d'isolement et fournit une décision finale. En outre, l'algorithme fournit une confiance graduée pour chaque alerte d'intrusion.

Dans leur article CHENG et al. [193] ont proposé une nouvelle méthode de détection d'anomalies réseau basée sur l'algorithme KNN-MARS, qui a été appliquée avec succès à la reconnaissance des formes, la détection des fraudes et la détection des valeurs aberrantes.

Les auteurs ont proposé un système hybride incluant l'algorithme KNN et l'algorithme MARS. Ce système traite naturellement avec le réglage multi-classe. Il a une complexité de calcul raisonnable à la fois dans l'apprentissage et au moment de l'exécution, et donne d'excellents résultats dans la pratique.

L'idée de base est de trouver des voisins à proximité d'un prototype de requête et de former un MARS local qui préserve la fonction de distance de collecte des voisins. Une grande variété de fonctions de distance a été utilisée dans les expérimentations montrant les performances sur un certain nombre de données de référence fixe pour la classification IDS et la reconnaissance d'objets.

La version simplifiée de KNN-MARS est lente, principalement parce que cet algorithme doit calculer les distances de la requête par rapport à tous les prototypes de formation, ce qui est très coûteux.

Le but de Dave et Richhariya [48] était de créer une méthode de détection d'intrusions en utilisant la classification KNN et la théorie de Dempster. Grâce à ces méthodes les auteurs ont rassemblé un nouveau motif d'intrusion, classé les catégories de modèles et appliqué une logique de preuve à l'aide de la Théorie-DS. Ils ont proposé un système de détection d'intrusions basé sur le classificateur KNN et la DS-Theory en incluant la logique floue. Les données d'entrée du système proposé ont été recueillies à partir du KDD Cup 1999, ce benchmark a été séparé en deux sous-ensembles : ensemble de données de formation et les données de test. Après cela, ils ont simplement extrait les modèles 1-longueur fréquentes à partir des données d'attaque ainsi que des données normales. Ces modèles minés (mined) fréquents ont été utilisés pour trouver les attributs importants de l'ensemble de données d'entrée. Les attributs identifiés efficaces ont été utilisés pour générer un ensemble de règles définies et indéfinies en utilisant la méthode de déviation.

Ensuite, les auteurs ont généré une règle floue en conformité avec la règle définie en la fuzzifyant , de manière à obtenir un ensemble de règles floues si-alors avec des parties conséquentes qui indiqent si c'est une donnée normale ou anormale. Ces règles ont été inclues dans la base de règles floues pour parfaire l'apprentissage du système. Dans la

phase de tests les données de tests ont été identifiées aux règles floues pour détecter si les données de test sont des données normales ou non. Les auteurs ont appliqué la classification KNN et la théorie de preuve de Dempster pour classer les données. A travers cela, ils ont pu réunir un nouveau modèle de découverte d'intrusions, classer les catégories de modèles et appliquer la logique de preuve d'événements à l'aide de la Théorie-DS. Le modèle de l'intrusion était comparé au modèle existant, s'il y a intrusion, un nouveau schéma de modèle était dans ce cas généré et la liste de modèles d'intrusions mise à jour et le taux réel de détection d'intrusions amélioré. Les auteurs ont utilisé les concepts de la théorie de Dempster afin de découvrir la validité des données et réduire le taux d'intrusions. Ils ont aussi utilisé les modèles pour élaborer des schémas et convertir les données de la forme chaine de caractères à la forme numérique, plus précisément, ils ont utilisé pour cela le ratio-mapping.

3.4 Ensembles de données à grande dimension

Une des limitations de la règle des k-plus proches voisins est la taille de son ensemble d'apprentissage [178][199]. Si le nombre d'instances de formation est trop petit, la précision du classifieur KNN n'est pas acceptable. Par contre, si l'ensemble d'apprentissage est trop grand, le temps d'execution peut devenir excessif pour beaucoups de classificateurs KNN.

Cette profusion de données à traiter sollicite une réduction de leur nombre pour arriver à un traitement de haute performance.

Ce problème peut être résolu de trois manières [186] : En réduisant les dimensions de l'espace de représentation, à l'aide de petits ensembles de données, ou en utilisant un algorithme amélioré qui peut accélérer le temps de calcul.

La réduction de données est un processus utilisé dans le but de transformer les grands ensembles de données brutes en une forme plus condensée, sans perdre d'importantes informations sémantiques. La réduction des données se réfère à la fonction de sélection et d'échantillonnage [3]. D'après [155] la réduction de données est considérée comme une tâche principale dans le data mining, donc, toute technique de data mining est vue par [155] comme une méthode de réduction de données.

Les techniques de réduction les plus connues sont : la sélection d'attributs [124], la génération d'attributs [82], La discrétisation d'attributs [123], la génération d'instances [112] et la sélection d'instances [122].

3.5 Sélection d'instances

La communauté scientifique s'est penchée sur le problème de réduction d'ensembles et a proposé la sélection d'instances qui tend à modifier un ensemble initial d'instances afin de réduire sa taille pour améliorer les performances de classification.

la sélection d'instances (IS) est le processus permettant de trouver les modèles représentatifs des données, qui peuvent aider à réduire le nombre de ces données. Ce problème est classé comme un problème NP-difficile par de nombreux chercheurs [44] [74], car il n'existe pas d'algorithme polynomial permettant d'obtenir la solution exacte. Les algorithmes existants peuvent néanmoins offrir des solutions acceptables.

Comme beaucoup d'autres problèmes combinatoires, la sélection d'instances necessite une recherche exhaustive afin d'obtenir des solutions optimales. Cela a conduit certains chercheurs à considérer ce problème comme un problème d'optimisation combinatoire et à utiliser des techniques générales qui sont connues pour leurs bons résultats dans des situations similaires.

Selon leur but final, les méthodes de sélection d'instances peuvent être divisées en deux types de techniques [39] [47] [54], pouvant être référencées comme les téchniques " editing " et les téchniques de " condensing ".

Les techniques d'editing [144] visent à éliminer les valeurs aberrantes et les instances provocant un chevauchement entre les classes. Ces méthodes n'entrainent généralement pas de réduction substantielle de la taille, mais produisent des clusters homogènes d'instances conduisant à des résultats optimaux pour la classification 1-NN.

Les algorithmes de condensing [54] [45] [105] [98] [40] essayent, quand à eux, de trouver une réduction significative de l'ensemble d'instances de manière à ce que le classement 1-NN donne des résultats aussi proches que possible de ceux obtenus par le même classifieur en utilisant toutes les instances originales.

Suivant le but souhaité, nous pouvons définir deux stratégies de sélection d'instances qui sont : la sélection de prototypes et la sélection d'ensembles d'entrainements.

3.5.1 Sélection d'instances pour la sélection de prototypes

Le classificateur 1-NN prédit la classe d'un exemple inédit en calculant sa ressemblance avec un ensemble d'instances stockées appelées prototypes. Stocker un sous-ensemble bien choisi d'instances de formation a été montré comme capable d'améliorer la précision du classificateur dans de nombreux domaines. En même temps, l'utilisation d'un nombre restreint de prototypes diminue considérablement, à la fois, les coûts de stockage et le temps de classification.

Un algorithme de sélection de prototypes (PS) est un algorithme de sélection d'instances (IS) qui tente d'obtenir un sous-ensemble de l'ensemble d'apprentissage qui permet au classificateur 1-NN d'obtenir le taux maximum de classification. Chaque algorithme PS est appliqué à un ensemble initial de données dans le but d'obtenir un sous-ensemble d'éléments de données représentatives. La précision du sous-ensemble sélectionné est évaluée à l'aide d'un classifieur 1-NN.

La figure 3.2 montre le fonctionnement d'un algorithme de sélection de prototypes.

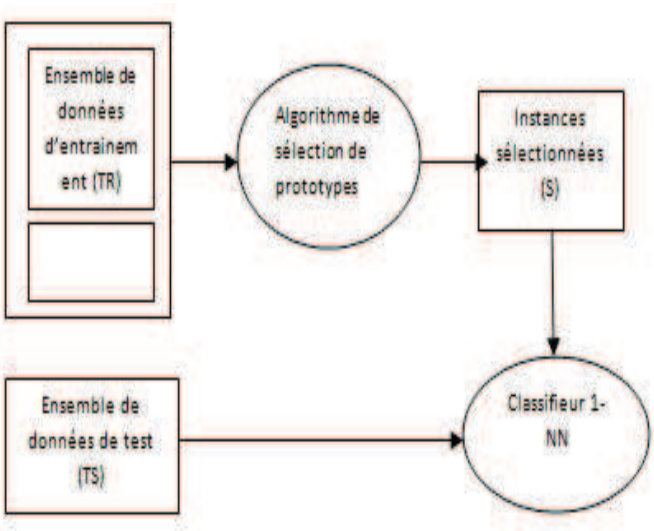

FIGURE 3.2 – Stratégie IS-PS

3.5.2 Sélection d'instances pour la sélection d'ensemble d'apprentissage

Dans certaines situations il peut y avoir trop de données, et dans la plupart des cas, ces données ne sont pas très utiles dans la phase de formation d'un algorithme d'apprentissage [164]. Les mécanismes de sélection d'instances ont été proposés pour choisir les points les plus appropriés dans l'ensemble de données comme des données d'apprentissage qui seront utilisées par un algorithme d'apprentissage.

3.6 Etat de l'art sur la sélection d'instances pour l'algorithme KNN

Selon la manière par laquelle les prototypes sont obtenus et représentés, une séparation peut être faite entre les techniques de sélection, où les prototypes résultants sont pris de l'ensemble initial, et les techniques de remplacement où les prototypes résultants sont construits et peuvent être différents de tous les prototypes de l'ensemble initial.

3.6.1 L'algorithme CNN

En 1968 Hart a été le premier à proposer dans [155] une méthode pour la réduction de la taille de données stockées pour la décision du plus proche voisin, appelée "The Condensed Nearest Neighbor Rule" ou CNN, le mot " condensed " fait référence à une procédure de choix d'un sous ensemble (TCNN) de l'ensemble d'apprentissage initial (TNN), ce sous ensemble doit être aussi performant que TNN dans la classification des modèles inconnus.

Effectivement, la règle permet de minimiser le nombre de modèles stockés en ne stockant qu'un sous ensemble de données de formation pour la classification, cela en utili-

sant une technique de faible absorption. L'idée de base est de chercher les modèles d'entrainement très similaires, et ceux qui n'ajoutent pas d'informations supplémentaires et les éliminer.

La notion de cohérence du TNN est importante dans la formation de TCNN, ce qui fait que la cohérence du sous ensemble minimal est critique dans le sens où il doit contenir toutes les informations essentielles de TNN. Bien qu'il soit exigé que TCNN soit cohérent rien ne garantit qu'il soit minimal.

3.6.2 L'algorithme RNN

Introduite par Gates dans [75], "The Reduced Nearest Neighbor Rule", ou RNN, est une extension de la règle CNN , et comme CNN, RNN réduit TNN.

L'algorithme de RNN commence à TRNN=TCNN et supprime chaque instance à partir de TRNN si cette suppression ne cause pas une mal classification d'une autre instance dans TNN par les instances restantes dans TRNN. Du point de vue du calcul, cette règle est plus couteuse que la règle proposée par Hart, mais elle permet toujours de produire un sous ensemble de CNN, et donc est moins couteuse en terme de calcul et de stockage lors de l'étape de classification.

3.6.3 L'algorithme FCNN

Angiulli a proposé "The Fast Condensed Nearest Neighbor" [29], ou FCNN, un algorithme pour la création de sous ensembles cohérents servants d'ensembles d'apprentissage, basés sur la règle de décision du plus proche voisin. Cet algorithme permet de sélectionner des points très proches de la frontière de décision. Il est indépendant de l'ordre, et a une faible complexité quadratique. Cet algorithme initialise le sous-ensemble compatible avec un élément de semence provenant de chaque étiquette de classe de l'ensemble d'apprentissage. En particulier, les semences utilisées sont les centroïdes des classes dans l'ensemble d'apprentissage. FCNN est incrémental ; il permet à chaque itération, d'augmenter l'ensemble S jusqu'à atteindre la condition d'arrêt.

3.6.4 Les algorithmes DROP 1-5 et DEL

Wilson et martinez ont proposé dans leur article [190] une suite de six algorithmes de réduction d'ensembles basés sur le kNN où chaque algorithme corrige et améliore son prédécésseur.

Ces deux auteurs ont aussi introduit deux concepts interessants relatifs au voisinage. Le premier concept est celui des voisins associés, un voisin d'une instance P lui est associé s'il est de même classe que P. Le second concept, inclut la notion d'ennemi, qui stipule qu'un voisin de P est son ennemi s'il est d'une classe différente.

Dans ce qui suit nous allons passer en revue cette série d'algorithmes, proposée par Wilson et martinez.

DROP1

La première technique de réduction présentée a été le DROP1 qui constitue le cadre de base sur lequel ont été bâties les cinq autres techniques. Le DROP1 représente une amélioration de la règle RNN, vue précédemment. Il vise à vérifier la précision sur l'ensemble résultant (S) au lieu de l'ensemble initial (T). Cet algorithme se base sur le principe qui stipule qu'une instance "P" n'est retirée que si, au moins, plusieurs de ses associés dans S peuvent être classés correctement sans elle.

Cette première proposition pose un problème quand aux instances bruyantes, qui ont typiquement des associés d'une classe différente et qui ne couvrent par conséquent qu'une faible partie de l'espace d'entrées.

DROP2

DROP2 essaie de remédier au probmème posé par DROP1 en éliminant les associés des instances bruyantes appartenant à d'autres classes leur permettant de couvrir de plus en plus d'espace d'entrée, et cela en considérant l'effet de la suppression d'une instance sur toutes les instances de la formation initiale T plutôt que sur S. Dans ce but, la règle du DROP2 propose de n'éliminer P que si au moins un bon nombre de ses associés dans T peuvent être classés correctement sans P.

En utilisant cette modification, chaque instance P dans l'ensemble de formation initial T continue à maintenir une liste de ses k+1 plus proches voisins dans S, même après qu'elle soit supprimée de S.

DROP2 trie S dans une tentative de supprimer les points centraux avant les points frontaliers, or, les instances bruyantes peuvent être, elles aussi, frontalières, ce qui peut causer un changement dans l'ordre de retrait.

Aussi, même si une instance bruyante est centrale, la tentative de la supprimer pourrait éliminer des points frontaliers qui devaient être maintenus.

DROP3

Afin de corriger DROP2, DROP3 utilise un filtrage de bruit avant de trier les instances de S. Ceci est fait en utilisant une règle qui élimine toute instance mal classée par ses k plus proches voisins.

En suivant ce principe, la règle du DROP3 peut dans certains cas retirer un très grand nombre d'instances, et voir même toutes les instances de l'ensemble initial.

DROP4

DROP4 améliore la règle du DROP3 qui peut retirer un très grand nombre d'instances. Dans DROP4, une instance n'est retirée que si :

1. Elle est mal classée par ses k plus proches voisins, et

2. Sa suppression ne nuit pas au classement d'autres instances

DROP5

DROP5 reprend l'algorithme DROP2, pour proposer une correction qui stipule que les instances soient considérées en commençant par celles qui sont les plus proches de leur ennemi le plus proche, et en procédant vers l'extérieur.

DEL

Le dernier algorithme proposé dans [190] est DEL qui est similaire au DROP3, sauf qu'il utilise l'heuristique de codage de longueur pour décider si une instance peut être retirée ou pas. Dans DEL une instance n'est supprimée que si :

1. Elle est mal classée par ses k plus proches voisins, et
2. La suppression de l'instance n'augmente pas le coût de codage de longueur

3.6.5 L'algorithme GMCA

Mollineda, Ferri et Vidal ont proposé un schéma de classification généralisé basé-prototype et fondé sur le clustering hiérarchique [163].

L'idée de base a été d'obtenir une règle de classification 1-NN en fusionnant les deux clusters les plus proches de même classe.

Les auteurs ont dévoilé leur algorithme "Generalized-Modified Chang Algorithm" (GMCA), qui est une extension de l'algorithme de Chang et de celui du Chang modifié (MCA) pour une utilisation sur des clusters, et cela en plusieurs étapes.

Ils ont d'abord présenté un algorithme général de fusion de prototypes qui considère les échantillons d'un ensemble de formation T comme des prototypes initiaux. L'idée principale a été de considérer des clusters constitués d'échantillons initiaux et de leurs représentants comme une sorte de prototypes étendus. De cette façon la fusion devient une union de deux clusters, tandis que la distance entre prototypes devient une distance interclusters. La cohérence de l'ensemble de représentants à l'égard de l'ensemble d'échantillons initial sera atteinte par le fait que chaque représentant est responsable de la classification correcte de son cluster.

Comme seconde étape, les auteurs ont proposé une procédure de vérification de cohérence basée sur trois conditions afin de valider les fusions possibles.

3.6.6 L'algorithme IKNN

Wu, Ianakiev et Govindraju ont proposé [198] d'augmenter la vitesse de classification du kNN classique tout en maintenant son niveau de précision , et cela en suggérant deux techniques nommées " Template Condensing " et " Preprocessing " construisant ensemble l'algorithme "Inproved K-Nearest Neighbor" (IKNN) qui se base sur l'élimination itérative des modèles exposants de hautes capacités d'attraction.

Les auteurs ont commencé par proposer un calcul pour la capacité d'attraction d'un modèle y, qui est définie comme étant le nombre de modèles de la classe c(y) qui sont les plus proches de y que les autres modèles appartenant à d'autres classes.

L'idée de L'algorithme a été d'alléger les ensembles de données initiaux en éliminant une large partie de prototypes qui ne sont pas susceptibles de correspondre au motif inconnu.

Pour la première technique "la condensation" (ou Condensing) les auteurs ont proposé que toutes les classes soient équiprobables, et que l'ensemble d'apprentissage soit initialement créé par l'extraction des vecteurs d'éléments à partir d'un très grand ensemble d'images d'intérêts où chaque classe a une représentation égale aux autres.

Pour arriver à cette condensation, les auteurs ont proposé un seuil d'attraction qui les aide à définir les éléments à éliminer, et cela en supprimant ceux qui le dépassent.

Dans la seconde technique, qui est le prétraitement (ou Preprocessing), un motif inconnu est comparé à un prototype en deux étapes successives. Dans la première étape une évaluation rapide du potentiel de correspondance est établie. Les prototypes qui échouent dans cette première correspondance ne sont pas pris en compte dans la seconde. Puis, pour qu'une correspondance complète ait lieu, dans la deuxième étape, la différence entre la norme du prototype et celle du modèle de test doit être inferieure à un seuil conçu pour chaque prototype individuellement.

3.6.7 La méthode TRKNN

Fayed et Atia ont proposé dans [73] le "Template Reduction for KNN" (TRKNN) qui est une manière d'alléger le problème posé par les exigences de calcul lors de la classification de modèles en utilisant le kNN sur de grands ensembles de données. L'objectif de leur approche a été d'éliminer les motifs qui sont une charge pour le calcul et qui ne contribuent pas à améliorer la classification.

Cette approche consiste à rejeter les prototypes qui sont loin des limites et ont peu d'influence sur la classification du kNN. Pour réaliser cela, les auteurs ont d'abord introduit le concept de chaine des plus proches voisins qui est une séquence des voisins les plus proches de classes alternées entre la classe du modèle et les classes de ses voisins.

Puis, pour chaque modèle de l'ensemble d'apprentissage, un élément est éliminé s'il satisfait la formule $d_{ij} > \alpha . d_{ij+1}$, où α est un seuil supérieur à 1, et cela jusqu'à la fin de la chaine. L'algorithme impose que seuls les voisins de la même classe que l'élément de départ puissent être éliminés.

3.6.8 L'algorithme CBP

Nikolaidis et al. [104] ont introduit " The Class Boundary Preserving Algorithm " (CBP), qui est une méthode en plusieurs étapes pour élaguer l'ensemble d'apprentissage. La méthode proposée combine la sélection et l'abstraction afin d'obtenir un nouvel

ensemble d'instances condensé, elle vise à préserver les instances qui sont à proximité des frontières de classes, car, d'après les auteurs, elles peuvent fournir la plupart des informations nécessaires pour décrire correctement la distribution sous-jacente. D'autre part, les instances lointaines des limites sont considérées comme redondantes par les auteurs, car elles n'affectent pas la surface de décision. Aussi, les positions relatives des instances par rapport à leurs plus proches ennemis sont prises en compte afin de faire la distinction entre les vecteurs frontaliers et ceux non frontaliers. L'innovation de cette approche réside dans la procédure utilisée pour diviser l'ensemble d'apprentissage en deux sous ensembles, un premier ensemble comprenant les instances à proximité de la surface de décision, et un second contenant les échantillons internes. Mais en raison d'une différence notable dans l'importance des informations détenues par ces deux ensembles, deux processus de réduction différents ont été appliqués sur l'un et l'autre.

3.6.9 Comparaison

Aprés avoir fait l'étude de certains travaux s'étant intéressés à la réduction d'ensembles basés sur le KNN, nous avons jugé utile de dresser un tableau comparatif regroupant l'idée, les avantages, les inconvénients ainsi que les données ciblées par chaque algorithme.

CNN	Idée	Eliminer les modèles d'entrainement très similaires et ceux qui n'ajoutent pas d'informations supplémentaires à la classification
	Avantages	- Améliore le temps de recherche et les besoins en mémoire - Réduit la taille des données d'entrainement
	Inconvénients	- CNN est dépendant de l'ordre, il est alors peu probable qu'il élimine des points frontaliers - Si l'ensemble initial est minimal alors l'ensemble résultant est égale à l'ensemble initial, ce qui provoque une incohérence de l'ensemble résultant à l'arrêt du programme - Rien ne peut garantir que l'ensemble résultant soit minimal
	Données ciblées	Ensembles de données où le besoin en mémoire est la principale préoccupation
RNN	Idée	Au départ l'ensemble résultant est égale à l'ensemble initial, puis chaque instance qui ne cause pas une mal classification d'une autre instance dans l'ensemble initial est éliminée de l'ensemble final
	Avantages	- Réduit la taille des données d'entrainement et élimine des modèles - Améliore le temps de recherche et les besoins en mémoire

	Inconvénients	- Coût de calcul élevé - Nécessite beaucoup de temps - Sa cohérence dépend de la cohérence de l'ensemble résultant du CNN
	Données ciblées	Grands ensembles de données
DROP1	Idée	Au départ l'ensemble résultant est égale à l'ensemble initial, puis une instance n'est éliminée que si au moins plusieurs de ses associés dans l'ensemble résultant peuvent être classés sans elle
	Avantages	- Réduit la taille des données d'entrainement et élimine des instances - Constitue une base sur laquelle sont bâtis le reste des algorithmes DROP et DEL - Sa précision ne se dégrade pas sous l'effet du bruit
	Inconvénients	- Vérifie la cohérence sur l'ensemble résultant au lieu de l'ensemble initial - Les instances bruyantes ne couvrent qu'une faible partie de l'espace d'entrée - Faible précision - Ne peut pas utiliser les informations des instances précédemment éliminées
	Données ciblées	Grands ensembles de données
DROP2	Idée	Améliore DROP1 et n'élimine une instance que si au moins un bon nombre de ses associés dans l'ensemble initial peuvent être classés sans elle
	Avantages	- Vérifie la cohérence sur l'ensemble initial plutôt que sur l'ensemble final - Réduit la taille des données d'entrainement et élimine des instances - Atteint une précision plus élevée que celle de KNN en cas d'instances bruyantes - Bonne réduction de stockage qui arrive à 1/6 de l'ensemble d'origine - Besoins en stockage inferieurs à ceux des DROP1, 4, 5
	Inconvénients	- Tente de supprimer les points centraux avant les frontaliers, ce qui néglige les instances frontalières bruyantes - La tentative d'éliminer une instance bruyante centrale peut supprimer des points frontaliers qui devaient être maintenus - Pour de très grands ensembles de données DROP2 n'élimine pas assez de points
	Données ciblées	Grands ensembles de données
DROP3	Idée	Améliore le DROP2 en éliminant les instances mal classées par leurs voisins

	Avantages	- Se base sur le classement de l'instance elle-même pour la supprimer - Réduit la taille des données d'entrainement et élimine des instances - Atteint une précision plus élevée que celle du KNN classique en cas d'instances bruyantes - Très bonne réduction de stockage qui arrive à 12% de l'ensemble d'origine - Besoins en stockage inferieurs à ceux des DROP1, 4, 5
	Inconvénients	- Peut dans certains cas retirer un très grand nombre d'instances
	Données ciblées	Grands ensembles de données
DEL	Idée	Améliore DROP3 et n'élimine une instance que si : 1. Elle est mal classée par ses k plus proches voisins, 2. La suppression de l'instance n'augmente pas le coût de codage de longueur
	Avantages	- Atteint une précision plus élevée que celle du KNN classique en cas d'instances bruyantes - Réduit la taille des données d'entrainement et élimine des instances
	Inconvénients	- Précision moins élevée que celle des DROP2-5 - Besoins en stockage plus élevés que ceux des DROP2-5
	Données ciblées	Grands ensembles de données
GMCA	Idée	Généralisation de l'algorithme MCA à une utilisation sur des clusters
	Avantages	- Donne de plus petits ensembles de prototypes que son précédent MCA - Le taux d'erreurs de test du 1-NN et du meilleur K-NN sont réduits - Temps de calcul moins élevé de 3% que celui de MCA - Propose une description naturelle de l'ensemble de données - Peut être utilisé avec toute métrique satisfaisant l'égalité triangulaire et la symétrie - Toute métrique de distance peut être utilisée en faisant quelques arrangements sur les conditions 2 et 3
	Inconvénients	- Temps de calcul relativement élevé - L'aspect naturel de l'ensemble n'est pas utilisé par le classificateur associé
	Données ciblées	Grands ensembles de données
IKNN	Idée	Eliminer itérativement les modèles exposants de hautes capacités d'attraction

	Avantages	- Allège l'ensemble de données en gardant les prototypes qui sont utiles - Le prétraitement permet une économie significative en temps de calcul - Le classificateur montre une légère amélioration dans la précision par rapport au kNN classique - Réduit la taille des modèles en maintenant le même niveau de précision
	Inconvénients	- Les classes doivent être équiprobables dans l'ensemble de formation - Le travail a été bâti sur l'intuition et pas sur un cadre mathématique citant que la norme est une caractéristique intrinsèque - Pas de preuve théorique que le prétraitement garanti de ne filtrer que les prototypes pertinents, ni de maintenir la précision - Un modèle de test est une version déformée d'un prototype lorsque la différence de norme est inferieur à un seuil associé à ce prototype
	Données ciblées	Grands ensembles de données
TRKNN	Idée	Eliminer les motifs qui sont une charge pour le calcul et qui ne contribuent pas à améliorer la classification
	Avantages	- Réduit la taille des modèles sans sacrifier la précision - TRKNN est jusqu'à 3 fois plus rapide qu'IKNN et jusqu'à 4 fois que DROP2
	Inconvénients	- Le taux de réduction reste relativement faible (35%) - Niveau moyen de précision
	Données ciblées	Grands ensembles de données
CBP	Idée	Vise à préserver les instances qui sont à proximité des frontières des classes
	Avantages	- Combine la sélection et l'abstraction - Applique à chaque cas une procédure de réduction appropriée - La phase de filtrage se base sur le classement de l'instance elle-même - Utilise les caractéristiques géométriques de la distribution - Permet une vision globale de la répartition des échantillons - Réduit la taille de l'ensemble d'apprentissage ainsi que le nombre de représentants - Très bon taux de réduction
	Inconvénients	- L'algorithme de filtrage peut dans certains cas retirer un très grand nombre d'instances - I(x) n'a été expérimenté que pour K=3 - Temps de calcul relativement élevé
	Données ciblées	Grands ensembles de données

3.7 Discussion

Les approches traditionnelles, telles que la règle du NN Condensé (CNN), la règle du NN Réduit (RNN) et les autres règles heuristiques basées sur le voisinage, sélectionnent les prototypes, à partir des échantillons de formation de l'ensemble initial, en ajoutant ou en taillant, dans l'objectif de préserver les performances de classification en utilisant des méthodes heuristiques. Cependant, aucune de ces méthodes ne peut donner de preuve sur la minimalité de l'ensemble résultant.

Nous pouvons donc conclure par l'hypothèse que l'utilisation des métaheuristiques, ou de l'apprentissage actif, dans l'amélioration de ces solutions pourrait garantir une meilleure convérgence des résultats.

3.8 Les métaheuristiques

Comme beaucoup d'autres problèmes combinatoires, la sélection d'instances exigerait une recherche exhaustive afin d'obtenir des solutions optimales dans le cas général.

Ceci a conduit certains chercheurs à suggérer l'utilisation de techniques générales qui sont connues pour de bons résultats dans des situations similaires telles que les algorithmes évolutionnaires.

On peut partager les métaheuristiques en deux grandes classes : les métaheuristiques à solution unique (S-meta) et celles à population de solutions. Les méthodes d'optimisation à population de solutions , telles que les algorithmes génétiques, améliorent, au fur et à mesure des itérations, une population de solutions (P-meta). L'intérêt de ces méthodes est d'utiliser la population comme facteur de diversité. Les méthodes d'optimisation à solution unique telles que les techniques de recherche locale, évoluent quand à elles avec une seule solution comme solution de départ et tentent à travers des itérations d'améliorer cette solution.

Les algorithmes évolutionnaires (AE) sont des stratégies qui orientent la recherche vers une solution optimale. Ces techniques ont pour but d'explorer l'espace de recherche de manière efficace afin de déterminer des solutions (presque) optimales. Ils contiennent des mécanismes permettant d'éviter le blocage dans les domaines de la recherche spatiale. Les AEs contiennent aussi des techniques permettant d'intensifier la recherche dans d'autres domaines. Les algorithmes évolutionnaires ont été utilisés avec succès dans différents problèmes de datamining [141], et ont été récemment utilisés pour la sélection d'instances, et ont offert de bons résultats [41].

Dans cette partie, nous allons évoquer les principales techniques de métaheuristiques et invoquer l'utilisation des algorithmes métaheuristiques dans la résolution du problème de sélection d'instances, en présentant les deux techniques les plus utilisées : les algorithmes génétiques et la recherche locale. Enfin, nous aborderons la présentation d'une nouvelle génération d'algorithmes dits mémétiques ainsi que leurs principales appari-

tions dans la sélection d'instances.

3.8.1 Les algorithmes génétiques

Les algorithmes génétiques (GA) sont une technique d'optimisation basée sur une population de solutions, et guidée par les principes de l'évolution et la génétique naturelle, marqués par une haute présence de parallélisme implicite. Ces algorithmes effectuent une recherche dans des paysages complexes, grands et multimodaux, et fournissent des solutions quasi-optimales pour la fonction objective.

Ces algorithmes ont été inspirés de la théorie de l'évolution et des processus biologiques permettant l'adaptation des organismes à leur environnement, et présentés pour la première fois par Holland en 1962. Ces algorithmes sont réputés pour bien fonctionner dans des espaces de recherche importants.

Un algorithme génétique permet de rechercher les extrema d'une fonction par rapport à un ensemble de données.

Pour pouvoir faire appel à ce type d'algorithmes, cinq éléments sont essentiels.

1. *Un principe de codage pour les éléments de la population :* Dans cette étape chaque point de l'espace d'état se voit associer une structure de données. Les structures les plus utilisées sont le codage binaire et celui réel. Notons que la qualité du codage des données conditionne le succès de l'algorithme.
2. *Un mécanisme de génération de la population initiale :* le but de cette étape est de produire une population d'individus non homogènes. Le choix de cette population est crucial. Et dans le cas où on ne connait pas la nature de l'élément inconnu, cette population doit couvrir tout le domaine de recherche.
3. *Une fonction à optimiser :* qui doit retourner la valeur de fitness.
4. *Des operateurs permettant de diversifier la population au cours des générations et d'explorer l'espace d'état :* L'operateur de croisement recompose les gènes d'un individu existant dans la population, et l'operateur de mutation permet de garantir l'exploration de l'espace d'état.
5. *Des paramètres de dimensionnement :* tels que la taille de la population, le nombre total de générations, les probabilités d'application des operateurs de croisement et la mutation.

Principe de fonctionnement

Dans l'algorithme génétique, une population initiale de solutions potentielles est créée au hasard. Puis, cette population est soumise à une imitation de l'évolution des espèces : croisement, mutation et sélection. Les individus vont donc générer d'autres individus. Ensuite, parmi les individus, certains, choisis aléatoirement, subissent des mutations, qui peuvent être vues comme un changement aléatoire d'une caractéristique de l'individu. Finalement une sélection naturelle est appliquée, où les individus les

moins adaptés au milieu d'évolution disparaissent. Ces étapes sont réitérées un certain nombre de fois en favorisant la survie des solutions les plus correctes.

Ce procédé permet que les générations successives soient de plus en plus adaptées à la résolution du problème au fur et à mesure des itérations.

En résumé l'algorithme génétique repose sur une boucle qui enchaîne des étapes de sélection et des étapes de croisements et de mutations, et peut être décrit par le pseudo-code 3.8.1 ci-après.

2 Pseudo-code d'un algorithme génétique standard

```
Initialiser la population initiale P
Evaluer P
Tant que ( !(condition d'arrêt)) faire
    P'=Sélectionner les individus autorisés pour le croisement (parents)
    P'=Appliquer l'opérateur de croisement sur P'
    P'=Appliquer l'opérateur de mutation sur P'
    P'= Remplacer les anciens individus de P par leurs descendants de P'
    Evaluer de nouveau P
Fin tant que
```

3.8.2 Etat de l'art sur la sélection d'instances basée sur les concepts génétiques

Dans cette section, nous présentons quelques principales approches génétiques utilisant différentes variantes de l'algorithme KNN.

Nous décrivons ces méthodes en tentant d'éclairer les nouveautés introduites ainsi que les apports de ces approches par rapport à la résolution du problème.

La méthode GGA

Un algorithme évolutionnaire de sélection d'instances intéressant a été décrit dans [79], cet algorithme est le "Generational Genetic Algorithm" (GGA). L'idée de base de GGA est de maintenir une population de chromosomes, qui représentent des solutions plausibles à ce problème particulier. Ces solutions évoluent avec les itérations successives (générations) à travers un processus de concurrence et de variation contrôlée. Chaque chromosome de la population posséde une fitness associée. Cette valeur de fitness permet de déterminer quels chromosomes sont utilisés pour en former de nouveaux dans le processus de concours.

La méthode de recherche adaptative CHC

L'algorithme CHC [62] est un algorithme de recherche adaptative. Il utilise une population mère afin de générer une population intermédiaire d'individus, qui sont croisés au hasard et utilisés pour générer une progéniture. Ensuite, une compétition de survie est tenue, où les meilleurs chromosomes parents et les populations progénitures

sont sélectionnés pour former la prochaine génération. CHC met également en œuvre une forme de recombinaison hétérogène en utilisant un opérateur de recombinaison spécifique.

L'algorithme SSGA

JR Cano, F. Herrera, et M. Lozano dans leur étude expérimentale [41] ont présenté le "Steady-State Genetic Algorithm". Dans SSGAs, usuellement seulement un ou deux descendants sont produits à chaque génération. Les parents sont sélectionnés pour produire une descendance, et une stratégie de remplacement / suppression définit les membres de la population qui seront remplacés par la nouvelle progéniture.

Un algorithme génétique avec MSE

R. Gil-Pita et X. Yao [77] ont proposé trois améliorations de l'algorithme des k plus proches voisins en faisant appel aux algorithmes génétiques. Compte tenu des propriétés statistiques du classificateur kNN, les auteurs ont proposé une nouvelle fonction objective basée sur la moyenne quadratique d'erreur (MSE), qu'ils ont appelé "Editing kNN with a MSE-Based Objective Function". Cette amélioration peut être, selon les auteurs, plus performante que la fonction objective basée sur l'estimateur de comptage. Leur deuxième proposition "Editing kNN Using a GA with Clustered Crossover" a été une analyse de la relation entre les gènes de l'AG, qui a été utilisé pour proposer un crossover en cluster ou un croisement groupé. Enfin, il ont proposé, dans leur troisiéme approche "Editing kNN Using a GA with a Fast Smart Mutation Scheme", un nouveau système intelligent de mutation rapide qui permet d'évaluer rapidement les variations de la fonction objective.

Une méthode hybride Génétique-Bayésienne

Les auteurs de [25] ont mis en place une technique hybride comprenant l'algorithme KNN, les algorithmes génétiques et la méthode bayésienne. Cette approche se compose de huit étapes. Elle commence par l'application de l'algorithme bayésien. Ensuite, la technique génère de nouvelles données en utilisant l'algorithme génétique et applique la méthode des K plus proches voisins. Enfin, les étapes réstantes comprennent plusieurs itérations de ces algorithmes supervisées par le mécanisme génétique.

Le KNN génétique

Suguna et Thanushkodi [178] ont combiné un algorithme génétique à l'algorithme des k-plus proches voisins pour proposer le "Genetic KNN" (GKNN). Dans cette méthode, à chaque itération de l'algorithme, k instances génétiques sont sélectionnées et l'exactitude des taux de classification est calculée comme fitness. La meilleure précision

est enregistrée chaque fois. Ainsi, il n'est pas nécessaire de calculer les distances entre toutes les instances, ou de tenir compte des poids de catégories.

3.8.3 La recherche locale

La recherche locale est une famille de métaheuristiques à solution unique fondée sur la notion de voisinage. Elle comprend les techniques de descente, mais aussi des métaheuristiques plus évoluées telles que le recuit simulé et la recherche taboue.

Cette recherche est liée à la notion de voisinage où un optimum local est une solution qui n'en connaît pas de meilleure parmi ses voisines. Aussi, les résultats sont fortement dépendants de la structure de voisinage, mais aussi de la fonction d'évaluation.

Une fonction de voisinage V : S $\rightarrow$ P(S) associe à toute configuration S l'ensemble V(S) des voisins de S (le voisinage de S).

La métaheuristique de recherche locale la plus simple est l'algorithme de descente ou amélioration itérative (dite aussi basic local search). L'algorithme de descente consiste, à chaque itération, à choisir un voisin qui améliore strictement la fonction de coût. Le principal défaut de cet algorithme de recherche est qu'il s'arrête nécessairement quand un optimum local est atteint.

La recherche Taboue est un algorithme métaheuristique utilisé pour résoudre des problèmes d'optimisation combinatoire. Elle utilise des procédures itératives locales ou par voisinage pour passer d'une solution x à une solution x' (dans le voisinage de x), jusqu'à ce que les conditions d'arrêt soient satisfaites.

Afin d'éviter les optimas locaux, la recherche taboue permet de poursuivre la recherche de solutions même lorsqu'un optimum local est rencontré et ce, en permettant des déplacements qui n'améliorent pas la solution.

3.8.4 Etat de l'art des algorithmes de recherche locale pour SI

Après avoir introduit les notions de base de la recherche locale et décrit les principaux types de cette catégorie, nous allons dans ce qui suit, présenter les approches qui nous ont parues les plus connues basées sur la recherche locale et traitant la sélection d'instances.

Un algorithme de recherche taboue et recherche paralléle aléatoire

Une première tentative d'appliquer la recherche taboue au problème de sélection d'instances a été incluse dans [185]. Dans cette proposition les auteurs ont considéré l'espace de solutions X constitué de tous les sous ensembles possibles de l'ensemble de référence initial, et une fonction objective à minimiser. A chaque étape, les auteurs sélectionnent le mouvement non-tabou avec le plus petit poids parmi ceux disponibles, et utilisent le meilleur critère d'aspiration amélioré pour permettre à un mouvement d'être considéré comme admissible en dépit de son statut tabou.

La recherche taboue sauvegarde les meilleures solutions courantes à tout moment et procède itérativement jusqu'à la satisfaction du critère de terminaison choisi.

Un algorithme de réduction basé sur la recherche taboue

Ceveron et Ferri [42] ont proposé une méthode pour obtenir un ensemble cohérent de S-prototypes, ceci en présentant une nouvelle approche, basée sur la recherche taboue, pour la sélection de prototypes pour la règle du plus proche voisin qui vise à obtenir une solution optimale ou proche de l'optimal. La particularisation du TS proposée réside dans le fait d'utiliser les équations objectives tirées des algorithmes génétiques. Dans cette approche tous les sous ensembles de prototypes possibles constituent l'espace de solutions. Les mouvements possibles à partir d'un sous ensemble particulier consistent en l'ajout ou la suppression de chacun des n prototypes initiaux. L'attribut utilisé pour déclarer les mouvements tabous est le prototype qui est ajouté ou supprimé. Le meilleur critère d'aspiration amélioré est utilisé.

Un algorithme de sélection d'instances de d'attributs basé sur la coévolution coopérative

Derrac, Garcia et Herrera [96] ont présenté le IFS-CoCo, basé sur l'évolution coopérative. Cette approche emploie deux techniques de réduction des données bien connues , qui sont la sélection d'instances et la sélection d'attributs. IFS-CoCo est bâti sur l'algorithme CHC, présenté ci-dessus (3.8.2). L'utilisation de la sélection d'instances a permis aux auteurs de choisir le sous-ensemble d'instances le plus approprié à partir des données initiales, en essayant simultanément d'augmenter la précision de la classification et de diminuer la quantité de données. La sélection d'attributs a permis de sélectionner le sous-ensemble de caractéristiques le plus approprié pour décrire les données.

3.8.5 Discussion

Les algorithmes génétiques ont prouvé que c'est de bonnes solutions pour résoudre le problème de sélection d'instances. Cependant, un inconvénient majeur de ces algorithmes est que les opérateurs standards de croisement et de mutation ne permettent pas d'intensifier suffisamment la recherche [91]. C'est pourquoi les algorithmes génétiques sont souvent hybridés avec des méthodes de recherche locale.

Ces deux méthodes sont complémentaires car l'une permet de détecter de bonnes régions dans l'espace de recherche alors que l'autre se concentre de manière intensive à explorer ces zones de l'espace de recherche. Ainsi, on peut explorer rapidement les zones intéressantes de l'espace de recherche pour les exploiter en détails.

3.8.6 Hybridation de métaheuristiques

L'hybridation permet de tirer profit des avantages cumulés des différentes métaheuristiques impliquées dans cette dernière. Les méthodes y afférant peuvent être divisées en deux classes principales [61] :

– Hybridation de bas niveau
– Hybridation de haut niveau

Une hybridation est dite de haut niveau lorsqu'une fonction d'une métaheuristique est remplacée par une autre métaheuristique. Par contre, une hybridation de bas niveau est obtenue lorsque deux métaheuristiques sont hybridées sans que leur fonctionnement interne ne soit en relation.

Chacune des deux classes précédentes se subdivise en deux sous-classes :

– A relais
– Co-évolutionnaire

Lorsque les metaheuristiques sont exécutées de façon séquentielle, l'une utilisant le résultat de la précédente comme entrée, l'hybridation est dite à relais. L'hybridation Co-évolutionnaire se fait, quant à elle, lorsque des agents coopèrent en parallèle pour explorer l'espace de solutions.

3.8.7 Les Algorithmes Mémétiques

Les approches génétiques et celles basées sur la recherche locale ont prouvé à travers les travaux réalisés qu'elles sont très utiles pour améliorer la précision des ensembles résultants des techniques heuristiques, donc, si ces deux techniques étaient mêlées ou hybridées, le résultat serait certainement plus pertinent.

De l'alliance de ces deux types d'approches résulte une autre approche qui est l'algorithme memetique [141] [145].

La notion de meme a été introduite pour la première fois par Dawkins [162] en présentant une alternative au gène. Ce meme permet non seulement de transmettre le matériel génétique à la génération suivante tout comme un gène mais aussi de léguer des concepts et des idées. Le meme peut être vu comme une unité d'information qui une fois passée à un individu, ce dernier peut l'adapter à son environnement.

Les algorithmes mémétiques sont introduits pour la première fois par Moscato [142], et définis par [68] pour englober les algorithmes introduisant la recherche locale après une mutation. Ces algorithmes hybrident les algorithmes génétiques avec des méthodes de recherche locale. Pour cette raison, ils ont aussi reçu la dénomination d'algorithmes génétiques parallèles ou algorithmes de recherche locale hybrides.

Principe

Le principe générale d'un algorithme mémétique est le même que pour un algorithme génétique à part l'ajout d'un opérateur de recherche locale qui remplace ou succède l'opérateur de mutation. La partie génétique de ces algorithmes peut être vue comme

une forte diversification alors que la partie recherche locale correspondrait à une forte intensification accompagnée d'une faible diversification.

Il existe de multiples façons de concevoir un algorithme génétique, les méthodes de recherche locale sont aussi nombreuses. L'hybridation de ces deux approches permet d'envisager un nombre considérable de combinaisons, que ce soit dans le choix des algorithmes à utiliser ou l'emplacement de la recherche locale au niveau de l'algorithme génétique.

Dans un cadre d'apprentissage automatique, les algorithmes memetiques consistent souvent à hybrider les AE avec un algorithme d'apprentissage plus classique. De telles techniques correspondent à une combinaison d'une recherche globale menant à l'obtention de caractères innés, et d'un apprentissage qui correspond à une recherche locale menant à l'émergence de caractères acquis.

Le pseudo code d'une telle hybridation peut être défini par les étapes de l'algorithme 3.8.7.

3 Pseudo-code d'un algorithme mémétique standard

Initialisation : générer une population initiale P de solutions S
Evaluer P
Tant que la condition d'arrêt n'a pas encore était atteinte faire
– P'=Sélectionner les individus autorisés pour le croisement (parents)
– E=Appliquer l'opérateur de croisement sur P'
– Pour chaque enfant E
  1. E= Appliquer l'opérateur de mutation sur E
  2. E= Améliorer E en appliquant la méthode de recherche locale
  3. Si (E satisfait la condition de remplacement)
     – Remplacer une solutionS de P par E

Fin tant que

3.8.8 Etat de l'art sur les algorithmes mémétiques pour SI

Comme indiqué ci-dessus, l'approche memtique intègre les concepts génétiques et ceux se référant à la recherche locale, c'est justement ce qui nous a permis d'émettre l'hypothèse que ces algorithmes pourraient être plus performants que ceux précédemment cités. Dans le reste de cette section, nous allons introduire une collection d'algorithmes memetiques traitant le problème de sélection d'instances et faisant appel à l'algorithme KNN.

L'algorithme ReliefF-MA

Dans [43] les auteurs ont proposé la combinaison de la méthode de filtrage "ReliefF" [103] et une méthode memetique appelée Wrapper, pour la classification. L'objectif de leur méthode a été de filtrer les éléments non pertinents et de sélectionner les sous-

ensembles les plus importants. Ils ont utilisé l'algorithme ReliefF pour calculer et mettre à jour les scores de chaque élément pour chaque ensemble de données, puis, ils ont appliqué l'algorithme memetique pour la sélection d'éléments.

ReliefF-MA constitue un processus à deux étapes. Dans la première phase, les éléments pertinents sont sélectionnés par ReliefF. Dans la seconde étape, la population MA est initialisée aléatoirement, de manière à ce que chaque chromosome code un sous ensemble candidat d'éléments. Par la suite, une recherche locale est effectuée sur la totalité ou une partie des chromosomes. Après cela, le croisement et la mutation sont appliqués aléatoirement, et une recherche locale est répétée.

L'algorithme SSMA

Salvador Garcia, José Ramon Cano et Francisco Herrera ont présenté [170] un modèle d'algorithme memetique pour la sélection d'instances. Il s'agit d'un MA statique (SSMA) qui incorpore une recherche locale ad hoc destinée à l'optimisation des propriétés du problème IS avec l'objectif de s'attaquer au problème de scaling up.

Les memes utilisés sont conçus ad hoc pour le problème de sélection d'instances, profitant de sa nature divisible et la simplicité de l'hybridation au sein de l'EA lui-même, et permettant une bonne convergence, même avec une augmentation de la taille du problème.

L'algorithme ISSMA

Dans une seconde étude [97] les auteurs de [170] ont amélioré leur SSMA en testant la combinaison de la stratification avec l'algorithme memetique statique précédemment proposé (SSMA), sur divers problèmes de taille allant de 50000 à 1 million d'instances. La stratégie de stratification divise les données de formation en strates disjoints avec une distribution de classes homogène. Puis, la téchnique SSMA a été appliquée à chaque partie de formation pour obtenir un sous ensemble sélectionné pour chaque partition. L'ensemble de prototypes sélectionnés, appelé sous-ensemble de prototypes stratifiés sélectionnés , est obtenu en joignant toutes les partitions obtenues.

3.9 L'apprentissage actif

L'apprentissage actif (Active Learning en anglais) est une technique d'apprentissage automatique supervisé dans lequel l'apprenant est en contrôle des données utilisées pour l'apprentissage. Il peut être défini par contraste à l'apprentissage classique. Dans l'apprentissage classique les étiquettes d'apprentissage sont obtenues sans référence à l'algorithme d'apprentissage. Cependant, un algorithme d'apprentissage actif sélectionne de manière interactive les points à étiqueter. Cette technique est bâtie sur le principe qui stipule que l'interaction peut réduire considérablement le nombre d'étiquettes

nécessaires, ce qui permet à la résolution des problèmes par le biais de l'apprentissage automatique d'être plus pratique.

Principe de base

Le processus d'apprentissage actif prend en entrée un ensemble d'exemples marqués, ainsi qu'un ensemble plus vaste d'exemples non marqués, et produit un classifieur ainsi qu'un ensemble de données relativement petit nouvellement marquées. L'objectif global est de créer un classifieur aussi performant que possible en ne lui fournissant que les données d'apprentissage nécessaires.

L'apprentissage actif permet au modèle de construire son ensemble d'apprentissage au coure de son entraînement, en interaction avec un superviseur. L'apprentissage débute avec peu de données étiquetées. Ensuite, le modèle sélectionne les exemples non étiquetés qu'il juge les plus " instructifs " et interroge le superviseur à propos de leurs étiquettes [36].

Le concept d'apprentissage actif est basé sur cinq piliers, qui sont : l'ensemble d'entrainement, l'ensemble des données non étiquetées, une fonction requête, un classificateur, et un superviseur. L'algorithme 3.9 donne les grandes lignes de l'apprentissage actif. L'apprenant actif reçoit un pool d'exemples non étiquetés (U) et un ensemble d'entrainement L, qui lui est initialisé à un petit nombre d'exemples étiquetés.

A chaque itération le classificateur (C) est construit à partir de toutes les données d'apprentissage étiquetées en utilisant un algorithme de classification. Le classificateur peut alors être utilisé par la fonction de requête (Q) afin d'aider à la sélection d'exemples informatifs. Un exemple est sélectionné en utilisant la fonction requête et supprimé du pool non étiqueté. L'étiquette (l) de l'exemple sélectionné est obtenue à partir du superviseur (S), qui est une entité externe. Une fois l'étiquette connue, l'exemple étiqueté est ajouté aux données d'entrainement.

L'algorithme est itéré jusqu'à atteindre un critère d'arrêt.

4 Algorithme général de l'apprentissage actif

```
Entrées : L, U
Tant que (! (critère d'arrêt)) faire
    q=Q (U, C)
    U=U-q
    l=S(q)
    L=L∪(q,l)
Fin tant que
Retourner L
```

L'apprentissage actif a été appliqué avec succès dans beaucoup de domaines tels que l'extraction d'informations et la catégorisation de texte.

3.9.1 Etat de l'art des algorithmes basés sur l'apprentissage actif pour SI

L'apprentissage actif, qui est très prometteur pour réduire la quantité de données nécessaires pour la formation, a été appliqué à diverses tâches. Il a été utilisé par plusieurs chercheurs afin de traiter divers problèmes de classification. Un certain nombre d'approches basées sur l'apprentissage actif ont été tentées pour sélectionner les exemples les plus instructifs.

Hasenjager et Ritter [85] ont remarqué que la marge de séparation est étroitement liée à l'apprentissage local et aux classificateurs basés sur les plus proches voisins. Ils ont proposé de faire des requêtes sur les sommets de la mosaïque de Voronoï induite dans l'espace d'entrée lors de l'apprentissage actif. Dans cette contribution, ils se sont intéressés à l'apprentissage actif, ce qui donne à l'apprenant le pouvoir de prélever des échantillons de formation. Les auteurs ont proposé un nouvel algorithme de requête pour les modèles locaux d'apprentissage, une classe d'apprenants qui n'a pas été prise en compte dans le contexte de l'apprentissage actif jusqu'à présent. Leur algorithme de requête est basé sur l'idée de la sélection d'une requête en s'appuyant sur les propriétés géométriques des modèles locaux qui induisent généralement une mosaïque de Voronoi sur l'espace d'entrée, de sorte que les sommets de Voronoï de cette mosaïque se présentent comme des points d'interrogation potentiels.

Lindenbaum et al. [129] ont développé "look-ahead" une méthode d'échantillonnage sélectif, qui plutôt que de considérer l'incertitude des points d'échantillonnage dans l'espace d'entrée, se base sur l'effet de l'étiquetage des points sur leur voisinage. Les auteurs ont proposé un algorithme d'anticipation pour la sélection d'exemples et ont abordé le problème de l'apprentissage actif dans le cadre des classificateurs de plus proche voisin. L'approche proposée repose sur l'utilisation d'un modèle de champ aléatoire pour l'étiquetage d'exemples, ce qui implique un changement dynamique des estimations d'étiquettes pendant le processus d'échantillonnage. Cette approche d'anticipation de l'échantillonnage sélectif est appropriée pour la classification du plus proche voisin. Les auteurs ont commencé par la formalisation du problème de l'échantillonnage sélectif qu'ils ont suivi par un cadre d'anticipation basé sur qui choisit l'exemple suivant (ou séquence d'exemples) afin de maximiser l'utilité espérée du classifieur résultant. Les principaux composants nécessaires à l'application de ce cadre sont une fonction d'utilité pour évaluer les classificateurs et estimations à posteriori de probabilité de classe pour les points dans l'espace d'instances. Ce modèle de champ aléatoire pour la structure de classification de l'espace caractéristique, selon les auteurs, sert de base à une estimation de la probabilité de classe.

Ho et Wechsler [89] ont présenté un article décrivant une stratégie d'apprentissage actif originale en utilisant les mesures de confiance universelles p-valeur et l'algorithmique basée sur le hasard, ainsi que l'inférence transductive. Les critères d'arrêt au

début de l'apprentissage actif sont basés sur le compromis biais-variance pour la classification. Cela correspond à l'instance d'apprentissage où la polarisation frontière devient positive, et exige que l'on passe de l'état actif à la sélection aléatoire des exemples d'apprentissage. Le signe du biais de limite et l'augmentation de l'erreur de classification représentent deux manifestations du même phénomène, à savoir, le surentraînement. La méthode proposée à cet effet est basée sur le compromis biais-variance pour la classification. Les données expérimentales présentées pour valider cette approche, ont été basées sur le KNN-TCM, qui est une technique de confiance transductive augmentée automatique (TCM) utilisant la localité fondée sur les preuves et les k plus proches voisins (KNN).

Une nouvelle approche d'apprentissage actif, pseudo élagage du K plus proches voisins couplé (CKNNPP), est proposée dans [120]. Cette approche est basée sur l'interrogation des exemples par la méthode KNNPP. La méthode applique la technique des k plus proches voisins (KNNPP) pour rechercher k échantillons voisins marqués des échantillons non étiquetés. Lorsque les k échantillons marqués n'appartiennent pas à la même classe, l'échantillon non étiqueté est interrogé et son étiquette est évaluée (validée) par le superviseur, puis il est ajouté à l'ensemble d'apprentissage étiqueté. L'échantillon non étiqueté n'est pas élagué lors de sa sélection et c'est le principe du pseudo élagage. Ces échantillons sélectionnés par KNNPP sont considérés comme étant à proximité ou sur l'hyperplan optimal de classification qui est crucial pour l'apprentissage actif. La méthode CKNNPP est proposée afin que deux échantillons ayant des étiquettes de classe différentes (comme un couple, annoté par le superviseur) peuvent être interrogés par KNNPP et peuvent être ajoutés dans le jeu de formation pour mettre à jour simultanément l'ensemble d'apprentissage à chaque itération. Le CKNNPP est simple, efficace et robuste, et permet de résoudre le problème de la classification des données. En outre, les auteurs ont appliqué un nouveau critère d'arrêt dans la méthode proposée, et ont mis en œuvre un classificateur à travers l'application de Support Vector Machines lagrangien dans les itérations de l'apprentissage actif.

3.10 Conclusion

Dans ce chapitre nous avons présenté l'algorithme KNN en abordant sa facilité, son principe, mais aussi l'un de ses problèmes principaux, qui est la taille critique de l'ensemble d'apprentissage. Dans le but de résoudre ce problème, nous avons exposé l'une des techniques suggérées par différents chercheurs, qui est la sélection d'instances. Nous avons aussi survolé brièvement les travaux les plus connus traitant ce domaine. Enfin, nous avons discuté et critiqué ces propositions, qui, rappelons le, malgré leurs trés bons résultats, ne permettent pas de prouver la minimalité de leurs ensembles résultants. Nous avons traité l'hypothèse que l'utilisation des métaheuristiques ou de l'apprentissage actif dans l'amélioration de ces solutions pourrait garantir une meilleure

convérgence des résultats ; dans cette optique nous avons traité les principaux travaux de sélection d'instances basés sur les métaheuristiques.

Nous avons conclu par la présentation des principales approches de détection d'intrusions qui sont basées sur ledit algorithme.

Deuxième partie

Nos contributions

Chapitre 4

Une approche de détection d'intrusions basée sur l'ACO

4.1 Introduction

Un algorithme d'optimisation par colonies de fourmis (ACO) [52] [56] [57] [80] est essentiellement un système à base d'agents qui simulent le comportement naturel des fourmis, y compris les mécanismes de coopération et d'adaptation [153]. Cette métaheuristique a été démontrée comme étant à la fois robuste et polyvalente. Elle a été appliquée avec succès à différentes gammes de problèmes d'optimisation combinatoire [128] [153] [169].

"Dans une approche standard, les fourmis peuvent trouver le chemin le plus court vers la cible en déposant de la phéromone tout en se déplaçant. D'autres fourmis suivent la trace de phéromone jusqu'à la cible. Les fourmis qui arrivent à prendre le plus court chemin créeront une forte traînée de phéromone plus rapidement que celles qui choisiront un chemin plus long. La phéromone la plus forte attire mieux les fourmis, donc, de plus en plus de fourmis choisissent le plus court chemin jusqu'à ce que finalement toutes les fourmis aient trouvé le chemin le plus court " [86].

Dans un algorithme ACO, chaque fourmi construit progressivement une solution pour le problème cible. Dans ce cas précis, le problème cible est la sélection d'un sous-ensemble d'instances à partir d'un ensemble de formation initiale. Le sous-ensemble sélectionné servira d'ensemble d'entrainement pour le classificateur 1NN.

Nous nous sommes basés sur la coopération et l'organisation des fourmis afin de développer une approche de sélection d'instances permettant de sélectionner l'ensemble d'entrainement nécessaire au 1NN afin de permettre la meilleure classification que possible des paquets entrants, et donc, de réduire le taux de faux positifs.

Nous avons conçu un algorithme de sélection d'instances basé sur les principes des ACO, mais aussi sur le clustering et l'apprentissage supervisé, que nous avons nommé Active Learning Ant Instance Selection, ou ALAIS. Nous avons proposé cet algorithme en améliorant et corrigeant les lacunes et défauts et en faisant évoluer une première

proposition, qui a été Ant-KNN.

Dans ce chapitre nous allons exposer en détails la suite des étapes et des propositions qui ont permit de créer cet algorithme.

4.1.1 Optimisation par Colonies de Fourmis

L'optimisation par colonies de fourmis (ACO) [52] [56] [57] est une approche métaheuristique assez récente basée sur une population de solutions. Elle a été proposée pour résoudre des problèmes d'optimisation combinatoire difficiles. La source d'inspiration de l'ACO est la trace de phéromone créée par le comportement des fourmis réelles qui utilisent des phéromones comme moyen de communication. Par analogie avec l'exemple biologique, l'ACO est basé sur la communication indirecte d'une colonie d'agents simples, représentant des fourmis artificielles. L'information numérique que les fourmis utilisent pour construire des solutions lors de la résolution du problème est que les fourmis s'adaptent lors de l'exécution de l'algorithme afin de refléter leur expérience de recherche.

Les fourmis artificielles utilisées dans l'ACO sont des procédures stochastiques de construction de solutions, qui permettent de construire de manière probabiliste une solution en ajoutant itérativement les composants de la solution à des solutions partielles en tenant compte (i) des informations heuristiques sur le problème à résoudre, (ii) des traces de phéromone (artificielle) qui changent dynamiquement lors de l'exécution.

Après une recherche bibliographique approfondie, nous avons constaté que l'optimisation par colonies de fourmis n'a pas encore été envisagée pour traiter le problème de sélection d'instances.

4.2 Sélection d'instances avec Ant-KNN

Ant-KNN [138] est le premier modèle que nous avons proposé. L'idée de base de cet algorithme a été de définir la plus courte chaîne reliant les éléments d'une classe à l'aide d'une méthode d'optimisation par colonies de fourmis, et ensuite, d'éliminer, à partir de cette chaîne, les éléments qui n'ajoutent pas d'informations supplémentaires pour la classification.

Nous avons cherché, dans ce travail, à conserver le minimum d'éléments pour un classement correct en éliminant les éléments qui n'ajoutent pas d'informations à la classification des autres éléments de l'ensemble initial d'entrainement. Pour cela, nous avons proposé un algorithme à trois étapes : la création de la chaîne d'associés (voir 4.2.1), l'application de l'algorithme de réduction et l'application de l'algorithme de filtrage de bruit. Nous avons fait appel à l'optimisation par colonies de fourmis [52] [56] [57] [31] afin de créer la chaîne d'associés reliant les instances de même classe à travers le chemin le plus court possible.

4.2.1 Création de la chaîne d'associés

D'après la définition de Wilson [190], l'associé d'un prototype P est son voisin le plus proche de la même classe que lui.

Une chaîne d'associés C_i d'une classe W_i est définie comme étant la séquence $\{X_{i0}$, x_{i1}, x_{i2}, ..., $x_{ik-1}\}$ où k = | w_i |, l'élément X_{i0} représente l'élément central de la classe W_i, et x_{ij} est le prochain élément de la même classe pouvant être atteint par le chemin le plus court à partir de x_{ij-1}. La séquence $\{d_{i0}$, d_{i1}, d_{i2}, ..., $d_{ik-2}\}$, où la distance euclidienne entre deux éléments x_{ij+1} et x_{ij} défini par $d_{ij} = \|x_{ij} + 1 - x_{ij}\|$ est celle utilisée par l'algorithme de colonie de fourmis pour définir les chemins entre les éléments de la classe. La chaîne s'arrête lorsque tous les éléments de la classe ont été parcourus et que l'élément suivant est l'élément racine de cette chaîne.

Conventions

X : ensemble d'éléments ;

n =| X | : nombre d'éléments ;

b_i (t) : nombre de fourmis dans l'élément i a l'instant t ;

n=$\sum_{x \in i} b_i$:nombre total de fourmis ;

$n_{ij} = \frac{1}{d_{ij}}$: visibilité d'un élément j à une fourmi quand elle est sur l'élément i ;

τ_{ij} (t) : valeur de la phéromone sur l'arc (i, j) ;

A tout instant t, chaque fourmi choisit un élément de destination selon un mode de choix déterminé. Toutes les fourmis sont placées à l'instant t +1 sur un élément de leur choix. Une itération de l'algorithme est définie par l'ensemble des mouvements de l'ensemble de la colonie entre t et t +1. Ainsi, après n itérations, la colonie entière a effectué un circuit hamiltonien [83] [90].

Choix de transitions

La fourmi k positionnée sur l'élément i va choisir sa destination j en fonction du taux de visibilité n_{ij} et du taux de phéromone τ_{ij}. Ce choix est fait avec une probabilité $Prob_{ij}^k$ de choisir l'élément j :

$$Prob_{ij}^k = \begin{cases} \frac{\tau_{ij}(t) * n_{ij}}{\sum_{i \in N_i^k(t)} \tau_{il} * n_{il}} if j \in N_i^k \\ 0 else \end{cases} \tag{4.1}$$

Où N_i^k est l'ensemble des éléments que la fourmi k installée sur l'élément i n'a pas encore visité à l'instant t.

Dépôt de phéromones

Une fourmi dépose un montant de phéromone sur chaque arête τ_{ij}^k appartenant à son chemin :

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \frac{Q}{L^{k}(t)} if(i,j) \in T^{k}(t) \\ 0else \end{cases} \tag{4.2}$$

Où$T^k(t)$ représente le tour fait par la fourmi k à l'itération t, $L^k(t)$ est la longueur du trajet et Q un paramètre de réglage.

Mise à jour de la phéromone

A la fin de chaque itération de l'algorithme, les phéromones déposées dans les précédentes itérations par les fourmis s'évaporent suivant un indice : $\rho * \tau_{ij}(t)$, où ρ $\in$ [0,1].

A la fin de chaque itération la somme des phéromones qui ne se sont pas évaporées et celles qui ont été déposées est calculée :

$$\tau_{ij}^{k}(t+1) = \ (1-\rho)P_{ij}^{k}(t) + \sum_{k=1}^{k} \Delta P_{ij}^{k}(t) \tag{4.3}$$

La création des chaines d'associés d'un ensemble à réduire se base sur les étapes suivantes :

Pour chaque classe de l'ensemble de la formation : créer une chaîne reliant tous les éléments le long du chemin le plus court :

1. Pour chaque élément, calculer ses distances par rapport à tous les autres éléments de sa classe.
2. Initialiser le tour en désignant la racine (le centre de la classe).
3. Créer autant de fourmis que d'éléments dans la classe
4. Chaque fourmi choisit son élément de destination en fonction de la formule (4.1).
5. Chaque fourmi calcule la valeur $L^k(t)$.
6. Les valeurs τ_{ij}^{k} (t) sont calculées selon la formule (4.2).
7. Les valeurs des phéromones τ_{ij} (t) sont mises à jour en fonction de la formule (4.3). En d'autres termes, la fourmi retourne sur ses pas, dans la direction opposée, tout en déposant de la phéromone.
8. Le meilleur tour trouvé jusqu'à maintenant est sélectionné, et sauvegardé.
9. Les mémoires des fourmis (liste des éléments visités) sont effacées.

4.2.2 Définition des paramètres

Une grande faiblesse des algorithmes basés sur l'optimisation par colonies de fourmis est le nombre élevé de paramètres en jeu. De ce fait, il est très difficile de les définir subtilement.

En l'absence d'un modèle mathématique, les seules justifications à la disposition des chercheurs pour fixer les paramètres d'un algorithme ACO sont des résultats expérimentaux. Nous avons donc défini les différents paramètres de l'algorithme Ant-KNN par expérimentations, que nous avons fixé comme suit :

En fixant le paramètre d'évaporation ρ à 0.5 et c à 0, seule la stratégie gloutonne est à l'oeuvre au début de l'algorithme : ceci est souhaitable, car les pistes de phéromones ont été généré artificiellement et n'ont donc aucune signification physique.

Ensuite, grâce au paramètre p>0, les fourmis commencent progressivement à exploiter les valeurs τ_j^i au fur et à mesure que les pistes de phéromones se renforcent. Afin de ne pas être esclave des choix anciens, il faut que la visibilité ait encore un impact. Ceci explique le choix selon lequel ρ = 0,5. En suite en fixant le paramètre d'évaporation et c, nous avons pu fixer α= 1, β= 1, et Q= 1

4.2.3 L'algorithme de réduction

L'idée de base de l'approche de réduction proposée est la suivante : Pour chaque classe w_i dans l'ensemble de la formation, nous construisons la chaîne d'associés, C_i, correspondante. Un élément x_{ij} d'une chaîne est ignoré s'il n'ajoute pas d'information supplémentaire à la classification des éléments de l'ensemble de formation initial.

Pour cela, les k voisins de chaque élément d'une chaîne sont évaluées en commençant par ceux de l'élément racine x_{i0}.

L'ensemble final F_i est créé progressivement. Il est initialisé à x_{i0}. L'ensemble des voisins couverts N_i comprends les k voisins des éléments de F_i. Il doit contenir une seule occurrence de chaque voisin couvert par les éléments de l'ensemble final réduit F_i, il est initialisé aux k voisins de l'élément x_{i0}. La chaîne est évaluée élément par élément, et un élément x_{ij} est ajouté à F_i s'il couvre au moins un voisin inexistant dans N_i, dans le cas échéant, les voisins inexistants dans N_i sont ajoutés. Sinon, l'élément est ignoré. Ceci est répété jusqu'à la fin de la chaîne C_i.

Les principales étapes de l'algorithme de réduction sont décrites comme suit :

1. Pour chaque classe w_i, construire C_i sa chaîne d'associés.
2. Pour chaque élément x_i appartenant à C_i, évaluer k_{xi} (ses k plus proches voisins)
3. pour i=1 à $|C_i|$

   - S'il existe au moins un y appartenant à k_{xi} non existant dans N_i

   F_i=F_i+x_i

   N_i=N_i+y

   Finsi
4. Fin pour
5. Créer F l'ensemble condensé final en combinant toutes les chaînes d'associés réduites de toutes les classes.

4.2.4 Critère d'arrêt

L'algorithme est répété jusqu'à ce que l'un des deux critères soit respecté :

- La taille de la classe est inférieure à un seuil égal à k (le nombre de voisins évalués lors de la réduction).
- Ou lorsque la précision diminue en dessous d'un seuil défini (80%)

4.2.5 Filtrage de bruit

La dernière étape du premier prototype proposé est le filtrage de bruit.

Après l'étape de réduction l'ensemble résultant est filtré. Les éléments causant la mal classification des éléments de l'ensemble de formation initial sont éliminés durant cette étape.

Tous les éléments dans l'ensemble de formation initiale " O " sont reclassés en utilisant le nouvel ensemble condensé F comme ensemble de la formation pour le classifieur 1-NN. Quand un élément mal classe un autre élément (ou plus), cet élément est retiré de manière temporaire. Un reclassement des éléments de O est refait en utilisant l'ensemble condensé sans l'élément éliminé temporairement. Si le taux de classement offert en utilisant l'ensemble filtré est supérieur ou égal au taux de classification utilisant l'ensemble condensé avant le filtrage, l'élément est définitivement supprimé, sinon, si ce taux est inférieur à celui obtenu en utilisant l'ensemble condensé avant le filtrage, l'élément est maintenu.

Les principales étapes de l'algorithme de filtrage de bruit peuvent être décrites comme suit :

1. Classer les éléments dans O en utilisant 1-NN et F
2. calculer class_rate
3. Pour tous les éléments j qui mal classent un élément dans O faire
   - reclasser O en utilisant 1-NN et F-{j}
   - calculer class_rateN
4. Si (class_rateN $\geq$ class_rate) alors F = F-j

4.2.6 Evaluation des performances

Vu que l'optimisation par colonies de fourmis n'a, à notre connaissance, jamais été utilisée pour la sélection d'instances jusqu'à ce jour, nous avons proposé d'évaluer les performances de notre approche d'abord par rapport à l'aspect de la sélection. Pour cela nous avons choisi de tester Ant-KNN sur cinq benchmarks différents téléchargés à partir du UCI Machine Learning Repository [2]. Ces ensembles de données sont les suivants : Ecoli, Glass, Heart (Cleveland), Heberman's Survival et Iris. Le tableau 6.5 présente les détails de ces ensembles de données.

Ces ensembles de données sont très utilisés dans la litérature [104][96][178].

TABLE 4.1 – Détail des cinq ensembles de données utilisés dans l'évaluation

Ensemble	Nom	♯Instances	Dimentionnalité	♯Classes
Ecoli	Ecoli	336	9	8
Glass	Glass Identification	214	11	6
Heart (C)	Heart Cleveland	303	14	5
Heberman	Heberman's Survival	306	4	2
Iris	Iris	150	4	3

TABLE 4.2 – Résultats de l'évaluation de Ant-KNN

Ensemble	Mesure	KNN standard	Ant-KNN	TRKNN	DROP3
Ecoli	Réduction	**	92.41	55.74	**92.52**
	Précision	100	81.18	72.66	79.55
Glass	Réduction	**	**89.59**	39.28	76.12
	Précision	73.83	100	81.42	65.02
Heart (C)	Réduction	**	**97.81**	26.46	87.24
	Précision	81.19	93.33	58.67	80.84
Hebermani	Réduction	**	**88.05**	44.38	80.9
	Précision	86.66	93.33	64.31	66.60
Iris	Réduction	**	81.64	45.21	**85.19**
	Précision	94.00	100	93.33	95.33
	MAcc	87.13	93.56	74.07	77.46
	Perf	+7.37	**	+26.31	+20.78

Description du plan d'évaluation

Le plan d'évaluation a été conçu pour aider à vérifier l'exactitude et l'efficacité de la méthode proposée.

Le plan stipule que l'ensemble de formation est divisé en dix parties égales. Neuf parties forment l'ensemble d'apprentissage à réduire et la dixième partie est utilisée pour la validation. Pour créer la chaîne d'associés, le nombre de voisins évalués a été défini expérimentalement pour Ecoli, Glass, Heart (Cleveland) et Iris à k = 16, et pour la Heberman's Survival à k = 9. Nous avons utilisé un classificateur 1-NN pour la classification.

Résultats

Afin de valider notre travail, nous avons comparé les résultats obtenus des tests à ceux obtenus avec le KNN standard et deux des algorithmes présentés dans le chapitre 3 qui sont TRKNN et Drop3. Le taux de réduction, la précision de classification, le taux moyen de précion (MAcc) et la pérformance de chaque algorithme sont présentés dans le tableau 4.2.

Le tableau 4.2 montre que Ant-KNN présente la meilleure précision moyenne et le meilleur taux de réduction moyen. Pour les cinq benchmarks Ant-KNN présente les meilleurs taux de classification (81,18%, 100%, 93,33%, 93,33% et 100%) par rapport à TRKNN et Drop3.

Par rapport au KNN standard, Ant-KNN présente la meilleure précision pour les quatre ensembles de données : Glass, Heart (C), Heberman et Iris.

Le KNN standard présente un taux d'exactitude plus élevé de 18,82% par rapport au taux présenté par le Ant-KNN.

Ant-KNN présente les meilleurs taux de réduction pour quatre ensembles de données : Ecoli, Glass, Heart (C) et Heberman, et se place juste après Drop3 pour le taux de réduction de Iris par un faible écart moyen de 3,55%, ce qui représente une très faible différence.

Analyse et discussion

L'algorithme a été réitéré entre une et quatre fois pour les différents ensembles de données afin d'améliorer les résultats. Nous avons effectué les analyses suivantes :

- **Analyse de classification.** Les résultats ont montré que l'algorithme présentait des taux élevés de précision de la classification moyennant les 93.56%, ce qui est supérieur aux taux moyens présentés par les algorithmes utilisés pour la comparaison.
- **Analyse de réduction.** L'algorithme proposé offre une bonne capacité de réduction d'une moyenne de 10,10%. Cela est dû à la création de la chaîne d'associés qui permet d'éliminer uniquement les éléments non-actifs pour le classement.

Le point négatif de cet algorithme est sa grande complexité égale à O $(N)^2$, qui est un obstacle à son utilisation sur de grands ensembles de données. Cette méthode est conçue pour des applications dans la classification de données bruitées pour des ensembles de moyenne ou petite taille.

Ce qui rend aussi l'utilisation de cet algorithme inadéquate pour les ensembles volumineux, c'est le nombre d'étapes et la complexité de chacune. A la seconde étape, qui est la réduction, la chaine d'associés est revisitée élément par élément, et les k voisins de chaque élément sont recherchés. Ceci a conduit à une complexité temporelle importante quand l'ensemble est de grande taille. Nous avons donc décidé qu'incorporer la réduction durant la construction de la chaine, améliorerait notre proposition, et qu'opter pour une base de sélection des éléments autre que l'évaluation des k voisins serait plus performante. Dans cette optique nous avons conçu notre seconde approche basée sur l'optimisation par colonies de fourmis que nous avons nommé Ant-IS.

4.3 Sélection d'instances basée sur les fourmis (Ant-IS)

Afin de construire un classificateur KNN efficace deux objectifs principaux doivent être atteints : 1) réaliser un taux de précision élevé et 2) minimiser l'ensemble d'instances pour permetre au classificateur d'être operationnel même avec de grands ensembles de données. Ces objectifs ne sont pas indépendants. Nous avons tenté à travers

cette seconde proposition d'aborder la question de minimisation des besoins en ressources de calcul pour la technique de KNN, tout en conservant une grande précision de classification. A cette fin, nous avons mis au point une méthode de sélection d'instances fondée sur les principes d'optimisation par colonies de fourmis, que nous avons appelée Ant Instance Selection algorithm [137] (Ant-IS) [135].

Une fourmi va explorer toutes les instances lors de la construction de son chemin. Pour chaque instance choisie, la fourmi va décider de la sélectionner ou non, comme une partie de son propre sous-ensemble.

Dans cette approche, nous nous somme basés sur les principes de l'ACO dans le choix des instances à intégrer dans l'ensemble d'apprentissage. Ceci a été réalisé grâce à l'introduction de deux nouveaux concepts : la recherche de voisinage et la recherche du meilleur chemin.

1. La recherche de voisinage : L'objectif du classificateur est de déterminer la classe d'une instance inconnue. Pour atteindre ce but, nous tenons compte des instances qui sont les plus proches de l'instance non classée. Si toutes ces instances considérées comme très proches voisines appartiennent à la même classe de l'instance la plus proche de l'instance inconnue, cette dernière est correctement classée. Donc, si une instance inconnue appartient en fait à une classe donnée, ses k plus proches voisins devraient logiquement faire partie de cette même classe. Dans une même classe, nous nous concentrons sur le maintient des instances importantes pour la classification, qui, en même temps, sont les plus proches que possible les unes des autres. Une fourmi se déplace à partir d'une instance vers l'instance la plus proche. Cette dernière doit se trouver, en même temps, aussi proche que possible de l'instance de départ. Cette démarche est guidée par trois paramètres principaux, comprenant la visibilité et le taux de phéromone, mais aussi la distance par rapport à l'instance racine, qui doit être minime. Le choix de l'instance se fait lors de la construction du voisinage. Lors du passage d'une instance à l'autre, la fourmi décide aléatoirement si elle souhaite inclure cette instance ou non dans son ensemble réduit.

2. Recherche parallèle du meilleur chemin : le nombre de fourmis utilisées détermine le nombre de solutions étudiées. C'est pourquoi, dans ce travail, nous avons proposé d'utiliser autant de fourmis que d'instances dans l'ensemble à réduire. A l'initialisation chaque fourmi est située sur une instance différente. Ensuite, toutes les fourmis recherchent leurs chemins optimaux en même temps. Lors de la recherche d'un chemin une fourmi sélectionne les instances à inclure dans sa solution. Quand une fourmi se déplace sur une instance, elle choisit de l'inclure comme partie de sa solution. Prendre chaque instance comme un point de départ potentiel, permet d'explorer toutes les probabilités de pistes sans que la recherche ne soit forcée par une contrainte de départ. Lorsque toutes les instances ont été visitées, toutes

les fourmis auront fini de déterminer leurs chemins et donc de sélectionner les instances.

Une fois la sélection terminée, les différents chemins sont évalués, et celui permettant la meilleure classification est choisi comme solution finale.

Nous détaillerons les principes de l'approche proposée dans les prochains paragraphes.

Conventions

Pour expliquer le raisonnement dans le développement de cette approche nous avons besoin de quelques conventions. Nous allons faire appel à certaines conventions déjà utilisées lors de la section précédente et nous allons définir de nouvelles, comme suit :

T : ensemble d'instances ;

n =| T | : nombre d'instances ;

b_i (t) : nombre de fourmis dans l'instance i à l'instant t ; n=$\sum_{x \in i} b_i$:nombre total de fourmis ;

$n_{ij} = \frac{1}{d_{ij}}$: visibilité d'une instance j à une fourmi quand elle est sur l'élément i ;

Pij (t) : valeur de la phéromone sur l'arc (i, j) ;

C : une matrice (b_i (t) x n) contenant la validation des villes de destination ;

a_j^k : paramètre aléatoire $\in$ 0,1.

La matrice C contient une ligne pour chaque fourmi. Chaque rangée comprend une cellule pour chaque instance de l'ensemble à réduire. Une fourmi k met le paramètre a_j^k à 1 si elle décide d'inclure l'instance j à sa sélection, et met a_j^k à 0 sinon.

A l'initialisation, tous les chemins entre deux instances sont notés avec la même valeur initiale de phéromone de sorte que toutes les destinations aient une chance équitable d'être choisies au départ. L'algorithme crée autant de fourmis que d'instances, et commence par placer une fourmi sur chaque instance. A tout moment t, chaque fourmi choisit une instance comme destination. Si la fourmi se déplace sur cette instance, elle la sélectionne pour faire partie de son sous-ensemble.

Toutes les fourmis sont placées à l'instant t +1 sur une instance de leur choix. Une itération de l'algorithme est définie par l'ensemble des mouvements de l'ensemble de la colonie entre les instants t et t +1.

L'algorithme s'arrête lorsque toutes les cellules de C ont été évaluées.

4.3.1 Mise en œuvre

- Chaque fourmi a une mémoire représentée par une liste d'instances déjà choisies, qui peuvent avoir été visitées ou non. Cela garantit qu'aucune fourmi ne puisse choisir deux fois la même instance.
- La mémoire de chaque fourmi est effacée lorsque le cycle est terminé.

Choix des transitions

La fourmi k positionnée sur l'instance i choisira sa destination j en fonction du taux de visibilité n_{ij}, la phéromone P_{ij} et un paramètre aléatoire a_j^k. Ce choix est fait en utilisant la probabilité $FProb_{ij}^k$.

Le choix de la destination suivante passe par deux étapes. Une fourmi choisit, d'abord, la destination probable. Ensuite, dans une seconde étape, elle valide ce choix comme destination finale ou l'annule et choisit une nouvelle destination probable.

Étape 1 : La fourmi k positionnée sur l'instance i va, d'abord, choisir sa destination probable j en fonction du taux de visibilité n_{ij} et la phéromone P_{ij}.

Ce choix se fait avec une probabilité de choisir l'instance j, calculée comme suit :

$$Prob_{ij}^k = \begin{cases} \frac{P_{ij}(t)*n_{ij}}{\sum_{i \in N_i^k(t)} P_{il}*n_{il}} if j \in N_i^k \\ 0 else \end{cases} \tag{4.4}$$

Où N_i^k est l'ensemble des instances que la fourmi k n'a pas encore visité à l'instant t (à partir de l'instance i).

Étape 2 : Une fois le choix de la destination probable fait, la fourmi k tire aléatoirement un paramètre a_j^k dans l'ensemble {0, 1} pour valider ou non la destination probable choisie.

Si a_j^k est égal à 1, l'instance de destination possible sélectionnée est utilisée en tant que destination et C (k, j) est mis à 1. Sinon, si a_j^k est égal à 0, l'instance de destination probable sélectionnée n'est pas validée comme destination, elle est marquée comme non sélectionnée, et la cellule C (k, j) est fixée à 0.

La fourmi k, calcule ensuite la probabilité $Prob_{ij}^k$ pour l'instance visible suivante. La fourmi k répète cette étape jusqu'à l'obtention d'une instance de destination valide.

La probabilité finale à choisir une instance en tant que destination suivante est donnée par la formule suivante :

$$FProb_{ij}^k = a_{ij} * Prob_{ij}^k \tag{4.5}$$

La fourmi répète cette étape jusqu'à l'obtention d'une instance de destination valide.

dépôt de phéromone

Une fourmi dépose une quantité de phéromone P_{ij}^k sur chaque arc de son chemin

$$\Delta P_{ij}^k(t) = \begin{cases} \frac{Q}{L^k(t)} if (i,j) \in T^k(t) \\ 0 else \end{cases} \tag{4.6}$$

Où $T^k(t)$ représente le chemin fait par la fourmi k à l'itération t, $L^k(t)$ est la longueur du chemin et Q un paramètre de réglage.

Mise à jour de la phéromone

A la fin de chaque itération de l'algorithme, les phéromones déposées lors des précédentes itérations s'évaporent suivant un taux : $\rho * P_{ij}(t)$, où $\rho \in [0,1]$.

A la fin de l'itération, la somme des phéromones non évaporées et celles qui ont été déposées est calculée comme suit :

$$P_{ij}^{k}(t+1) = (1-\rho)P_{ij}^{k}(t) + \sum_{k=1}^{k} \Delta P_{ij}^{k}(t) \tag{4.7}$$

Validation du tour

Après que les fourmis aient terminé leurs tours respectifs, le meilleur tour est calculé.

Un tour k contient la liste des instances validées par une fourmi k. Cette liste servira d'ensemble de formation au classificateur 1NN afin de reclasser les instances de l'ensemble d'entrainement. Le taux de classification, acc_k, est calculé selon la formule 5, et chaque tour est associé à son taux de classification. Le meilleur tour est celui qui permet d'avoir le taux de classification le plus élevé.

La précision de la classification d'un tour k est calculée comme suit :

$$acc_k = \frac{N_{ck}}{|S_t|} \tag{4.8}$$

Où N_{ck} est le nombre d'instances correctement classées par le classificateur 1NN utilisant les instances de la liste k comme ensemble d'entraînement, et S_t est le nombre total d'instances de test.

4.3.2 Fonctionnement de l'algorithme

Les principales étapes de l'algorithme sont les suivantes :

1. Pour chaque instance, calculer les distances vers toutes les autres.
2. Initialiser le tour par l'initialisation de la matrice C.
3. initialiser les valeurs de phéromone sur tous les arcs.
4. Créer autant de fourmis que d'instances
5. Chaque fourmi choisit son instance de destination probable, selon les formules (4.4) et (4.5).
6. chaque fourmi k met à jour la matrice de validation C en fonction du paramètre a_j^k :
   - Si $\text{a}_j^k = 1$, l'instance de destination j est choisie et C (k, j) = 1.
   - Dans le cas contraire, C (k, j) = 0, retour à 5.

7. Chaque fourmi calcule la valeur $L^k(t)$.
8. Les valeurs P_{ij} (t) sont calculées selon la formule (4.6).
9. Les valeurs de phéromone P_{ij} (t) sont mises à jour en fonction de la formule (4.7). En d'autres termes, la fourmi repart dans la direction opposée tout en déposant des phéromones.
10. Si toutes les fourmis ont terminé leurs solutions aller à 11, autrement revenir à 5.
11. Trouver la fourmi qui a construit le tour avec le taux de classification le plus élevé en utilisant la formule (4.8) (ie on recherche la fourmi k tel que $acc_k = \max(acc_k)$). Si ce tour est meilleur que celui trouvé jusqu'ici, il est stocké.
12. Les mémoires des fourmis (liste des instances visitées) sont effacées.

L'algorithme Ant-IS peut être présenté par le pseudo-code 4.3.2 ci-aprés.

4.3.3 Définition des paramètres

Tout comme pour le paramétrage de l'algorithme Ant-KNN, nous avons défini les paramètres de l'algorithme Ant-IS par expérimentation, et ceci comme suit : α à 2, β à 6, Q à 2.71 et ρ à 0.5

4.3.4 Evaluation des performances

Tout comme pour l'algorithme précédent, nous avons d'abord voulu nous assurer des performances de réduction que pouvait offrir la nouvelle version du prototype avant de le tester sur la sélection d'un ensemble d'entrainement pour la classification d'attaques.

L'algorithme de sélection d'instances proposé a été appliqué à diverses données réelles et synthétiques toutes téléchargées à partir de l'UCI (Université de Californie et Irvine) Machine Learning Repository [2]. Les résultats ont été comparés à ceux offerts par d'autres approches connues.

Les données utilisées

Les performances du Ant-IS ont été testées sur onze benchmarks de la vie réelle dont sept petits jeux de données et quatre ensembles de moyenne dimension. Comme cité précédemment, ces ensembles de données ont tous été largement utilisés par de nombreux algorithmes d'apprentissage automatique ([104][96][178]). Ces benchmarks sont les suivants : Bupa, Glass, Heart (Cleveland), Iris, Pima, Tic Tac Toe, Vote, Chess, German, Sat image et Splice.

En choisissant ces onze bases de données, nous avons essayé de tester les performances de la méthode proposée dans onze domaines différents, ce qui permet d'avoir des conclusions plus générales.

Les principales caractéristiques des ensembles de données utilisés dans les expériences sont résumées dans le tableau 4.3 La première colonne de ce tableau donne les noms

5 Algorithme Ant-IS

Entrée : Première série T
Sortie : L'ensemble final réduit F
Début

1. Initialiser C ;
2. n = | T | / / Initialisation nombre de fourmis ;
3. Pour k = 1 à n faire
   – Initialisation P_{ij} ;
4. Pour finir ;
5. Pour k = 1 à n faire
   (a) i = k / / chaque fourmi débute à une instance différente
   (b) Répéter
      i. Calculer n_{ij} ;
      ii. Calculer $Prob_{ij}^{k}$ par le biais de la formule (4.4) ;
      iii. Choisir l'instance j avec la meilleure probabilité ;
      iv. Choisir aléatoirement une valeur pour a_j^k ;
      v. Calculer $FProb_{ij}^{k}$ en utilisant la formule (4.5) ;
      vi. Si ($FProb_{ij}^{k}$!= 0) Alors i = j ; / / passage à j ;
      vii. Sinon Répéter
         A. j = l'instance avec la prochaine Probk ;
         B. Choisir aléatoirement a_j^k ;
      viii. Jusqu'à ce que le passage à une instance est possible ;
      ix. Mettre à jour P_{ij} au moyen de la formule (4.7) ;
   (c) Jusqu'à ce que toutes les cellules C sont évaluées par 1 ou 0 ;
6. Fin pour ;
7. Pour chaque liste de fourmi
   – Calculer acc_k au moyen de la formule (4.8) ;
8. Fin Pour ;
9. Choisir F la liste avec max (acc_k) ;
10. Retour F ;

Fin.

TABLE 4.3 – Détail des ensembles de données utilisés pour l'évaluation

Ensemble	Nom	♯Instances	Dimentionnalité	♯Classes
Bupa	Liver Bupa	345	7	2
Chess	Chess End-Game	3196	36	2
Glass	Glass Identification	214	11	6
German	German Credit Data	1000	25	2
Heart	Heart Cleveland	303	14	5
Iris	Iris	150	4	3
Pima	Pima Indians Diabetes	768	9	2
Sat	Landsat Satellite	6435	36	7
Splice	Primate splice-junction gene	3190	62	3
Tic	Tic Tac Toe End Game	958	9	2
Vote	1984 US Congressional Voting Records	435	17	2

d'abréviation du jeu de données. La deuxième colonne donne le nom complet de l'ensemble de données, tandis que le reste des colonnes indiquent, respectivement, le nombre d'instances, le nombre d'attributs et le nombre de classes de chaque ensemble de données.

Résultats

Les expériences ont été limitées à des algorithmes qui choisissent un sous-ensemble d'instances appartenant à l'ensemble d'entrainement initial, et en construisent un nouvel ensemble d'entrainement et ne modifient pas les instances elles-mêmes. Notre technique proposée a été testée sur les onze ensembles de données. Les résultats ont été comparés à ceux offerts par un nombre important de techniques de réduction, toutes étant décrites dans le chapitre 3 Celles incluses dans ces expériences sont : IFS-CoCo, CHC, SSGA, GGA, Drop3, ICF et Relief.

Tous ces résultats ont été comparés à ceux obtenus par le 1NN standard sans sélection d'instances, ceci nous a permit d'analyser d'éventuelles améliorations.

L'algorithme Ant-IS utilise k = 1 et la fonction de la distance Euclidienne standard. Le plan expérimental a été conçu pour vérifier l'exactitude et l'efficacité de la méthode proposée.

Pour la validation, nous avons utilisé un modèle de validation croisée ou cross validation appelée "holdout method" [12]. Afin de mener à bien les expériences, chaque ensemble de données a été divisé en dix parties égales. A chaque technique de réduction est donné un ensemble d'apprentissage constitué par neuf portions (soit 90% de l'ensemble de données) à partir duquel la technique de réduction doit renvoyer un sous-ensemble. La dixième partie (ie, les 10% restants) est utilisée comme ensemble de test pour la validation.

Les résultats de ces expérimentations sont exposés dans le tableau 4.4 Dans ce

tableau deux mesures importantes sont présentées pour chaque algorithme, pour les onze ensembles de données : le taux de réduction et celui de précision.

TABLE 4.4 – Résultats d'évaluation de Ant-IS

Ensemble	Measure	1NN	Ant-IS	IFS-CoCo	CHC	SSGA	GGA	DROP3	ICF	Relief
Bupa	Réduction	**	84.40	97.97	94.10	90.84	93.39	70.05	70.46	51.66
	Précision	61.08	65.71	67.05	60.64	62.28	68.25	60.86	63.16	52.46
Glass	Réduction	**	64.07	98.60	95.92	89.60	92.49	76.35	67.75	17.78
	Précision	73.61	99.99	69.60	67.81	65.80	65.95	65.02	77.79	76.25
Heart	Réduction	**	65.94	98.02	95.75	94.76	96.72	87.24	79.72	72.31
(C)	Précision	53.14	96.66	57.99	55.56	53.60	55.91	80.84	55.82	57.65
Iris	Réduction	**	93.88	95.93	95.83	95.01	95.60	85.19	64.22	10.00
	Précision	93.33	99.99	95.33	95.33	94.22	87.97	95.33	93.33	94.67
Pima	Réduction	**	99.86	98.38	97.03	94.56	94.42	83.10	77.29	77.50
	Précision	33.26	77.10	72.27	72.38	72.20	72.71	75.01	69.32	67.85
Tic	Réduction	**	78.81	98.61	97.33	90.62	91.14	92.87	70.90	54.44
	Précision	73.07	99.99	83.51	82.11	86.79	78.87	69.31	72.97	65.35
Vote	Réduction	**	64.55	99.11	96.69	97.36	97.44	94.89	87.51	35.62
	Précision	92.16	97.00	94.62	94.47	93.86	93.54	95.87	89.64	92.18
Chess	Réduction	**	89.52	81.64	98.19	94.95	94.23	-	-	-
	Précision	84.70	91.37	96.56	85.27	85.00	86.56	-	-	-
German	Réduction	**	66.40	97.39	95.78	93.81	93.37	-	-	-
	Précision	28.00	97.00	23.35	21.29	22.84	25.26	-	-	-
Sat	Réduction	**	61.38	71.65	99.36	97.99	98.14	-	-	-
	Précision	90.58	80.95	91.29	87.87	89.74	90.98	-	-	-
Splice	Réduction	**	96.61	92.96	99.02	95.42	95.65	-	-	-
	Précision	74.95	93.54	86.83	71.16	73.35	76.80	-	-	-
	Moy(Red)	**	76.02	93.66	96.72	94.08	94.78	84.23	73.97	45.61
	Moy(Acc)	68.90	90.84	76.21	72.17	72.69	72.98	77.45	74.57	72.34
	Perf	+31	**	+19	+25	+24	+24	+21	+29	+25

La précision est définie comme le nombre de classifications correctes par rapport au nombre total de classifications. Elle a été de loin la métrique la plus couramment utilisée pour évaluer les performances des classificateurs au fil des années [1] [11]. Le taux de réduction est défini par le rapport des données sélectionnées par l'algorithme sur l'ensemble initial de données.

Le taux de réduction moyen (Moy(Red)) et la précision moyenne (Moy(Acc)) de chaque algorithme sont présentés dans le même tableau.

La performance globale d'un algorithme de réduction est caractérisée par sa précision moyenne de classification. Pour cela, nous avons jugé nécessaire d'établir une comparaison des performances globales. Nous avons comparé les performances de l'Ant-IS à tous les autres algorithmes en calculant l'amélioration de la précision (positif) et la détérioration de la précision (négatif) et nous avons présenté les résultats sur la dernière ligne du tableau 4.4 Ces taux sont calculés par la formule 4.9 comme suit[148] :

$$Perf = ((Ant - IS - Other)/Other) * 100 \tag{4.9}$$

Analyse et discussion

Plusieurs observations peuvent être faites à partir des résultats du tableau 4.4.

Comme la sélection d'instances pour Relief se fait à travers un échantillon de taille fixe, il présente les pires taux de réduction et de classification pour la plupart des ensembles de données.

Drop3 et ICF présentent des résultats presque identiques en raison de la similitude de leurs principes. Ces deux algorithmes présentent une plus grande précision et une réduction plus faible comparés aux autres algorithmes, à l'exception de l'IFS-CC. Cela est dû au fait que ce sont des méthodes heuristiques, de sorte que le calcul de leur résultat est plus axé sur le taux de classification et moins sur le taux de réduction.

CHC, VEC et SGA sont trois algorithmes évolutionnaires basés sur la recherche (CHC) et les principes génétiques (GGA et SSGA). Les trois algorithmes sont basés sur l'évaluation de leur population en termes de sélection. Ce qui permet à ces algorithmes d'offrir un meilleur taux de réduction, mais un moindre taux de précision.

IFS-CoCo est un algorithme récent et très puissant, qui permet de sélectionner le sous-ensemble le plus approprié d'instances, tout en augmentant la précision de la classification. Cet algorithme présente des résultats supérieurs à la fois à ceux présentés par les algorithmes heuristiques cités et les algorithmes métaheuristiques.

Ant-IS peut être situé entre les deux groupes précédemment cités, où il tire de chaque groupe ses avantages. Nous devons d'abord rappeler que le Ant-IS tente de réduire l'ensemble d'apprentissage autant que possible sans pour autant négliger la précision de la classification. Ant-IS présente une plus grande précision que celles offertes par tous les algorithmes comparés, car il ne conserve que les instances efficaces pour

la classification. Cependant, il présente un taux moyen de réduction plus petit que celui proposé par les autres algorithmes évolutionnaires, car quand Ant-IS évalue les instances efficaces de classification, il tend à maintenir un nombre important de ces dernières.

L'algorithme Ant-IS surpasse tous les algorithmes comparés en termes de précision de classification. Il présente également un taux moyen de réduction dépassant les 76%, ce qui est un taux acceptable pour de nombreuses applications, où le taux de classification est le critère essentiel.

Cependant, bien qu'il soit plus rapide que son précédant, son application sur de grands ensembles de données tels que DARPA reste impossible à cause du nombre important de fourmis qui doivent être impliquées dans la recherche de la solution. Ceci nous a conduits à penser à améliorer cet algorithme en proposant un choix stratégique du nombre et des emplacements de départ des fourmis en invoquant les techniques de clustering.

Cette hypothèse a permis la création d'un troisième algorithme que nous avons nommé Clustering Ant-IS ou Clust Ant-IS.

4.4 Clustering Ant-IS

Ant-IS n'a maintenant été appliqué que dans le domaine de pattern recognition ou de classification, et n'a pas encore été évalué dans le domaine de la détection d'intrusions. Afin de pouvoir appliquer cet algorithme pour la classification d'intrusions, et afin de minimiser au maximum le nombre d'alertes fausses positives et fausses négatives, l'ensemble d'entrainement doit être aussi performant que possible. Aussi, afin d'espérer utiliser cet algorithme dans la détection d'intrusions en temps réel, l'ensemble d'apprentissage efficace doit être créé en tentant de minimiser le temps de calcul nécessaire. Fixer ce temps revient principalement à fixer le nombre de solutions de recherche à évaluer, et donc, à fixer le nombre de fourmis participantes à la recherche.

Afin de fixer le nombre exact de fourmis nécessaires pour l'application du Ant-IS ainsi que leurs emplacements nous nous somme inspirés des qualités des algorithmes de clustering. Nous avons proposé de faire appel à une phase rapide de clustering, telle une étape de prétraitement, afin de désigner les centres et le nombre de clusters dans l'ensemble à réduire. Pour un ensemble primaire i le nombre de clusters sera le nombre de fourmis impliquées dans la sélection, et les centres de ces clusters leurs positions.

Le clustering permet de trouver des clusters d'objets de données qui sont similaires les uns aux autres. Le but de l'analyse de clustering est de trouver des clusters de haute qualité tels que la similarité inter-cluster soit faible et la similarité intra-cluster soit élevée.

Le clustering est utilisé pour segmenter les données. Contrairement à la classification, le clustering permet le regroupement des modèles de données dans des groupes

TABLE 4.6 – Comparaison de certains algorithmes de clustering

Algorithme	1252 pts	2503 pts	3910 pts	6256 pts	7820 pts	10426 pts	12512 pts
Cobweb	5.64	11.27	17.61	28.19	35.22	47.1	56.4
CLARENS	758	3026	6845	18029	29826	60540	80638
DBSCAN	3.1	6.7	11.3	17.8	24.5	32.7	41.7
EM	882	1764	2755	4409	5511	7348	8819
Farthest First	3.2	6.7	11.5	18.01	23.7	32.7	42.3
kMeans	3.51	7.01	11.01	17.54	21.92	29.23	35.07

qui n'ont pas été préalablement définis.

Le clustering est utile pour l'exploration des données. S'il y a beaucoup d'instances et pas de groupements évidents, les algorithmes de clustering peuvent être utilisés pour trouver des regroupements naturels. Le Clustering peut aussi servir comme une étape de prétraitement des données utiles pour identifier des groupes homogènes sur lesquels des modèles supervisés peuvent être construits.

4.4.1 Critères de choix

Puisque le clustering n'intervient qu'en phase de prétraitement dans notre approche, nous avons basé notre choix pour l'algorithme de clustering que nous allons utiliser principalement sur le critère de vitesse de traitement, où l'algorithme doit être le plus rapide que possible. Toutefois, l'algorithme doit aussi être efficace sur de grandes bases de données et ne doit pas nécessiter de connaitre le nombre de clusters apriori.

Afin de bien choisir l'algorithme qui nous permettra d'initialiser le nombre et les emplacements initiaux des fourmis, nous avons dressé une comparaison à travers le tableau 4.6, basée sur la vitesse de traitement de certains algorithmes de clustering qui fixent le nombre de clusters pendant le traitement, et qui sont connus pour leur efficacité sur de grandes bases de données.

Le tableau 4.6 donne le temps nécessaire pour chaque algorithme afin d'effectuer le clustering d'un ensemble de points que nous avons incrémenté.

Les résultats décrits dans ce tableau montrent clairement que le choix le plus judicieux suivant nos critères, précédemment cités, serait l'algorithme DBSCAN.

4.4.2 L'algorithme DBSCAN

Dans cette section, nous présentons l'algorithme Density Based Spatial Clustering of Applications with Noise ou DBSCAN qui est conçu pour découvrir les clusters et le bruit dans un espace de base de données.

Cet algorithme est l'un des algorithmes de clustering les plus rapides, il ne necessite pas de connaitre le nombre de clusters finaux apriori, il prend en considération la notion de bruit et est trés efficace sur les ensembles de données de grande taille.

Pour construire un cluster, DBSCAN commence par un point arbitraire p et récupère tous les points de densité accessible depuis p ainsi que WRT. Eps et MinPts. Si p est un point de base, cette procédure donne une WRT, Eps et MinPts du cluster. Si p est un point frontalier, aucun point n'est à une densité accessible à partir de p et DBSCAN visite le point suivant de la base de données.

Puisque nous utilisons des valeurs globales pour Eps et MinPts, DBSCAN peut fusionner deux groupes de données dans un même cluster, si deux groupes de densité différente sont "proches" l'un de l'autre. Soit la distance entre deux ensembles de points S1 et S2 définie par : dist (S1, S2) = min dist (p, q) — p ∈ S1, q ∈ S2.

Ensuite, les deux ensembles de points ayant une densité égale au moins à celle du plus petit cluster, seront séparés, l'un de l'autre seulement si la distance entre les deux ensembles est supérieure à Eps. Par conséquent, un appel récursif de DBSCAN peut être nécessaire pour les clusters détectés ayant une valeur plus élevée pour MinPts.

Dans l'algorithme 4.4.2 nous présentons une version de base de DBSCAN.

6 Algorithme DBSCAN

```
DBSCAN (Init, Eps, MinPts)
// Init : ensemble initial d'instances
ClusterId := nextId(NOISE) ;
Pour i= 1 : |Init| faire
    Point := Init(i) ;
    Si Point.ClId = NONCLASSÉ alors
        Si ExpandCluster(Init, Point, ClusterId, Eps, MinPts) alors
            ClusterId := nextId(ClusterId)
        Fin si
    Fin si
Fin pour
Fin.
```

4.4.3 L'algorithme Clust Ant-IS

Nous avons choisi l'algorithme DBSCAN pour sa complexité temporelle réduite.

Nous avons proposé d'améliorer l'algorithme Ant-IS à travers l'introduction d'une phase préliminaire, permettant de désigner le nombre et l'emplacement des foumis participantes à la recherche.

Cette phase intègre l'algorithme de clustering. Elle fait appel à ce dernier afin de dresser une carte des clusters de l'ensemble en précisant leur nombre et leurs centres. Ces informations serviront au paramétrage de l'algorithme Ant-IS afin de l'optimiser. Le nombre de fourmis sera donc réduit, ce qui réduira le nombre de solutions à évaluer. En même temps leurs emplacements ne se font pas de manière aléatoire, chaque fourmi sera affectée à un centre de cluster d'où elle démarrera la construction de sa solution. En construisant sa solution une fourmi parcoure tout l'ensemble d'instances.

7 Prétraitement dans Clust Ant-IS
(n,C)= DBscan(T) ;
Pour i=1 à n faire
    Positionner (P_i,C) ;
Fin Pour ;

Le pseudo-code décrivant l'étape introduite peut être noté par l'algorithme 4.4.3.

Où n est le nombre de clusters et C la liste des centres de clusters.

Après cette étape l'algorithme Ant-IS muni des nouveaux emplacements des fourmis est itéré jusqu'à obtenir une solution.

4.4.4 Paramétrage de l'algorithme DBSCAN

Afin d'évaluer les performances de la technique implémentée, il était dur de déterminer des valeurs précises pour Eps et Minpts pour l'algorithme DBSCAN utilisé pour le prétraitement. Cependant, nous sommes arrivés après plusieurs tests à fixer ε à 1, et la valeur de MinPts à 4.

4.4.5 Analyse et discussion

En examinant la qualité des résultats obtenus par l'algorithme Clust Ant-IS (voir chapitre 6), nous pouvons noter que l'algorithme offre un bon taux de classification, un bon taux de détection et un excellent taux de réduction d'alertes fausses positives, ce qui signifie qu'une partie de nos objectifs a été atteinte. Cependant, l'algorithme en question offre un bon taux de réduction. Toutefois, il reste encore insatisfaisant pour une utilisation en temps réel du classificateur 1NN. Nous avons pensé qu'un nombre de ces instances, choisies, pouvait, peut être, former un surplus dans l'ensemble, dans le sens où ces instances étaient neutres pour la classification et n'amélioraient pas ses résultats.

Nous avons, donc, proposé d'agir différemment pour construire l'ensemble d'apprentissage, en n'incluant que les instances qui ont une participation importante dans la classification.

Afin d'améliorer le taux de réduction, sans sacrifier pour autant les résultats de classification atteints, nous avons pensé à intégrer l'apprentissage actif. Cela nous permettrait de construire l'ensemble d'apprentissage minimal en n'intégrant que les instances nécessaires pour atteindre ces taux de classification.

4.5 Sélection d'instances avec Ant-IS et l'apprentissage actif

Nous avons, dans ce travail, tenté d'améliorer notre proposition précédente, grâce à l'introduction de l'apprentissage actif. Dans notre nouvelle approche, Active Learning

Ant-IS, nous avons continué le processus principal de Clust Ant-IS en l'orchestrant par les mécanismes d'apprentissage actif. L'algorithme Clust Ant-IS offre un sous ensemble minimal pour l'entrainement de l'algorithme KNN. Pour la création de cet ensemble minimal l'algorithme Clust Ant-IS exige un balayage entier de l'ensemble initial. Ce qui nous a menés à penser que cet ensemble peut encore être réduit.

La version active de l'algorithme Clust Ant-IS permet d'améliorer les performances de réduction de cet algorithme.

Rappelons que l'apprentissage actif, est utilisé afin de sélectionner les instances, une à une pour construire l'ensemble d'apprentissage.

Par conséquent, jumelé au Clust Ant-IS, l'apprentissage actif peut permettre de minimiser encore d'avantage l'ensemble d'apprentissage, tout en conservant les qualités de ce dernier.

Pour créer un ensemble de formation efficace pour la classification des alertes, nous avons opté pour un ensemble initial basé sur KDD'99. Nous avons divisé la formation KDD'99 en trois sous-ensembles : l'ensemble initial de formation (L), l'ensemble non étiqueté (U) et l'ensemble de test de formation (TT).

L'ensemble d'apprentissage initial est un ensemble minimal initial qui sert de base pour construire la base d'apprentissage. L'ensemble non étiqueté contient toutes les instances qui pourraient être incluses dans l'ensemble de la formation. Enfin l'ensemble d'apprentissage de test est un ensemble d'instances marquées utilisées pour évaluer l'ensemble d'apprentissage pendant sa construction.

L'algorithme ALAIS commence d'abord par créer L, qui va contenir les instances issues du clustering. Ces dernières serviront de points de départ pour les fourmis afin de construire l'ensemble réduit final. A partir de L, les fourmis vont ajouter les instances annotées une par une, et ce jusqu'à la condition d'arrêt. La sélection de ces instances est lancée par l'apprentissage actif, mais le choix des éléments à inclure se fait en utilisant Ant-IS. Ant-IS est lancé lors du premier appel de la procédure d'apprentissage actif. Il effectue un pas à chaque appel jusqu'à ce que la condition d'arrêt soit atteinte ou que l'ensemble d'entrainement initial soit entièrement couvert.

L'approche proposée peut être décrite par la figure 4.1.

Pour expliquer le raisonnement suivi dans le développement de cette approche, nous avons besoin de redéfinir certaines conventions et d'en rajouter de nouvelles :

T : ensemble d'instances ;

n =$\mid T \mid$: nombre d'instances ;

b_i (t) : nombre de fourmis dans l'instance i à l'instant t ;

n=$\sum_{x \in i} b_i$:nombre total de fourmis ;

$n_{ij} = \frac{1}{d_{ij}}$: visibilité d'une instance j à une fourmi quand elle est sur l'élément i ;

Pij (t) : valeur de la phéromone sur l'arc (i, j) ;

C : une matrice (bi (t) xn) contenant la validation des villes de destination ;

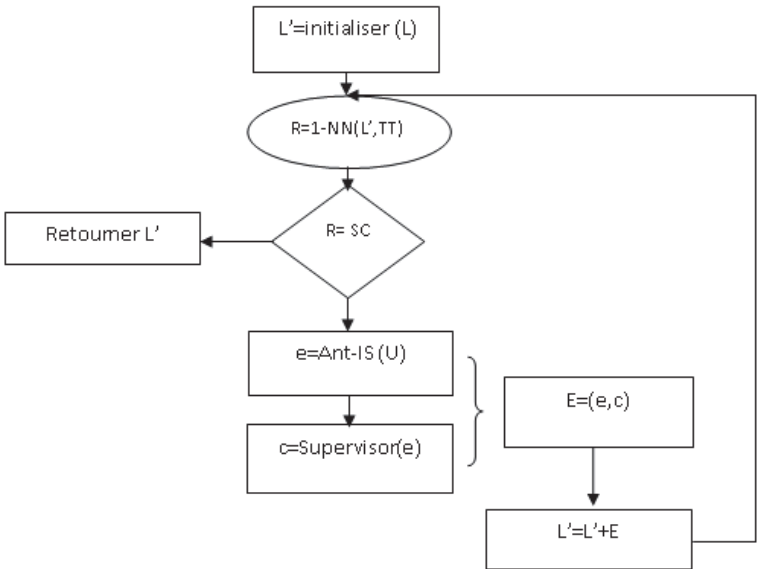

FIGURE 4.1 – Schéma général de l'algorithme ALAIS

a_j^k : paramètre aléatoire ∈ 0,1.

L : ensemble d'apprentissage initial

L ' : ensemble d'apprentissage construit

U : ensemble non marqué

TT : ensemble de test d'apprentissage

Étant basé sur des principes d'apprentissage actif, la participation du Ant-IS passe par deux étapes : l'initialisation et la boucle, et ceci comme suit :

4.5.1 Initialisation

L'initialisation de ALAIS comprend l'initialisation des deux algorithmes qui le constituent. Elle inclue donc l'initialisation du Clust Ant-IS et celle de l'apprentissage actif, et ce comme suit :

Initialisation par l'algorithme Clust Ant-IS

A l'initialisation, l'algorithme de clustering est lancé afin de récupérer le nombre et les centres de clusters. Le nombre de fourmis est ensuite fixé au nombre de clusters obtenus et ces fourmis sont ensuite placées sur les différents centres, à raison d'une fourmi sur chaque centre.

Aussi, tous les chemins entre deux instances sont notés avec la même valeur initiale.

Enfin, la matrice de sélection (C) est initialisée.

Initialisation de l'apprentissage actif

Choix de l'ensemble initial : Les instances constituant l'ensemble de la formation initiale (L) sont les centres des clusters récupérés à partir de l'algorithme de clustering.

Entrainement : L'ensemble L initial est utilisé comme ensemble de formation pour former le classificateur 1NN. Les performances de classification sont évaluées en utilisant l'ensemble de test d'entrainement. Le taux de classification, dans ce cas, représente l'évaluation de l'ensemble L.

L'étape d'initialisation peut être présentée comme suit :

1. Initialiser L ;
2. L'=L ;
3. Initialiser C ;
4. (n,C)= Clustering(T) ; //Initialiser le nombre de fourmis ;
5. Pour k=1 à n faire Positionner (P_{ij},C) ;
6. Fin Pour ;
7. 1NN (L, TT) ;

4.5.2 Close-loop

La boucle close-loop concerne la phase principale de l'exécution de l'algorithme d'apprentissage. Cette phase inclue les étapes permettant la création de l'ensemble d'apprentissage.

Fonction requête

La fonction de requête, tel que mentionné précédemment (chapitre 3) est utilisée pour sélectionner des instances afin de construire l'ensemble de la formation. Elle permet ainsi de créer cet ensemble, mais surtout de déterminer le nombre d'instances incluses dans ce dernier. Si les instances sont mal choisies, l'évaluation de l'ensemble encourage plus d'ajouts, et donc la taille de l'ensemble L peut inutilement et exessivement augmenter. Pour choisir les instances à inclure dans L, nous avons choisi comme fonction de requête l'algorithme Ant-IS [137] précédemment décrit, où à chaque invocation de la fonction requête, toutes les fourmis font un pas et une sélection du meilleur choix est faite. Les étapes d'exécution de l'algorithme Ant-IS sont maintenues, sauf l'étape de validation de choix, qui va cette fois-ci valider une seule instance comme choix au lieu d'un ensemble entier.

Validation de choix

Après que les fourmis aient fait leurs choix, le meilleur d'entre eux est calculé.

Chaque instance choisie servira seule à l'entrainement du classificateur 1-NN pour reclasser les instances de l'ensemble de test d'entrainement (TT). Le taux de classement acck est calculé selon la formule 6, puis chaque instance choisie est associée à son taux de classification. Le meilleur choix est celui qui offre le taux de classification le plus élevé.

La précision de la classification d'une instance k est calculée comme suit :

$$acc_k = \frac{N_k}{|S_t|} \tag{4.10}$$

Où N_k est le nombre d'instances correctement classées par le classificateur 1-NN utilisant l'instance k pour son entrainement, et St est le nombre total d'instances de test.

L'algorithme de fonction requête peut être présenté comme suit :

---

n= nombre de fourmis utilisées

1. Pour k=1 à n faire
   - i=k ; // chaque fourmi commence à partir d'une instance différente
2. Répéter
   (a) Calculer n_{ij} ;
   (b) Calculer $Prob_{ij}^k$ en utilisant la formule (4.4) ;
   (c) Choisir l'instance j avec la meilleure probabilité ;
   (d) choisir aléatoirement une valeur pour a_j^k ;
   (e) Calculer $FProb_{ij}^k$ suivant la formule (4.5) ;
   (f) Si ($FProb_{ij}^k$! =0) alors i=j ;//se déplacer sur j ;
   (g) Sinon Répéter
      i. j= l'instance avec la prochaine $Prob_k$;
      ii. Choisir aléatoirement a_j^k ;
   (h) Jusqu'à ce qu'un déplacement ne soit possible ;
   (i) Mettre à jour P_{ij} à travers la formule (4.7) ;
3. Fin pour ;
4. J=j_i avec le meilleur (acc_k) ;
5. Retourner J ;

Création de L'

l'ensemble d'apprentissage est augmenté en ajoutant l'exemple choisi par la fonction de requête. L'exemple choisi et sa classe sont inclus dans l'ensemble d'apprentissage pour construire le nouvel ensemble L.

Réentrainement de 1NN

le classificateur 1NN est réentrainé en utilisant le nouvel ensemble de formation et l'ensemble TT. Le résultat de classification représente l'évaluation de l'ensemble L.

4.5.3 Condition d'arrêt

L'algorithme ALAIS est réitéré jusqu'à ce qu'un taux de classification minimum ne soit atteint. Le résultat de réentrainement du 1NN est comparé à un seuil minimum de classification initialement fixé.

4.6 Conclusion

Dans ce chapitre, nous avons présenté notre première proposition dont le principe est fondé sur l'optimisation par colonies de fourmis. Nous avons fait appel à ce concept en proposant d'utiliser les fourmis dans la sélection des instances destinées à la construction d'un ensemble d'entrainement efficace permettant de réduire le nombre d'alertes fausses positives lors de la classification d'attaques.

Afin d'améliorer les performances de ces fourmis nous avons mêlé la notion de clustering à l'initialisation de notre approche.

Enfin, l'algorithme proposé est basé sur le concept d'apprentissage actif, qui permet d'orchestrer les déplacements des fourmis et d'optimiser la sélection des instances.

Chapitre 5

Une approche mémétique pour la détection d'intrusions

5.1 Introduction

Ce chapitre décrit un algorithme mémétique de sélection d'instances permettant l'amélioration de la classification dans un IDS. L'algorithme en question comprend une nouvelle recherche locale qui permet l'orientation et l'amélioration d'une solution génétique. Par ailleurs, l'algorithme mémétique proposé renforce les capacités du classificateur 1NN classique dans le domaine de la détection d'intrusions. Cet algorithme vise à réduire l'ensemble d'apprentissage utilisé par l'algorithme 1NN.

L'algorithme MCLS tente d'abord de trouver une solution primaire acceptable en utilisant les principes de la génétique, et puis d'améliorer cette solution, en appliquant la recherche locale proposée.

5.2 GIS : Un algorithme génétique pour la sélection d'instances

Notre objectif est de construire un ensemble d'apprentissage minimal et efficace pour le classificateur 1NN, permettant à ce dernier d'arriver à une très bonne classification des paquets entrants, et par conséquent de réduire le taux d'alarmes fausses positives.

Dans l'optique d'atteindre ce but , nous avons d'abord conçu un algorithme génétique binaire, que nous avons nommé Genetic Instance Selection algorithm (GIS). Cet algorithme vise à trouver la solution la plus proche de l'optimal, permettant d'achever une bonne classification par le 1NN standard, tout en réduisant son ensemble d'apprentissage.

Cet algorithme a pour mission de sélectionner les instances d'entrainement les plus pertinentes afin d'atteindre des taux de classification satisfaisants. L'algorithme a été conçu en se basant sur une fonction objectif offrant un compromis entre la classification et la réduction.

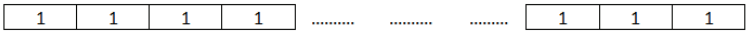

FIGURE 5.1 – Schéma du chromosome représentant l'ensemble initial

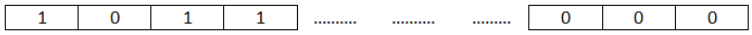

FIGURE 5.2 – Schéma du chromosome représentant l'ensemble réduit

Dans cette section nous allons discuter nos choix pour les caractéristiques de cet algorithme.

5.2.1 Chromosome

Etant un élément de base pour la solution génétique, les choix des caractéristiques du chromosome sont primordiaux.

Dans la solution que nous proposons, le chromosome représente l'ensemble d'entrainement résultant (réduit). Chaque chromosome de la population représente une proposition de réduction de l'ensemble d'entrainement initial. Puisque nous n'avons aucune information à propos de l'importance d'une instance, que cette instance apporte une amélioration ou non, qu'elle peut générer des faux positifs ou pas, la seule solution est de générer des combinaisons aléatoires d'instances, cette combinaison est représentée sous forme d'un chromosome, sa taille est identique à celle de l'ensemble à réduire. Nous avons choisi pour représenter ce chromosome un codage binaire.

Chaque gène du chromosome représente une instance de l'ensemble initial. Si ce gène est égale à 1, cela veut dire que l'instance représentée par ce gène est présente dans l'ensemble réduit représenté par le chromosome. Cependant, si le gène est égal à 0, l'instance en question ne fait pas partie dudit chromosome.

L'ensemble initial de taille n pourrait être représenté par le chromosome illustré par la figure 5.1 , où toutes les instances sont représentées dans ce chromosome.

Un exemple d'un ensemble réduit pourrait être illustré par le chromosome montré par la figure 5.2, où seulement les instances 1,3 et 4 sont inclues dans l'ensemble réduit représenté par ce chromosome.

L'évaluation d'un chromosome revient à identifier les taux de classification et de réduction atteints par ce dernier. Les choix pour la fonction d'évaluation sont détaillés plus loin.

5.2.2 Opérateurs génétiques

Le choix des opérateurs génétiques revient à choisir la manière et les conditions d'évolution et de développement de la population génétique, et donc, la façon par laquelle de nouvelles solutions seront conçues.

Tout comme le développement naturel, les opérateurs génétiques concernent la sélection des individus, leur croisement et la mutation de la population.

Afin d'élaborer un environnement de développement aussi optimal que possible, nous avons testé diverses combinaisons de ces opérateurs Cette combinaison a permi la création de notre algoritme GIS.

Les différentes opérations du cycle génétique sont décrites ci-aprés.

Sélection

Pour générer des enfants, les individus actuels doivent être évalués et sélectionnés. La sélection naturelle garantit que les individus forts survivent, tandis que les plus faibles meurent. Pour l'approche proposée, et après plusieurs tests (voir tableau 5.1), nous avons mis en place une sélection par tournoi de deux. Un tournoi est une rencontre entre un certain nombre d'individus pris au hasard dans la population. Le vainqueur du tournoi est le meilleur individu.

TABLE 5.1 – Détail des cinq ensembles de données utilisés dans l'évaluation

Méthode	Classification (%)	Réduction (%)
Sélection par roulette	91.0	85.0
Sélection élitiste	92.0	82.0
Tournois (2) avec remise	93.0	80.0
Tournois (2) sans remise	**96.0**	**83.0**
Tournois (4) avec remise	94.0	77.0
Tournois (4) sans remise	94.0	84.0
Tournois (8) avec remise	95.0	83.0
Tournois (8) sans remise	93.0	81.0
Tournois (16) avec remise	93.0	84.0
Tournois (16) sans remise	94.0	76.0

La sélection par tournoi n'utilise que les comparaisons entre individus, sans nécessiter de tri de la population. Elle possède un paramètre T, taille du tournoi, qui permet de tirer T individus de la population afin de sélectionner le meilleur d'entre eux.

Croisement

Le croisement est conçu pour améliorer la diversification de la population en manipulant la structure des chromosomes. Le croisement génère deux descendants à partir de deux parents. Nous avons décidé d'utiliser le croisement à un seul point pour ce travail. Un point de croisement est sélectionné de façon aléatoire avec une probabilité précédemment établie. Pour créer un nouveau chromosome, la chaîne binaire du début du chromosome parent est copiée jusqu'au point de croisement, et le reste est copié à partir de l'autre parent. Le modèle de croisement suivi est présenté par la figure 5.3.

FIGURE 5.3 – Croisement utilisé

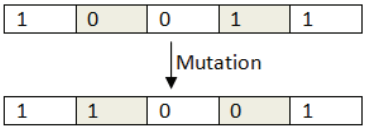

FIGURE 5.4 – Mutation utilisée

Mutation

La mutation permet de completer la sélection en générant des solutions originales à partir d'une solution donnée. Elle consiste en une transformation du chromosome gérée par une probabilité. La mutation d'un chromosome représentant une solution donnée, permet de générer de cette dernière une seconde solution en changeant certaines valeurs de gènes.

En appliquant la probabilité de mutation, les gènes sélectionnés pour être mutés, seront inversés. Un gène de valeur 1 changera sa valeur pour 0, et viceversa. La mutation, donc, permet de désélectionner des instances déjà sélectionnées comme faisant partie de la solution proposée, et d'en sélectionner de nouvelles, ne faisant pas partie de la solution actuelle, en mettant les valeurs des gènes correspondants à ces dernières à 1.

La figure5.4 décrit le principe de cette mutation.

5.2.3 Evaluation d'un individu

Chaque chromosome de la population initiale est évalué à l'aide d'une fonction objectif fitness1. Cette fonction est basée sur deux objectifs :

- Réduire au maximum le nombre de gènes actifs dans le chromosome
- Maximiser la précision de classification 1NN utilisant les instances sélectionnées comme ensemble d'apprentissage.

Afin de concevoir une fonction objectif qui couvre ces buts nous avons proposé d'utiliser deux variables : la précision de classification (Class) et le taux de réduction (Red), pondérés par deux paramètres $\lambda 1$ et $\lambda 2$, tel que $\lambda 1+\lambda 2=1$.

Précision de classification

Le taux de classification attribué à un chromosome concerne les performances atteintes par le classificateur 1NN lorsqu'il admet l'ensemble réduit représenté par ce

chromosome comme ensemble d'entrainement.

La précision de classification dépend du nombre d'échantillons correctement classés. Elle est calculée comme étant le rapport du nombre d'instances correctement classées par 1NN comparé au nombre total d'instances.

Le classificateur 1NN utilise l'ensemble représenté par le chromosome comme ensemble d'apprentissage, et reclasse les instances appartenant à l'ensemble d'apprentissage initial. Le taux de classification résultant correspond à la précision offerte par le chromosome évalué.

La précision de classification est calculée comme suit :

$$Class = \frac{\#CC}{Taille} \tag{5.1}$$

Où CC est le nombre d'instances correctement classées en utilisant le chromosome i , et Taille est la taille de l'ensemble d'entrainement initial.

Taux de réduction

Le taux de réduction permet d'évaluer les performances du chromosome concernant la diminution du nombre d'instances nécessaires pour une classification efficace. Le nombre de gènes actifs dans un chromosome représente le nombre d'instances inclues dans l'ensemble réduit représenté par le chromosome. Le taux de réduction est le résultat de la division du nombre d'instances retenues sur la taille initiale de l'ensemble d'apprentissage. En d'autres termes, c'est le nombre de gènes actifs comparé à la taille du chromosome.

$$Red = \frac{\#actifs}{||Chromosome||} \tag{5.2}$$

Où #actifs est le nombre de gènes actifs dans le chromosome i.

Après avoir défini les paramètres la composant, la fonction objectif peut être formulée comme suit :

$$Fitness1 = \lambda 1 * Class + \lambda 2 * Red \tag{5.3}$$

Nous avons effectué plusieurs tests sur les combinaisons ($\lambda 1$, $\lambda 2$) (voir figure 5.5) afin de préciser les valeurs des deux paramètres.

Durant ces tests nous avons fait varier $\lambda 1$ et $\lambda 2$ et évalué la population grâce à fitness1, ainsi que les taux de réduction et de classification.

D'après les résultats, les combinaisons (0.6 ; 0.4) et (0.9 ; 0.1) offrent les meilleurs résultats pour *fitness1*. Cependant, dans le domaine de la détection d'intrusions, la précision de classification reste un critère trés sensible et trés important à satisfaire.

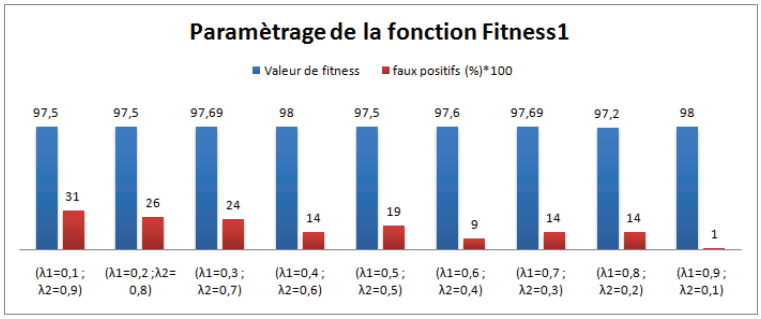

FIGURE 5.5 – Paramétrage de la fonction Fitness1

Pour cela nous avons choisi pour la fonction *fitness1* les valeurs 0.9 et 0.1 pour $\lambda 1$ et $\lambda 2$ respectivement.

Ces deux valeurs permettent d'atteindre les objectifs fixés pour cette fonction, i.e. cette combinaison offre un trés haut taux de classification parmi les combinaisons testées tout en permettant un excellent taux de réduction.

5.2.4 Taille de la population

La population est composée de l'ensemble des chromosomes participant à la conception de la meilleure solution. Chaque chromosome de cette population représente une éventuelle solution à évaluer.

La taille de cette population est donc intrinsèquement liée au nombre de solutions initiales.

Pour cela, nous avons pris l'initiative de régler la taille de la population initiale à travers des tests (voir tableau 5.2).

TABLE 5.2 – Paramétrage de la taille de la population initiale

Taille de population	Classification (%)	Réduction (%)
25%	96	91
50%	96	94
75%	97	91
100%	98	92

Nous avons testé les performances de l'algorithme pour une taille égale à trois fractions (1/4, 1/2, 3/4) différentes de l'ensemble d'apprentissage initial et enfin à une taille égale à celle dudit ensemble.

Pour chaque taille évaluée, nous avons évalué la valeur *fitness1* de la solution résultante en nous basant sur cette taille de population pour l'exécution de l'algorithme génétique, ainsi que le temps nécessaire pour cet algorithme afin d'atteindre cette solution.

Les résultats des tests montrent que la population de taille égale à celle de l'ensemble à réduire.permet d'obtenir une valeur de fitness satisfaisante en un temps acceptable.

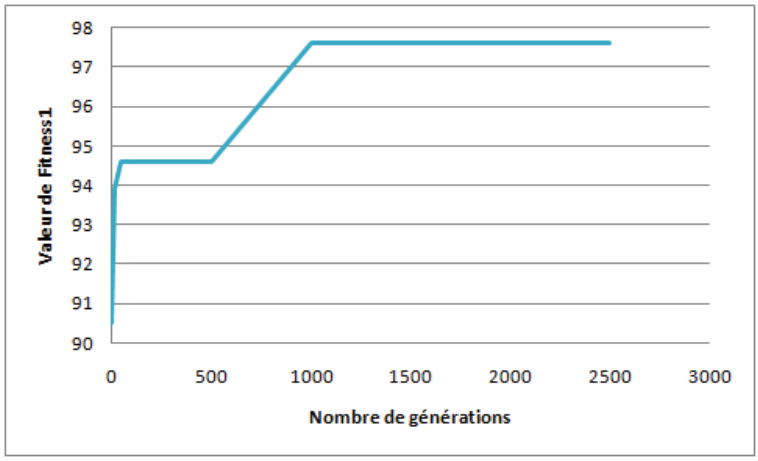

FIGURE 5.6 – Evolution des valeurs de Fitness1 suivant le nombre de générations

5.2.5 Nombre de générations

Le nombre de générations influe sur l'amélioration des solutions. Ce nombre définit le nombre de fois où la population est mise à jour, par la création de nouvelles propositions de solutions, à travers l'application des opérateurs génétiques.

Nous avons fixé le nombre de générations en évaluant la valeur de fitness1 pour la solution obtenue, lors de la variation du nombre de générations, les résultats de ces tests, résumés par la figure 5.6, montrent qu'à 1000 générations la solution obtenue atteint la meilleure valeur pour *fitness1*.

5.2.6 Discussion

En résumé, l'algorithme génétique que nous avons proposé comprend une population de taille égale à la taille de l'ensemble d'entrainement initial, composée de chromosomes binaires de taille égale à celle de l'ensemble à réduire, où chaque gène représente l'état d'une instance. Cette population, afin d'évoluer, est soumise à une sélection par tournoi de deux, à un croisement à un point et à une mutation d'un taux égal à 0.1 . L'évolution de cette population, ainsi que le choix de la meilleure solution sont orchestrés par la fonction *fitness1*, qui est basée sur les taux de classification et de réduction afin d'évaluer un chromosome.

En examinant les résultats obtenus par l'algorithme génétique proposé (voir chapitre Tests) nous avons pu constater la qualité de la solution finale, en termes de classification et de réduction d'alertes fausses positives. La solution présente aussi une réduction satisfaisante de l'ensemble initial. La qualité de ces résultats et les caractéristiques des algorithmes génétiques nous ont menés à penser que cette solution pouvait être améliorée.

L'amélioration dans ce cas devrait toucher les trois niveaux, elle concerne donc, la

réduction de l'ensemble résultant ainsi que le nombre d'alertes fausses positives, mais aussi, l'amélioration de la précision de classification des paquets entrants.

Ce qui est clair, c'est que l'efficacité de la solution obtenue à partir de l'approche génétique, est due au choix d'instances. Pour cela, l'amélioration de la solution devrait se baser sur ce choix. Ceci nous a conduit à émettre l'hypothèse qu'une amélioration possible de la solution en question serait de se baser sur les choix faits jusqu'ici, et donc sur l'ensemble résultant, pour tenter de le réduire en veillant à améliorer la réduction de faux positifs ainsi que le taux de précision ou au moins essayer de les maintenir.

Cette hypothèse a été à la base du développement d'une recherche locale contrôlée, visant à faire évoluer une solution primaire en respectant les choix d'instances faits pour cette solution, et en tentant de les réduire. En d'autres termes, cette recherche ne change pas les instances sélectionnées, elle réduit cet ensemble sans introduire de nouvelles instances.

Cette recherche locale nous a permis de transformer l'approche génétique proposée en un algorithme mémétique efficace, que nous avons appelé Memetic Controlled Local Search (MCLS). Les détails concernant la recherche proposée ainsi que la proposition mémétique sont décrits dans les sections suivantes.

5.3 Amélioration locale

Pour améliorer la précision sans pour autant diminuer le taux de réduction déjà réalisé, nous avons proposé de limiter le nombre de gènes mutants lors de la recherche locale.

L'évaluation des fonctions supplémentaires peut s'avérer très coûteuse. Pour cette raison, il est nécessaire de ne vérifier le bon fonctionnement de la recherche locale que sur les solutions visitées.

Pour le faire, nous avons proposé un algorithme de recherche locale contrôlée (CLS). Cette recherche est dite contrôlée en raison de sa portée limitée et sa sélection des gènes à muter. CLS n'affecte pas toutes les composantes du chromosome sur lequel il est appliqué. Il n'est appliqué que sur les gènes actifs du chromosome.

La recherche locale permet la création de solutions voisines d'une solution de départ par mutation de gènes actifs. Ces solutions voisines sont évaluées pour définir la meilleure solution.

A chaque étape de la recherche locale, cette méthode se déplace vers une meilleure solution voisine.

La recherche CLS s'arrête lorsque tous les candidats voisins sont considérés comme pires que la solution actuelle, c'est à dire quand un optimum local a été atteint. Alternativement, elle peut être définie comme suit : à partir d'une solution s, choisir une solution s', dans le voisinage de s, de façon que s' améliore la recherche.

CLS repose sur deux paramètres principaux : la définition du voisinage ou les solutions de voisinage et la fonction d'évaluation.

Définition du voisinage : Le voisinage comprend des solutions dérivées de la solution principale, qui devraient améliorer les résultats de l'évaluation. Le résultat du croisement génétique est la solution initiale. Cette solution fournit déjà un bon compromis entre la précision et le taux de classification. La recherche locale dans ce cas, devrait améliorer les taux de réduction de faux positifs et de classification sans atténuer le taux de réduction. En plus, cette recherche locale tente même d'améliorer ce dernier. Pour atteindre ce dernier objectif, nous avons proposé une mutation des gènes qui sont déjà actifs. En d'autres termes, cette recherche considère l'ensemble réduit représenté par le chromosome comme ensemble initial et tente de trouver des sous-ensembles permettant d'obtenir un meilleur taux de faux positifs et une meilleure classification à travers l'application de l'opérateur de mutation. Pour ce faire, nous avons proposé une mutation binaire contrôlée (Mutc) en fonction du voisinage. Cette mutation, en plus du taux de mutation, comprend une sélection des gènes à muter.

Mutc, par conséquent, considère l'ensemble de gènes actifs (gènes marqués à 1) lors de l'application de l'opérateur de mutation. En utilisant cet opérateur, les gènes identifiés sont inversés de 1 à 0. Mutc permet de certifier que le taux de réduction ne sera pas détérioré. Le taux de réduction est conservé si la solution initiale est la meilleure et amélioré s'il existe une solution voisine qui est meilleure.

Pour éviter d'être bloqués dans un optimum local, nous avons choisi, un taux de mutation de 0.05 pour Mutc.

Le pseudo-code de la mutation proposée (Mutc) est décrit par l'algorithme 8.

5.3.1 La fonction d'évaluation

Notre premier objectif est d'appliquer l'algorithme mémétique dans le domaine de la détection d'intrusions. Ce domaine est basé sur deux critères principaux : le taux de détection et le taux de faux positifs.

Le taux de détection dépend directement de la performance de la classification des différents types d'attaques, tandis que le taux de faux positifs dépend de la reconnaissance des instances normales, à savoir la classification des paquets non-intrusifs.

Afin de couvrir ces deux objectifs, nous avons proposé d'utiliser une seconde fonction objectif (fitness2) avec des variables paramétrées, $\lambda 3$ et $\lambda 4$, où $\lambda 3+\lambda 4=1$. La fonction objectif proposée est basée sur la somme des deux taux pondérés comme suit :

$$Fitness2 = \lambda 3 * (Class) + \lambda 4 * (FP) \tag{5.4}$$

Le taux de classification (class) est calculé selon la formule 5.1 présentée dans la section 5.2.3.

8 Mutation binaire controllée proposée

Mutrate : taux de mutation ;
L : taille du chromosome ;
J : nombre de genes actifs ;

1. Pour i =1 :L
   (a) si (chromosome(i)==1)
      i. K(a)= i ; a++ ;
      ii. J++ ;
   (b) finsi
2. fin pour
3. For i=1 :j
   - cromTemp=1 ; // création d'un nouveau chromosome ;
4. fin pour
5. Calculer le nombre total de mutations ;
6. Designer les colonnes à muter ;
7. Mut (cromTemp) ;
8. Pour i=1 :a
   - Chromosome(k(i))=cromTemp(i) ; // incorporer chromTemp dans le chromosome
9. Fin pour ;

5.3.2 Taux de faux positifs (FP)

Le nombre de faux positifs représente le nombre de paquets non intrusifs classés comme intrusifs. Il est égal au nombre d'instances normales classées comme attaques divisé par le nombre total d'instances normales.

Le classificateur 1NN est utilisé pour effectuer la classification. 1NN utilise l'ensemble réduit représenté par le chromosome qui doit être évalué comme jeu de formation. 1NN tente également de reclasser les instances de la classe normale initiale. Le nombre de cas correctement classés est l'inverse du nombre de faux positifs.

L'algorithme de recherche locale proposée peut être décrit par l'algorithme 9 :

5.4 Un algorithme mémétique pour la détection d'intrusions

Comme cité précédemment, l'approche génétique proposée et l'amélioration locale discutée ultérieurement sont à la base de la création de l'algorithme mémétique proposé.

Le principe de l'approche mémétique proposée [136] peut être résumé comme suit :

A l'initialisation, une population Pop est créée aléatoirement. Cette population contient n chromosomes binaires de taille n, où n est le nombre d'instances dans l'en-

9 Recherche Locale Contrôlée proposée

Sg : progéniture

1. S := Sg ;
2. Final=Sg ;
3. Ev=fitness2(Sg) ;
4. Tant que S n'est pas un optimum local, répéter
   (a) S'= Mutc(S) ;
   (b) Evs=fitness2(S') ;
   (c) si (Evs $\geq$ Ev)
      – Final=S ;
   (d) finsi
5. Retourner Final ;

semble d'entrainement initial.

Rappelons que chaque chromosome représente un individu de la population, et chaque gène d'un chromosome représente une instance. Si un gène est égal à 1, cela signifie que l'instance représentée est inclue dans la solution représentée par le chromosome. L'instance ne fait pas partie de la solution si le gène correspondant est égal à 0. Ces individus sont d'abord évalués en utilisant la fonction objectif fitness1, où leur taux de réduction et de classification correspondants sont évalués.

Ensuite, une procédure de sélection est appliquée pour sélectionner deux individus en fonction de leurs valeurs de fitness. Les individus sélectionnés sont alors désignés comme les parents P1 et P2. Ces parents sont soumis à un croisement. Ce croisement produit deux enfants off1 et off2.

Nous avons proposé de remplacer la mutation génétique par cette recherche locale orientée afin de gérer le nombre d'instances sélectionnées lors du cycle génétique. Rappelons que notre objectif est de construire un ensemble d'entrainement réduit qui satisfait une bonne classification des attaques et une réduction significative du nombre d'alertes fausses positives.

A travers ce remplacement nous voulons travailler sur l'amélioration des progénitures issues du croisement au lieu de créer de nouvelles solutions.

Les deux progénitures sont ensuite améliorées grâce à l'application de CLS. La recherche locale proposée ne fonctionne que sur les gènes actifs de chromosomes (gènes marqués à 1). Cette recherche porte des mutations sur les gènes actifs, tout en évaluant les gènes résultant à travers une fonction objectif, nommée *fitness2* qui mesure le taux de classement global et le taux de faux positifs.

Après cette recherche, les enfants initiaux sont améliorés. Ces deux enfants remplaceront les deux pires individus de la population initiale, et la nouvelle population remplacera la première.

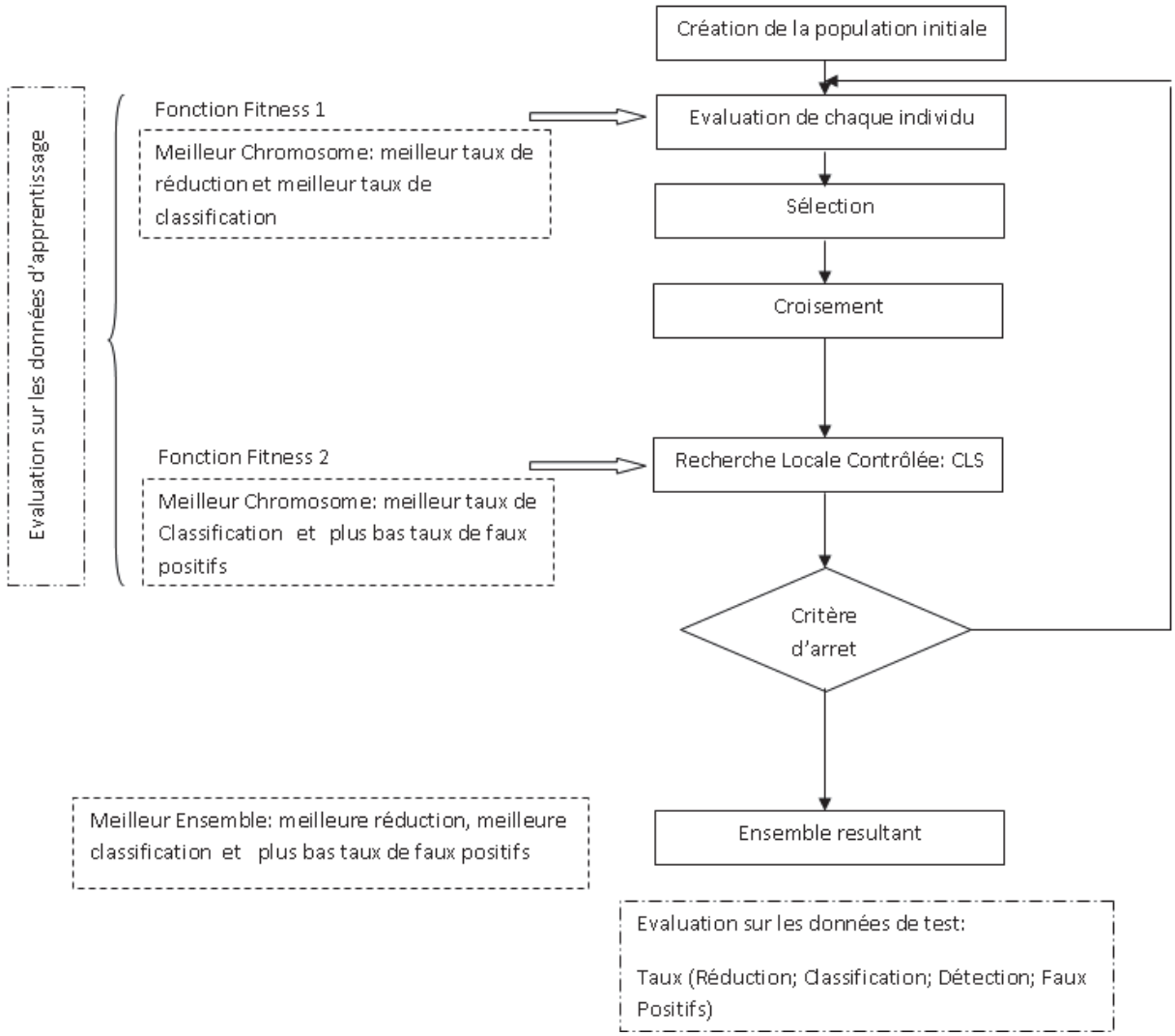

FIGURE 5.7 – Schéma de l'algorithme mémétique proposé : MCLS

Ce schéma d'évaluation est répété pour un nombre fixe de générations. Lorsque le critère d'arrêt est atteint, une évaluation finale est effectuée à l'aide de *fitness1* et le meilleur individu est sélectionné en tant que résultat final.

Le résultat dans ce cas répond aux trois critères cités qui sont :

1. Un bon taux de réduction : Mesuré par fitness1. Les individus, qui sont désignés comme meilleurs ont nécessairement un meilleur taux de réduction que les autres individus ;
2. Un taux de classification élevé : Les deux fonctions d'évaluation intègrent le taux de classification en tant que critère. Le taux de classification est évalué avant, pendant et après l'amélioration ;
3. Un faible taux de faux positifs : vérifié pendant l'amélioration directement à

travers *fitness2*, et indirectement lors de l'évaluation de la classification dans *fitness1*. Là, il serait utile de rappeler que le taux de faux positifs dépend de la classification des instances de la classe normale.

L'organigramme résumant le principe de l'intéraction entre les différents constituants de l'algorithme proposé est décrit par la figure 5.7.

5.5 Conclusion

Dans ce chapitre, nous avons proposé une méthode pour réduire le nombre de fausses alarmes découlant d'un IDS. Pour cela, nous avons créé un algorithme mémétique amélioré comprenant une recherche locale contrôlée. Le but de cette approche proposée consiste à améliorer l'utilisation du KNN dans la détection d'intrusions.

Avant de construire l'algorithme mémétique, nous avons proposé une approche génétique, qui elle aussi permet d'atteindre l'objectif visé. Une fois achevée, cette recherche offrirait une solution représentant un ensemble réduit pouvant servir comme ensemble d'entrainement au 1NN. Cet ensemble offre des performances remarquables en qualité de classification et de réduction de fausses alertes. Cependant, nous avons proposé de l'améliorer encore en introduisant une recherche contrôlée travaillant à améliorer principalement le taux de réduction dans cette solution.

L'approche mémétique résultante de la combinaison de ces approches est évaluée dans le domaine de la détection d'intrusions sur des données synthétiques et d'autres réelles dans le chapitre 6.

Chapitre 6

Tests

6.1 Données utilisées

Afin de valider notre travail, nous avons fait appel à la base de données KDD cup'99, qui est l'ensembles de données le plus utilisé dans le domaine de la détection d'intrusions.

La base de données DARPA 1999 ou KDD'99 [181] a été développée au MIT Lincon Labs. Elle contient une description des connexions TCP, qui sont composées chacune de 41 attributs. Une connexion est un ensemble de paquets envoyés d'une machine à l'autre en utilisant le même protocole. Ces connexions contiennent des données de test recueillies au cours de cinq semaines de simulation (5 jours par semaine) sur le réseau de l'Air Force qui contient différents types de machines (Linux, Solaris et Sun OS). Les données d'entrainement proposées comprennent les trois premières semaines de simulation et les données de test les deux dernières. Dans les trois premières semaines, des attaques ont été relevées à la deuxième semaine et aucune attaque en première et troisième semaine.

Les types d'attaques contenues dans cette base de données peuvent être classées en quatre catégories : Déni de service (DoS), Root to local (R2L), User to Root (U2R) et Probing. Ces attaques sont décrites comme suit :

- Déni de service : L'attaquant vise à rendre les services ou les ressources indisponibles pour une durée indéterminée.
- Root to Local (R2L) : L'attaquant ne dispose pas d'un compte sur la machine de la victime, et donc tente d'y accéder.
- User to Root (U2R) : L'attaquant a un accès local à la machine de la victime et tente d'obtenir les privilèges de super-utilisateur.
- Probing : L'attaquant tente d'obtenir des informations sur l'hôte cible.

L'ensemble de données de formation initiale contient 4.940000 connexions comprenant chacune 41 attributs. Chaque connexion code un flot de données (entre deux instants définis) entre une source (identifiée par son adresse IP) et une destination (également identifiée par son adresse IP), sous un protocole donné (TCP, UDP...).La

taille de la base de données est d'environ 744 Mb. Vue la difficulté de traitement d'un fichier d'une taille pareille avec un matériel standard, la plupart des chercheurs utilisent la version réduite du KDD'99 train qui représente 10% de celui-ci. Cette version est disponible en téléchargement par l'UCI [2] ($http://archive.ics.uci.edu/ml/datasets/KDD+Cup+1999+Data$). Nous avons utilisé cette partie de la base (10% de KDD'99) qui comprend 494,020 connexions, réparties en cinq classes (quatre attaques et une classe normale) comme décrit dans le tableau 6.1.

TABLE 6.1 – Nombre d'instances et distribution dans KDDCup'99 Train 10%

Classe	Normal	Dos	R2L	U2R	Probing	Total
Taille	9727	39145	1126	52	4107	494020
Distribution (%)	19.69	79.23	0.22	0.01	0.83	100

Les deux semaines de données de test comprennent environ 2 millions d'enregistrements. Il est important de noter que l'ensemble de données de test n'a pas la même distribution de probabilités que les données d'entraînement. Il inclut des types d'attaques spécifiques, qui ne sont pas présentes dans les données d'apprentissage, et qui rendent la tâche plus réaliste.

La base de données KDD99 recense, donc, 38 attaques possibles, dont, un nombre total de 24 types d'attaques pour l'apprentissage, et 14 types supplémentaires pour les données de test. Ces attaques, ainsi que leurs familles sont listées dans le tableau 6.2.

Certains experts de la détection d'intrusions croient que la plupart des attaques sont de nouvelles variantes d'attaques connues et que les signatures des attaques connues peuvent être suffisantes pour capturer ces nouvelles variantes. L'un des ensembles de test les plus utilisés est l'ensemble KDDcup'99 sans étiquette 10% (KDDcup'99 unlabeled 10%). Cet ensemble comprend 10% des données recueillies lors des tests.

Attributs utilisés

Chaque " connexion " est caractérisée par 41 attributs tels que sa durée, le type du protocole, etc. Ces attributs ont été fixés suite à un travail de fouille de données effectué par Lee et al. [116]. A partir des valeurs de ces attributs, chaque "connexion" dans KDD99 est considérée comme étant une "connexion" normale ou bien une attaque.

Les attributs caractérisant chaque "connexion", sont détaillés dans le tableau 6.3. Certains attributs sont de type discret (admettant un nombre fini de valeurs), d'autres sont de type continu.

Lors de nos expérimentations, nous avons choisi d'utiliser l'ensemble d'attributs proposés par D. Denning [6]. Cet ensemble comprend 16 attributs des 41 qui définissent une connexion. Ces attributs choisis peuvent être subdivisés en trois catégories :

– **Les compteurs d'événements :** le nombre d'occurrences d'un événement au cours d'une période donnée ;

TABLE 6.2 – Classes d'attaques dans KDD'99

DOS	Pribing	R2L	U2R
Apache2	Ipsweep	Ftp_write	Buffer_overflow
Back	Mscan	Guess_passwd	Httptunnel
Land	Nmap	Imap	Loadmodule
Mailbomb	Portsweep	Multihop	Xterm
Neptune	Saint	Named	Perl
Pod	Satan	Phf	Ps
Processtable		Dict	Rootkit
Smurf		Snmpguess	
Teardrop		Spy	
Udpstorm		Sqlattack	
		Warezclient	
		Warezmaster	
		Xlock	
		Xsnoop	
		Guest	

– **Les compteurs de durée :** temps entre deux événements ;
– **Les compteurs de ressources :** le nombre de ressources utilisées par une entité ;

6.2 Mode expérimental

Pour réaliser les différents tests, nous avons téléchargé KDD cup'99 à partir des données KDD sur le site UCI Machine Learning Repository [2]. Ces données contiennent les ensembles dits d'entrainement et ceux de tests. Les approches de sélection d'instances et le classificateur de 1NN ont été aussi implémentés en utilisant le langage Matlab [86] [179].

Nous avons choisi pour effectuer nos expérimentations de nous baser sur le schéma 6.1.

Nous avons d'abord appliqué les différentes approches de sélection d'instances proposées, une à une, sur l'enssemble d'apprentissage initial, pour en tirer des ensembles d'apprentissage réduits. Ces ensembles résultants serviront, par la suite, d'ensembles d'apprentissage pour le classificateur 1NN.

Nous avons analysé les ensembles résultants en détaillant leur composition, et donc, les proportions de présence des différentes classes d'attaques dans la composition finale de l'ensemble réduit, ainsi que le taux de réduction de ce dernier par rapport à l'ensemble initial.

Ensuite, nous avons utilisé le classificateur 1NN muni des différents ensembles réduits pour classer les trois ensembles de tests. Ceci nous a permi d'évaluer les résultats de classification globale, et le temps necessaire pour les réaliser, ainsi que les taux de détection d'attaques, de faux positifs et de classification de chaque classe.

TABLE 6.3 – Liste des attributs de KDD'99

Attribut	Désignation	Description
A1	duration	durée de la "connexion" (nombre de secondes)
A2	protocol_type	type du protocole, ex. tcp, udp, icmp...
A3	service	service réseau (destination) ex. http, telnet
A4	flag	statut de la "connexion" (normal ou erreur)
A5	src_bytes	nombre de données (en octets) de la source vers la destination
A6	dst_bytes	nombre de données (en octets) de la destination vers la source
A7	land	1 si la "connexion " est de/vers le même hôte/port ; 0 sinon
A8	wrong_fragment	nombre de fragments "erronés"
A9	urgent	nombre de paquets urgents
A10	hot	nombre d'indicateurs "hot"
A11	num_failed_logins	nombre d'essais login ratés
A12	logged_in	1 si succès du login ; 0 sinon
A13	num_compromised	nombre de conditions de "compromis"
A14	root_shell	1 si la racine shell est obtenue ; 0 sinon
A15	su_attempted	1 s'il y'a tentative de la commande "racine su" ; 0 sinon
A16	num_root	nombre d'accès à la "racine"
A17	num_file_creations	nombre de créations d'opérations de fichiers
A18	num_shells	nombre de shell prompts
A19	num_access_files	nombre d'opérations sur les fichiers de contrôle d'accès
A20	num_outbound_cmds	nombre de commandes outbound dans une session ftp
A21	is_hot_login	1 si le login appartient à la liste "hot" ; 0 sinon
A22	is_guest_login	1 si le login est login "invité" ; 0 sinon
A23	count	nombre de connexions pour le même hôte
A24	srv_count	nombre de connexions pour le même service
A25	serror_rate	% de connexions pour le même hôte ayant l'erreur "SYN"
A26	srv_serror_rate	% de connexions pour le même service ayant l'erreur "SYN"
A27	rerror_rate	% de connexions pour le même hôte ayant l'erreur "REJ"
A28	srv_rerror_rate	% de connexions pour le même service ayant l'erreur "REJ"
A29	same_srv_rate	% de connexions pour le même hôte utilisant le même service
A30	diff_srv_rate	% de connexions pour le même hôte utilisant différents services
A31	srv_diff_host_rate	% de connexions pour le même service utilisant différents hôtes
A32	dst_host_count	nombre de connexions pour le même hôte
A33	dst_host_srv_count	nombre de connexions pour le même hôte utilisant le même service
A34	dst_host_same_srv_rate	% de connexions pour le même hôte utilisant le même service
A35	dst_host_diff_srv_rate	% de connexions pour le même hôte utilisant différents services
A36	dst_host_same_src_port_rate	% de connexions pour le même hôte ayant le port src
A37	dst_host_srv_diff_host_rate	% de connexions pour le même hôte et le même service utilisant différents hôtes
A38	dst_host_serror_rate	% de connexions pour le même hôte ayant l'erreur" SYN"
A39	dst_host_srv_serror_rate	% de connexions pour le même hôte et le même service ayant l'erreur "SYN"
A40	dst_host_rerror_rate	% de connexions pour le même hôte ayant l' erreur "REJ"
A41	dst_host_srv_rerror_rate	% de connexions pour le même hôte et le même service ayant l'erreur "REJ"

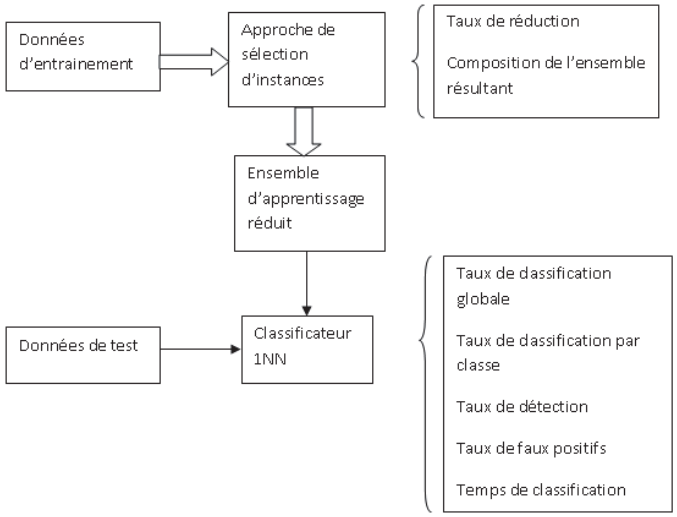

FIGURE 6.1 – Schéma de Test

6.3 Résultats et discussions

Afin d'évaluer nos propositions, nous avons utilisé trois principaux indicateurs de performances : la précision, le taux de détection et le taux d'alertes fausses positives.

La précision (Acc) est la capacité d'un système à classer une instance inconnue. Elle représente la capacité de ce système à distinguer les paquets normaux de ceux intrusifs. La précision d'un système peut être calculée comme le taux de connexions correctement classées parmi toutes les connexions.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{6.1}$$

Le taux de détection (DR) représente l'efficacité du système dans la reconnaissance des entrées intrusives. Ce taux permet, en utilisant un ensemble de connexions, toutes intrusives, de calculer le drgré de détection que le système peut offrir. Ce taux peut être calculé comme le quotient du nombre de connexions intrusives détectées par rapport au nombre total de connexions intrusives.

$$DR = \frac{TP}{TP + TN} \tag{6.2}$$

Le taux d'alarmes fausses positives (FPr) représente le taux d'erreur accompagnant la classification des instances non intrusives dans le système. Ce taux est tout simplement le ratio du nombre de connexions normales classées comme attaques comparé au nombre total de connexions normales.

$$FPr = \frac{FP}{FP + TN} \tag{6.3}$$

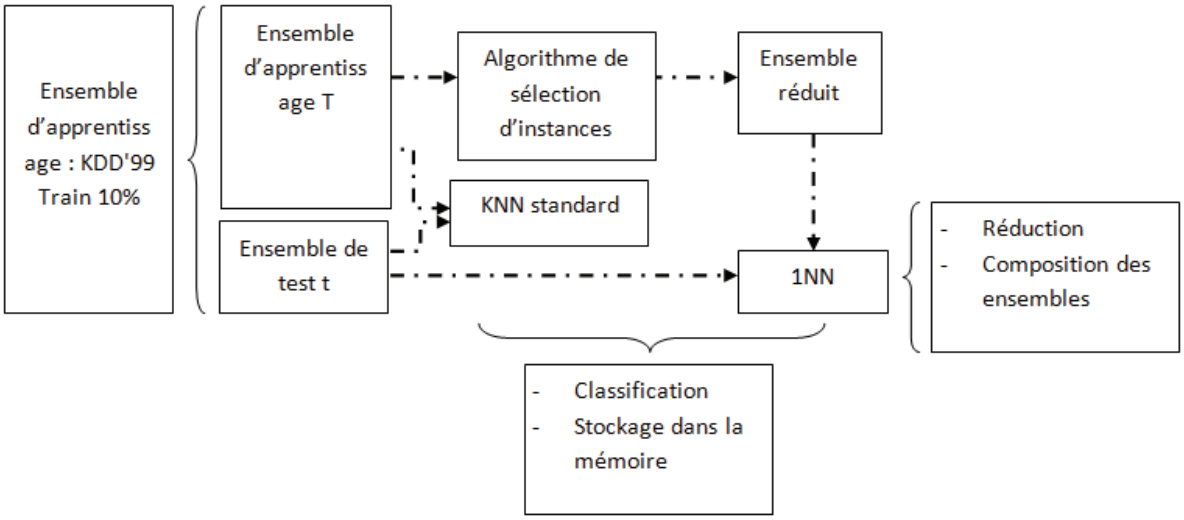

FIGURE 6.2 – Evaluation de l'impact de la sélection d'instances

Où :

- **TP (vrais positifs) :** Nombre de connexions qui ont été correctement classées comme attaques.
- **TN (vrais négatifs) :** Nombre de connexions qui ont été correctement classées comme normales.
- **FP (faux positifs) :** Nombre de connexions normales qui ont été classées comme attaques.
- **FN (faux négatif) :** Nombre de connexions d'attaques qui ont été classées comme normales.

Ces trois indicateurs donnent une idée globale des performances du système. Ils permettent de connaître sa capacité à distinguer une attaque à partir d'un paquet normal et de connaitre le taux d'erreurs dans les deux cas.

6.3.1 Impact de la Sélection d'instances

Pour étudier la contribution des algorithmes de sélection d'instances proposés dans la détection et les taux de faux positifs, et dans le stockage dans la mémoire, nous avons d'abord effectué des expériences en n'utilisant que les données d'entraînement.

L'ensemble d'entrainement, train KDD'99 10%, a été réparti en deux sous ensembles (Figure 6.2) : un grand ensemble, T, comprenant 70% des instances de ce dernier, servant comme ensemble d'entrainement et un petit ensemble, t, contenant 30% des instances, servant d'ensemble de test.

Les différents algorithmes ont été appliqués sur le premier ensemble afin de tenter de le réduire, et les ensembles réduits ont été testés sur l'ensemble de test dérivé. Le

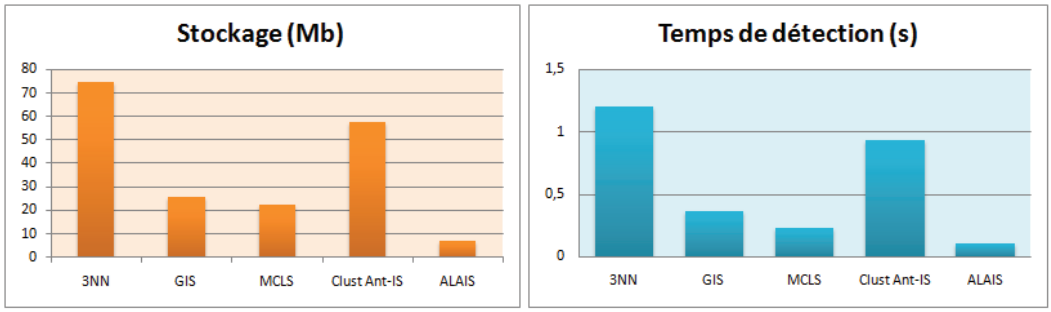

FIGURE 6.3 – Comparaison des coûts de calcul

classificateur KNN standard, par contre, utilise tout l'ensemble T comme ensemble d'apprentissage pour la classification de t. Nous avons comparé les résultats offerts par ces algorithmes avec ceux obtenus pour le classificateur KNN standard. Le tableau 6.5 et figure 6.3 comparent les résultats obtenus par les algorithmes de sélection d'instances proposés et l'algorithme KNN standard avec un k = 3, 5 et 10. Et la figure 6.4 montre la composition des différents ensembles résultants de l'application des techniques de sélection d'instances proposées.

Le tableau 6.5 montre clairement l'amélioration apportée par les approches de sélection par rapport au KNN standard. Cette amélioration provient principalement de la sélection d'instances offerte par les différentes approches, et qui est basée principalement sur l'exactitude de la classification.

TABLE 6.5 – Résultats expérimentaux pour GIS, MCLS, Clust Ant-IS, ALAIS et KNN

Algorithm	Normal	Dos	R2L	U2R	Probing	Acc	Taille
Clust Ant-IS	99.40	99.35	60.00	76.00	85.00	97.10	70683
ALAIS	99.40	99.98	67.00	94.00	89.00	98.90	48119
GIS	99.00	99.00	64.00	94.00	78.00	98.00	169888
MCLS	100	99.87	66.66	99.07	99.99	99.98	149786
kNN (k=3)	96.00	99.61	66.66	10.01	40.00	98.60	494020
kNN (k=5)	98.66	99.74	12.27	10.01	41.66	98.26	494020
kNN (k=10)	97.33	99.48	12.27	10.01	41.66	97.91	494020

Les approches de réduction doivent permettre la sélection des instances qui offrent la meilleure classification de celles qui restent, tout en ignorant, les instances conduisant à une mauvaise classification des autres.

Réduction

Le tableau 6.5 montre que, grâce à la sélection d'instances proposée, nous pouvons voir clairement que les ensembles d'entrainement ont été fortement réduits (en moyenne 70%). Aussi, il est aisé de constater, en se basant sur la figure 6.3, que le temps de

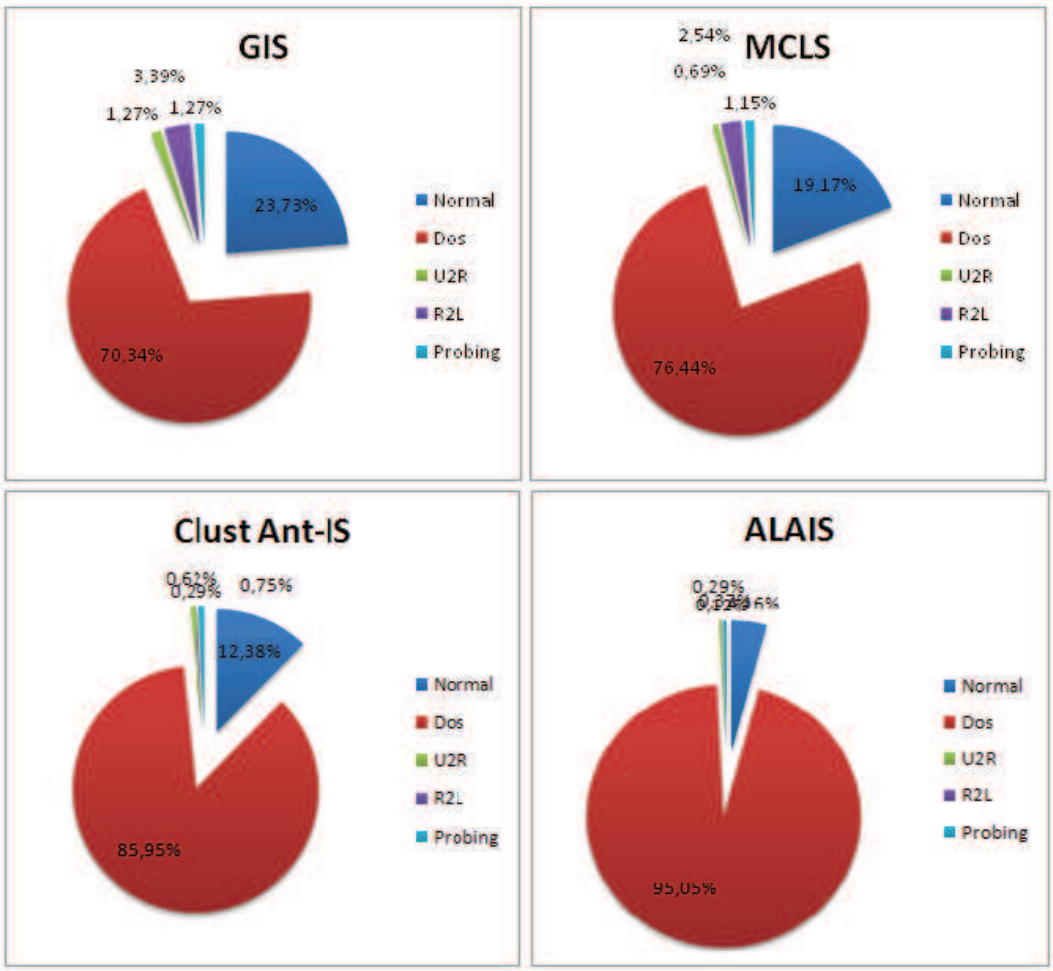

FIGURE 6.4 – Composition des ensembles issus des réductions

détection est lui aussi largement réduit. Enfin, la figure 6.3 montre également que le stockage de mémoire a été, lui aussi, réduit pour les quatre algorithmes.

En comparant les coûts de calcul de GIS et de MCLS, nous pouvons clairement constater l'amélioration apportée par MCLS, dans la composition de ces derniers. Rappelons que MCLS comprend une recherche guidée, qui améliore les résultats de la phase génétique, en réduisant l'ensemble résultant de cette phase tout en prenant en compte la précision de classification ainsi que du taux de faux positifs engendrés par cet ensemble. L'algorithme MCLS présente, par conséquent, des coûts de calcul meilleurs que ceux offerts par GIS.

L'algorithme Clust Ant-IS présente une réduction très proche de celles présentée par GIS, mais propose des coûts de calcul moins bons que ces derniers. Cependant, sa réduction reste assez significative comparée à l'ensemble initial.

ALAIS, améliore ces taux, et présente la meilleure réduction parmi les trois algorithmes. L'ensemble issu de cet algorithme occupe moins de mémoire pour son stockage et nécessite le plus bas temps de classification.

Analyse des ensembles réduits

Les quatre techniques de sélection ont conservé, globalement, la distribution des classes de l'ensemble initial. Dans la composition des quatre ensembles, la classe DOS est la plus représentée, suivie de la classe normale, ensuite de la classe R2L, et enfin des classes U2R et Probing, qui comportent le plus petit nombre d'instances dans l'ensemble initial.

Cette composition est aussi guidée par la construction des ensembles de tests utilisés pour la réalisation et l'évaluation de la réduction. Dans ces ensembles, le nombre d'attaques de dénie de service est largement majoritaire.

Les algorithmes MCLS et GIS présentent la même composition de leurs ensembles résultants, la réduction a donc touché les cinq classes, avec une légère amélioration pour la classe DOS.

Etant la classe incluant le plus d'attaques dans les ensembles de test, la classe DOS influe grandement sur la sélection des instances à inclure dans les ensembles d'entrainement issus des algorithmes Clust Ant-IS et ALAIS. Le choix étant principalement basé sur la précision de classification offerte par l'ensemble résultant, ce dernier est influencé par la classification de la classe majoritaire dans l'ensemble de test. Ceci implique que mieux les instances de la classe DOS sont correctement classées, meilleure est la précision de classification, et cela en dépit de la classification des autres classes.

Classification

Les résultats présentés pour les cinq classes montrent les performances des algorithmes proposés.

MCLS fournit une classification excellente de la classe normale, ce qui signifie que les

connexions non intrusives sont reconnues en tant que telles. Cela est dû à la fonction d'évaluation de la recherche locale proposée qui se base sur le taux de classification des instances. Avec ce taux de classification pour la classe normale, MCLS surpasse les autres méthodes comparées. Pour les quatre classes d'attaques, MCLS fournit également des taux de classification très élevés, en majorité, meilleurs que ceux offerts par les autres méthodes.

Les algorithmes Clust Ant-IS, ALAIS et GIS présentent des taux de classification très proches. Ces taux sont très bons comparés à ceux présentés par les trois variantes du KNN standard. Cependant ALAIS présente le meilleur taux de classification pour la classe DOS, comparé aux autres méthodes, cela est dû, rappelons-le, à la construction de son ensemble réduit, qui est principalement composé des instances de cette classe.

En termes de classification, nous pouvons dire que MCLS surpasse toutes les autres méthodes comparées pour les cinq classes, et que Clust Ant-IS permet la classification des instances des classes minoritaires.

Précision

MCLS offre un excellent taux d'exactitude allant jusqu'à 99,98% pour l'ensemble de données KDD'99. Cette précision est le meilleur taux présenté dans le tableau 6.5. La qualité de la précision de l'algorithme proposé provient principalement du choix de la première fonction objectif (fitness1) et la fonction objectif de la recherche locale proposée (fitness2). Tout comme pour les taux de classification,Clus Ant-IS, GIS et ALAIS présentent de très bons taux de précision, qui sont presque égaux.

6.3.2 Performances de détection

Le tableau 6.6 montre les résultats obtenus par les approches de réduction en utilisant l'ensemble de test KDD'99. Ces résultats sont comparés à ceux présentés par certains chercheurs pour leurs méthodes utilisant les mêmes données KDD cup'99.

L'ensemble d'entrainement KDD'99 train a été utilisé pour entrainer les différentes approches et l'ensemble KDD'99 test, afin d'évaluer les ensembles réduits résultants à travers la classification de ces données de test par le classificateur 1NN, entrainé grâce à ces ensembles. Chaque ensemble réduit a été testé à part, et a, donc, servi seul pour entrainer l'algorithme 1NN. Cette évaluation nous a permit d'estimer les taux de détection et de faux positifs conséquents.

Taux de vrais positifs / Taux de détection

L'algorithme GIS présente le plus mauvais taux de détection par rapport aux trois autres approches proposées, quoique ce taux reste meilleur que ceux proposés par quelques méthodes comparées, dont certaines qui sont récentes, telles que KNN-DS et Multistage Filter adaboost, ainsi qu'un taux acceptable de faux positifs. Cependant, ces taux ont été améliorés grâce à l'algorithme MCLS.

TABLE 6.6 – Taux de détection et de faux positifs

Algorithme	DR	FPr
GIS (Our)	98.02	1.05
MCLS (Our)	99.87	0
Clust Ant-IS (Our)	98.64	0.53
ALAIS (Our)	99.38	0.60
TCM−KNN [118]	99.7	0
GACL [66]	91.99	3.73
GACL−Anomaly [66]	92.16	3.86
SF−KNN [127]	91.05	2
KMNB [71]	99.80	0.009
MHLC [193]	99.21	3.20
EFCM [194]	99.48	1.03
TANN [67]	98.95	0.8
KM−KNN [192]	98.68	0.98
KNN−DS [69]	91.14	2.74
Multistage Filter adaboost [65]	82.52	1.20
Mem−Bayes [70]	70.89	2.88
NSGA−II [49]	99.84	0
SVM	99.5	1.00
Neural Nets	99.8	0.80

L'algorithme MCLS, qui est une amélioration de l'algorithme GIS, présente le taux de détection le plus élevé dans le tableau 6.6. qui équivaut à 99,87%. MCLS permet une meilleure reconnaissance des connexions intrusives parmi les méthodes comparées. Ceci constitue un atout considérable pour l'algorithme de détection d'intrusions.

L'algorithme Clust Ant-IS, présente lui aussi un bon taux de détection des quatre approches proposées, et qui est très bon comparé à la majorité des méthodes et techniques comparées.

L'approche que nous avons nommé ALAIS, améliore les taux atteints par l'algorithme précédent, et présente un taux de détection surpassant ceux offerts par la quasi majorité des méthodes comparées.

Taux de faux positifs

Comme cité précédemment, le taux de faux positifs est relatif à la classification des paquets non intrusifs. Ceci revient à dire qu'un ensemble d'entrainement bien construit devrait entre autres permettre une classification aussi infaillible que possible des instances non intrusives.

Tout comme TCM-KNN, MCLS permet de traiter efficacement le problème de fausses alertes avec un taux de faux positifs de 0%, ce qui est évidemment le meilleur taux indiqué dans le tableau 6.6. Ce taux soutient les performances de MCLS, dont le principal objectif est de réduire le taux d'alertes fausses positives.

Les algorithmes GIS et Clust Ant-IS, présentent un très faible taux de faux positifs, leur permettant d'être très bien classés parmi les approches comparées.

L'algorithme ALAIS présente le plus mauvais taux de faux positifs comparé aux trois autres approches suggérées. Ceci revient principalement à la composition de son ensemble réduit. Rappelons que les classes d'attaques et précisément la classe DOS représente la partie écrasante dans l'ensemble réduit issu de cet algorithme ce qui a influé sur la classification des instances normales.

6.4 Conclusion

Dans ce chapitre, les approches proposées ont été évaluées et comparées au KNN standard et à d'autres techniques récentes et connues dans le domaine de la détection d'intrusions. Ces comparaisons ont été faites en utilisant l'ensemble de données KDD cup'99.

Les résultats ont confirmé l'amélioration atteinte à travers ces algorithmes. Les différentes approches ont prouvé leur capacité à réduire ou éliminer les alertes fausses positives à travers leurs modes de sélection d'instances, ainsi que leur aptitude à détecter les attaques et à les classer correctement.

Aussi, ces approches ont permi de réduire sensiblement l'ensemble d'entrainement de l'algorithme KNN permettant de réduire considérablement les coûts de calcul de ce dernier.

Chapitre 7

Conclusion

La détection d'intrusion est considérée comme l'un des piliers de la sécurité de l'information. Après son apparition dans les années 1980, de nombreuses contributions de chercheurs ont été publiées au fil des années, et certaines d'entre elles ont été transformées en systèmes de détection d'intrusions commerciaux.

Depuis lors, les IDS jouent un rôle important dans l'architecture de sécurité dans de nombreuses organisations. Cependant, la plupart des IDS souffrent d'un problème commun, qui est le taux élevé de fausses alarmes.

Des milliers de fausses alarmes peuvent être générées à partir d'un IDS à l'intérieur d'un réseau opérationnel chaque jour. Une grande quantité de fausses alarmes peut rajouter une lourde charge de travail pour les administrateurs réseau. A force de ne traiter que ces fausses alarmes, les administrateurs peuvent baisser leur vigilance, et risquent d'avoir tendance à ignorer certaines vraies alarmes.

Dans notre étude, nous avons examiné certains concepts de base dans l'historique de la détection d'intrusions. Diverses techniques s'étant intéressées à la détection d'intrusions, développées par plusieurs chercheurs, ont été présentées au chapitre 2. Nous avons conclu qu'il était nécessaire de faire appel à un classificateur performant pour classer les alertes, dans le but de réduire le nombre d'alertes fausses positives.

Les KNN sont des classificateurs les plus simples et les plus performants, qui ont démontré leur efficacité dans divers domaines, et qui a récemment reçu un grand interet de la part de la communauté de la détection d'intrusions.

Nous avons exposé le problème principal affectant ce classificateur, qui était la taille critique de son ensemble d'apprentissage. Nous avons ensuite abordé différentes propositions visant à proposer des solutions à ce problème. Afin de réduire l'ensemble d'apprentissage de ce classificateur et de le rendre encore plus efficace et réduire davantage le nombre de fausses alarmes et tenter d'automatiser entièrement le processus de réduction de l'alarme, nous avons proposé deux méthodes de sélection d'instances pour l'algorithme des k-plus proches voisins.

La première méthode de réduction que nous avons proposée se base sur le concept

de l'optimisation par colonies de fourmis. A notre connaissance, ce concept n'avait jamais été utilisé dans la sélection d'instances. Nous nous sommes basés sur le travail collaboratif des fourmis pour sélectionner les instances les plus pertinentes pour la classification des paquets entrants. Nous avons proposé d'améliorer la réduction offerte par cette méthode à l'aide de l'apprentissage actif, qui permet, de minimiser autant que possible le nombre d'instances sélectionnées, en construisant de manière incrémentale l'ensemble d'apprentissage, pour le faire, l'apprentissage actif a servi à orienter les recherches des fourmis.

La seconde méthode que nous avons proposée s'est basée sur le concept mémétique. Nous avons proposé pour cela une recherche locale contrôlée. Cette recherche invoque une mutation spécialement adaptée au problème, qui contrôle la création du voisinage, en se basant sur les solutions déjà validées lors du processus génétique pour créer de nouvelles propositions.

Pour vérifier l'efficacité de nos méthodes proposées, nous avons mené des expériences sur les données de trafic réseau DARPA 1999.

Les résultats des différentes expérimentations ont montré que les méthodes de réduction proposées pouvaient produire des taux de réduction de fausses alarmes tout à fait cohérents, même lorsque différents ensembles de données ont été utilisés. Les taux de réduction obtenus par nos algorithmes sont meilleurs que ceux présentés par les approches de recherches antérieures examinées dans le Chapitre 3.

Ces résultats prometteurs ont soutenu notre argument selon lequel le choix de l'ensemble d'apprentissage pouvait influer d'une manière très importante sur les résultats de classification du KNN, et donc sur le nombre d'alarmes fausses positives issues de cette classification. Après avoir mené des expériences pour vérifier l'efficacité des méthodes proposées, nous pouvons avancer qu'elles peuvent réduire considérablement le nombre de fausses alarmes issues d'un IDS sans affecter le niveau de sécurité offert par les IDS.

Nous proposons comme objectifs futurs d'améliorer ces approches sur un environnement temps-réel. Pour une exécution dans un environnement réel, il serait interessant de parfaire les données de l'ensemble, qui doivent être similaires aux différentes situations rencontrées sur les réseaux réels. Pour le faire, on devrait construire un jeu de données de fausses alarmes recueillies auprès d'un IDS aussi précis que possible et aussi semblables à des situations réelles que possible.

Enfin, pour atteindre une exécution temps réel, il sera aussi inreressant de travailler sur la classification incrémentale en ligne, ou aussi en intégrant un calcul de similarité plus rapide en impliquant plus de parallélisme grâce à des systèmes dédiés.

Bibliographie

[1] *Alpaydin E. Introduction to Machine Learning, MIT Press, Cambridge, MA. 2004.*

[2] Asunción, a., & newman, d. j. (2007). uci machine learning repository. irvine, ca. university of california, school of information and computer science. [http ://www.ics.uci.edu/ mlearn/mlrepository.html].

[3] Bishop, c. m. neural networks for pattern recognition. oxford university press, 1995.

[4] Comission of the european communities. information technology security evaluation criteria. version 2.1, 1991.

[5] Fayyad u. m., g. piatetsky-shapiro, et p. smyth. from data mining to knowledge discovery : an overview. advances in knowledge discovery and data mining, 1-34, mit press. 1996.

[6] Hettich s. et bay s. d. the uci kdd archive [http ://kdd.ics.uci.edu], irvine, ca : University of california, department of information and computer science, 1999.

[7] Martin roesch. snort - lightweight intrusion detection for networks. in usenix lisa 99, 1999.

[8] Mining audit data to build intrusion detection models. wenke lee and salvatore j. stolfo and kui w. mok. 1998.

[9] Nist. icat metabase. web page at http ://icat.nist.gov/, 20002004.

[10] *Swatch : Simple Watchdog. http ://swatch.sourceforge.net.*

[11] *Witten I.H. & Frank E. Data Mining : Practical Machine Learning Tools and Techniques, Morgan Kauffman, Los Altos, CA. 2005.*

[12] *Zadeh P.R., Tang L. & Liu H. Cross-Validation, Arizona State University.2008.*

[13] *F. Cuppens and A. Miége. Alert Correlation in a Cooperative Intrusion Detection Framework. Proceedings of the 2002 IEEE Symposium on Security and Privacy. 187 200.*, 2002.

[14] *P. Ning. Y. Cui and D. S. Reeves. Constructing Attack Scenarios through Correlation of Íntrusion Alerts. Proceedings of the 9th ACM conference on Computer and communications security. ACM Press. 245 254.*, 2002.

[15] *Ming-Yang Su, Kai-Chi Chang, Hua-Fu Wei, Chun-Yuen Lin, A Real-time Network Intrusion Detection System Based on Incremental Mining Approach, 1-4244-2415-3/08. IEEE. 179 - 184*, 2008.

[16] *Ratchadaporn Amornchewin, Worapoj Kreesuradej, Probability-based incremental association rule discovery algorithm, International Symposium on Computer Science and its Applications, 978-0-7695-3428-2/08. IEEE. 212 - 215*, 2008.

[17] *Wang Taihua, Guo Fan, Associating IDS Alerts by an Improved Apriori Algorithm, Third International Symposium on Intelligent Information Technology and Security Informatics, 978-0-7695-4020-7/10. IEEE. 478 - 482*, 2010.

[18] *Zhang yanyan,Yao Yuan, Study of Database Intrusion Detection Based on Improved Association Rule Agorithm,978-1-4244-5540-9/10. IEEE. 673 - 676*, 2010.

[19] *B. Morin and H. Debar. Correlation of Intrusion Symptoms : an Application of Chronicles. Proceedings of the sixth International Symposium on Recent Advances in Intrusion Detection (RAID). Springer Verlag. 97 112.*, November 2003.

[20] *X. Qin and W. Lee. Statistical Causality Analysis of INFOSEC Alert Data. Proceedings of the sixth International Symposium on Recent Advances in Intrusion Detection (RAID). Springer Verlag. 73 93.*, November 2003.

[21] *A. Valdes and K. Skinner. Probabilistic Alert Correlation. Proceedings of the fourth International Symposium on Recent Advances in Intrusion Detection (RAID). Springer Verlag. 54-68.*, October 2001.

[22] *H. Debar and A. Wespi. Aggregation and Correlation of Intrusion-Detection Alerts. Proceedings of the fourth International Symposium on Recent Advances in Intrusion Detection (RAID). Springer Verlag. 85-103.*, October 2001.

[23] *P. A. Porras, M. W. Fong and A. Valdes. A Mission-Impact-Based Approach to INFOSEC Alarm Correlation. Proceedings of the fifth International Symposium on Recent Advances in Intrusion Detection (RAID). Springer Verlag. 95114.*, October 2002.

[24] Alharby A. and Imai H. Ids false alarm reduction using continuous and discontinuous patterns. *Lecture Notes in Computer Science, Springer-Verlag*, 3531 :192–205, 2005.

[25] Inan C. Aci M. and Avci M. A hybrid classification method of k nearest neighbor, bayesian methods and genetic algorithm. *Expert Systems with Applications*, 37 :5061–5067, 2010.

[26] Gehrke J. Gunopulos D. & Raghavan P. Agrawal, R. Automatic subspace clustering of high dimensional data for data mining applications. In *SIGMOD 1998. 94-105.*, 1998.

[27] R. Agrawal, R. & Srikant. Fast algorithms for mining association rules. In *20th International Conference of Very Large Data Bases (VLDB). 487-499*, 1994.

[28] Fithen W. McHugh J. Pickel J. Allen J., Christie A. and Stoner E. State of the practice of intrusion detection technologies. Technical report, Technical report CMU/SEI99-TR-028, ESC-99-028, Carnegie Mellon, Software Engineering Institute, Pittsburgh, Pennsylvania, 1999.

[29] Fabrizio Angiulli. Fast condensed nearest neighbor rule. Technical report, Proceedings of the 22 nd International Conference on Machine Learning, Bonn, Germany, 2005.

[30] Breunig M. M. Kriegel H.-P. & Sander J. Ankerst, M. Optics : ordering points to identify the clustering structure. *SIGMOD Rec.*, 28(2) :49–60, 1999.

[31] S. Aron S.-Deneubourg J.L. Goss and J.M Pasteels. Selforganized shortcuts in the argentine ant. *Naturwissenschaften*, 76 :579–581, 1989.

[32] Stefan Axelsson. Intrusion detection systems : A survey and taxonomy. *Departement of Computer Engineering Chalmers University of Technology, Goteborg, Sweden*, . :15–23, 14 March 2000.

[33] Stefan Axelsson. *Understanding Intrusion Detection Through Visualization*. PhD thesis, Chalmers University of Technology, 2005.

[34] Jajodia S.-Popyack L. Barbara D., Julia Couto J. and Wu N. Adam : detecting intrusions by data mining. *Proceedings of the 2001 IEEE workshop on information assurance and security*, NY, USA :310 – 18, 2001.

[35] Steven M. Bellowin. Packets found on an internet. *Computer Communications Review*, 23(3) :26 31, 1993.

[36] Gael Loosli Benoit Gandar and Guillaume Deffuant. Sélection de points en apprentissage actif, discrépance et dispersion : des critères optimaux. Technical report, MajesSTIC, 2009.

[37] L Todd Heberlein Biswanath Mukherjee and Karl Levitt. Network intrusion detection. *IEEE Network*, 8(3) :26–41, May/June 1994.

[38] Christiansen A. Skorupka C. Talbot L. Bloedorn E., Hill B. and Tivel J. Data mining for improving intrusion detection. Technical report, Technical report, MITRE Corporation, 2000.

[39] Dasarathy B.V. *Nearest Neighbor (NN) norms : NN pattern classification techniques*. LosAlamitos, CA, 1990.

[40] Dasarathy B.V. Minimal consistent subset (mcs) identification for optimal nearest neighbor decision systems design. *IEEE Transactions on Systems, Man and Cybernetics*, 25 :511–517, 1994.

[41] Herrera F. Cano J.F. and Lozano M. Using evolutionaru algorithms as instance selection for data reduction in kdd : An experimental study. *IEEE Transactions on Evolutionary Computation*, 7 (6) :561–575, 2003.

[42] Vicente Cerveron and Francesco J. Ferri. Another move toward the minimum consistent subset : A tabu search approach to the condensed nearest neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, 31 :408–413, 2010.

[43] Yu-Jung Chen Cheng-San Yang, Li-Yeh Chuang and Cheng-Hong Yang. Feature selection using memetic algorithms. *Third 2008 International Conference on Convergence and Hybrid Information Technology*, . :408–423, 2008.

[44] T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *IEEE transaction on Information theory*, IT-13 :21–27, Jan 1967.

[45] Swonger C.W. Sample set condensation for a condensed nearest neighbor decision rule for pattern recognition. *Frontiers of Pattern Recognition Ed. S. Watanable, Academic Press*, 1 :511–519, 1972.

[46] T. Frivold D. Anderson and A. Valdes. Next-generation intrusion-detection expert system (nides). Technical report, Computer Science Laboratory, SRI International, Menlo Park, CA 94025-3493, USA, May 1995.

[47] Sanchez J.S. Dasarathy B.V. and Townsend S. Nearest neighbor editing and condensing tools- synergy exploitation. *Pattern Analysis & Applications*, 3 :19–30, 2003.

[48] Deepika Dave and Prof. Vineet Richhariya. Intrusion detection with knn classification and ds- theory. *International Journal of Computer Science and Information Technology & Security (IJCSITS)*, 2 N. 2 :274–281, 2012.

[49] A. Agarwal S. et Meyarivan T. Deb K. Pratap. A fast and elitist multiobjective genetic algorithm : Nsga-ii. *IEEE Transactions on Information Evolutionary Computation*, 6 (2) :182–197, 2002.

[50] Dacier M. Debar H. and Wespi A. A revised taxonomy for intrusion detection systems. *Annales des Telecommunications*, 55 :7–8, 2000.

[51] Harold Javitz Ann Tamaru Debra Anderson, Teresa F.Lunt and Alfonso Valdes. Detecting unusual progragram behavior using the statistical component of the next-generation intrusion detection system (nides). Technical report, Computer Science Laboratory, SRI Internationnal, Menlo Park, CA, USA, May 1995.

[52] R. Deneubourg J.L. Beckers and S. Goss. Trails and u-turns in the selection of the shortest path by the ant lasius niger. *Journal of Theoretical Biology*, 159 :397–415, 1992.

[53] Dorothy Denning. An intrusion detection models. *IEEE, transaction on software engineering*, 13(2) :222–232, 1987.

[54] Kittler J. Devijver P.A. *Pattern Recognition : A statistical approach.* Englewood Cliffs, NJ, 1982.

[55] D. Ventre dir. *Cyberguerre et guerre de l'information.* Hermes Science, 2010.

[56] Marco Dorigo and Luca Maria Gambardella. Ant colonies for the traveling salesman problem. *BioSystems*, 1997.

[57] Marco Dorigo and Thomas Stützle. *Ant Colony Optimization*. July 2004.

[58] Amoroso E. Intrusion detection. Technical report, Intrusion.net Books, January 1999.

[59] Denning D. E. A lattice model of secure information flow. *Communications of the ACM*, 19 (5) :236 – 243, 1975.

[60] M. Prerau L. Portnoy E. Eskin, A. Arnold and S. Stolfo. *A geometric framework for unsupervised anomaly detection : Detecting intrusions in unlabeled data.* Applications of Data Mining in Computer Security, 2002.

[61] N. Melab G. Luque E. G. Talbi, E. Alba. *Parallel Hybrid Metaheuristics : A New Class of Algorithms*. Wiley, 2005.

[62] L. J. Eshelman. The adaptive search algorithm : How to have safe search when engaging in nontraditional genetic recombination. *in Foundations of Genetic Algorithms, G. J. E. Rawlins, Ed. San Mateo, CA : Morgan Kauffman*, 1 :265–283, 1991.

[63] Stephen E.Smaha. Haystack : An intrusion detection system. *21th National Computer Security Conference*, . :37–44, 1988.

[64] Kriegel H.-P. Sander J. & Xu X. Ester, M. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD-96. 226-231*, 1996.

[65] Natesan P. et Balasubramanie P. Multi stage filter using enhanced adaboost for network intrusion detection. *International Journal of Network Security and Its Applications*, 4 (3) :121–135, 2012.

[66] Shirazi H. M. et Kalaji Y. An intelligent intrusion detection system using genetic algorithms and features selection. *Majlesi Journal of Electrical Engineering*, 4 :33–43, 2010.

[67] Tsai C. F. et Lin C. Y. A triangle area based nearest neighbours approach to intrusion detection. *Pattern Recognition*, 43 :222–229, 2010.

[68] Moscato P. et Norman M. G. A mimetic approach for the travelling salesman problem - implementation of computational ecology for combinatorial optimisation on message-passing systems. In *the International Conference on Parallel Computing and Transputer Applications. IOS Press (Amsterdam)*, 1991.

[69] Deepika D. et Richhariya V. Intrusion detection with knn classification and ds-theory. *International Journal of Computer Science and Information Technology and Security*, 2 (2) :274–281, 2012.

[70] Shirazi H. M. Namadchian A. et Tehrani A. K. A combined anomaly base intrusion detection using memetic algorithm and bayesian networks. *International Journal of Machine Learning and Computing*, 2 (5) :706–710, 2012.

[71] Muda. Z. Yassin. W. Sulaiman M. N. et Udzir N. I. A k-means and naive bayes learning approach for better intrusion detection. *Information Technology Journal*, 10 (3) :648–655, 2011.

[72] C.Kruegel F. Valeur, G. Vigna and R. Kemmerer. A comprehensive approach to intrusion detection alert correlation. *IEEE Transactions on Dependable and Secure Computing*, 1(3) :146–169, 2004.

[73] Hatem A. Fayed and Amir F. Atiya. A novel template reduction approach for the -nearest neighbor method. *IEEE Transactions On Neural Networks*, 20 (5) :890–896, MAY 2009.

[74] Wilfong G. Nearest neighbor problems. *Journal of Computational Geometry & Applications*, 2 (2) :383–416, 1992.

[75] G.Gates. The reduced nearest neighbor rule. *IEEE Transactions on Information Theory*, 18 :431–433, 1972.

[76] Schwartzbard A. Ghosh A. A study in using neural networks for anomaly and misuse detection. In *the eighth USENIX security symposium, Washington, USA. 14151*, 1999.

[77] R. Gil-Pita and X. Yao. Using a genetic algorithm for editing k-nearest neighbor classifiers. *IDEAL 2007, LNCS*, 4881 :1141–1150, 2007.

[78] Steve Eckmann Giovanni Vigna and Richard A. Kemmerer. The stat tool suite. In *DISCEX 2000, Hilton Head, South Carolina,IEEE Computer Society Press*, January 2000.

[79] D.E. Goldberg. Genetic algorithms in search. *Optimisation and Machine Learning, Addison-Wesley Reading*, Massachusetts :., 1989.

[80] Aron S. Deneubourg J.L. Goss, S. and J.M Pasteels. Self-organized shortcuts in the argentine ant. *Naturwissenschaften*, 76 :579–581, 1989.

[81] TRW Defense System Groupe. Intrusion detection expert system feasability study. Technical report, Final report 46761, 1986.

[82] Nikravesh M. Guyon I., Gunn S. and Zadeh L.A. (eds). *Feature extraction : foundations and applications*. Springer, New York, 2006.

[83] Michael Hahsler and Kurt Hornik. Tsp - infrastructure for the traveling salesperson problem,. *Journal of Statistical Software*, 23(2) :1–21, 2007.

[84] Pei J. & Yin Y. Han, J. Mining frequent patterns without candidate generation. In *the 2000 ACM SIGMOD Intl. Conference on Management of Data, ACM Press. 1-12.*, 2000.

[85] M. Hasenjager and H. Ritter. Active learning with local models. *Neural Processing Letters*, 7 :107–117, 1998.

[86] R.L. Haupt and S.E. Haupt. *Practical Genetic Algorithms.* John Wiley & Sons, 2004.

[87] Maccabe A. Heady R., Luger G. and Sevilla M. The architecture of a network-level intrusion detection system. Technical report, Technical report, CS90-20, Department of Computer Science, University of New Mexico, Albuquerque, NM87131., 1990.

[88] Monique Becker Hervé Debar and Didier Siboni. A neural network component for an intrusion detection system. *Proceeding of the 1992 IEEE Computer Society Symposium on Research in Security and Privacy, Oakland,CA, USA*, . :240–250, May 1992.

[89] Shen-Shyang Ho and Harry Wechsler. Transductive confidence machine for active learning. *Proceedings of the International Joint Conference on Neural Networks*, 2 :1435 – 1440, 2003.

[90] Karla Hoffman and Manfred Padberg. Salesman problem. *Encyclopedia of Operations Research*, 2nd ed, 2000.

[91] Holger H. Hoos and Thomas Stützle. *Stochastic Local Search : Foundations and Applications.* Elsevier, 2004.

[92] Koral Ilgrun. Ustat : A real-time intrusion detection system for unix. *Proceeding of the 1993 IEEE Symposium on Security and Privacy, Oakland, California, USA*, pages 16–28, 24-26 May 1993.

[93] 07 janvier 2013 08 :01 ; IRIB Iran Radio Francophone : lundi. Cyberattaques us/israël : l'iran est prêt. *http ://french.irib.ir/analyses/commentaires/item/235344-rlundi, 07 janvier :08 :01, 2013.*

[94] Leyden J. Ids users swamped with false alerts. Technical report, http ://www.theregister.co.uk/2001/12/15/ids_users_swamped_with_false/, 15th December, 2001.

[95] J.Kim. Integrating Artificial Immune Algorithms for Intrusion Detection*. PhD thesis, University College London, 2002.*

[96] Salvador Garcia Joaquin Derrac and Francisco Herrera. Ifs-coco : Instance and feature selection based on cooperative coevolution with nearest neighbor rule. Pattern Recognition*, 43 :2082–2105, 2010.*

[97] Francisco Herrera Joaquín Derrac, Salvador García. Stratified prototype selection based on a steady-state memetic algorithm : a study of scalability. Memetic Comp*, 2 :183–199, 2010.*

[98] Fukunaga K. and Mantock J.M. Nonparametric data reduction. IEEE Transaction on Pattern Analysis and Machine Intelligence*, 6 (1) :115–118, 1984.*

[99] *Julisch K. Clustering intrusion detection alarms to support root cause analysis.* ACM Transactions on Information and System Security, *6(4) :443–471, 2003.*

[100] *Julisch K.* Using Root Cause Analysis to Handle Intrusion Detection Alarms. *PhD thesis, PhD thesis, University of Dortmund, Germany, 2003.*

[101] *Julisch K. Mining alarm clusters to improve alarm handling efficiency.* Proceedings of 17th Annual Computer Security Applications Conference, *2001 :12–21, December 2001.*

[102] *Julisch K. and Dacier M. Mining intrusion detection alarms for actionable knowledge.* Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, *pages 266–375, 2002.*

[103] *Kira K. and Rendell L. A practical approach to feature selection.* International conference on Machine Learning-ICML, *1 :249–256, 1992.*

[104] *Q.H. Wu K. Nikolaidis, J.Y. Goulermas. A class boundary preserving algorithm for data condensation.* Pattern Recognition, *44 :704–715, 2011.*

[105] *Gowda K.C. and Krishna G. The condensed nearest neighbor rule using the concept of mutual nearest neighborhood.* IEEE Transactions on Information Theory, *24 (4) :488–490, 1979.*

[106] *Richard A Kemmerer Koral Ilgrun and Philip A Porras. State transition analysis : rule-based intrusion detection approach.* IEEE Transactions on Software Engineering, *21(3) :181–199, March 1995.*

[107] *Liwei Kuang and Mohammad Zulkernine. An anomaly intrusion detection method using the csi-knn algorithm. In* SAC '08 Proceedings of the 2008 ACM symposium on Applied computing, pp. 921-926, *2008.*

[108] *Sandeep Kumar.* Classification and Detection of Computer Intrusions. *PhD thesis, Purdu University, West Lafayette, Indiana, USA, August 1995.*

[109] *Sandeep Kumar and Eugene H.Spafford. A software architecture to support misuse intrusion detection. Technical report, The COAST Project, Department of Computer Sciences, Purdu University, West Lafayette, IN, 47907-1398, USA, 17 March 1995.*

[110] *Sandeep Kumar and Eugene H.Spafford. A pattern matching model of misuse intrusion detection.* Proceeding of the 17th National Computer Security Conference, Baltimore MD, USA, *. :11–21, 1994.*

[111] *Sandeep Kumar and Eugene H.Spafford. An application of pattern matching in intrusion detection. technical report csd-tr-94-013, the coast project. Technical report, Dept of Computer Sciences, Purdue University, West Lafatette, IN, USA, June 1994.*

[112] Ludmila I. Kuncheva and James C. Bezdek. Nearest prototype classification : Clustering, genetic algorithms, or random search ? IEEE Transactions on Systems, Man and Cybernetics*, 28 :160–164, 1998.*

[113] Kwok Ho Law and Lam For Kwok. Ids false alarm filtering using knn classifier. Departement of Computer Science, City University of Hong Kong, Kowloon, Hong Kong*, . :114–121, 2004.*

[114] Kwok Ho Law and Lam For Kwok. Ids false alarm filtering using knn classifier. In WISA'04 Proceedings of the 5th international conference on Information Security Applications, pp. 114-121*, 2005.*

[115] Wenke Lee and Salvator J.Stolfo. Data minig approaches for intrusion detection. Proceeding of the 7th USENIX Security Symposium*, San Antonio, 1998.*

[116] Wenke Lee and Salvator J.Stolfo. A framework for constructing features and models for intrusion detection systems. Information and System Security*, 3(4) :227–261, 2000.*

[117] LEMONDE.FR. Stuxnet : une "bonne idée", juge un ancien patron de la cia. AFP*, page 5 Mars, 2012.*

[118] Yang Li and Li Guo. Tcm-knn algorithm for supervised network intrusion detection. computers & security*, 26 :459467, 2007.*

[119] Yihua Liao and V. Rao Vemuri. Use of k-nearest neighbor classifier for intrusion detection. Computers & Security*, 21 N. 5 :439–448, 2002.*

[120] Shasha Mao Lin Xiong, L. C. Jiao and Li Zhang. Active learning based on coupled knn pseudo pruning. Neural Computing and Applications*, 21(7) :1669–1686, 2012.*

[121] Cunningham P.K. Lippmann R.P. Improving intrusion detection performance using keyword selection and neural networks. International Journal of Computer and Telecommunications Networking*, 34(4) :597–603, 2000.*

[122] Motoda H Liu H. Instance selection and construction for data mining*. The Springer international series in engineering and computer science, 2001.*

[123] Motoda H Liu H. On issues of instance selection. DataMining Knowledge Discovery*, 6 (2) :115–130, 2002.*

[124] Motoda H (eds) Liu H. Computational methods of feature selection. CRC Data mining and knowledge discovery series*, Chapman & Hall :London, 2007.*

[125] Le quotidien Libanais d'expression française LOrient Le Jour. Israel reconnaît avoir été la cible de millions de cyber-attaques. AFP, 18/11/2012, 17h34*, Samedi 12 janvier, 2013.*

[126] Susan M.B. Luo J.X. Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection. International Journal of Intelligent Systems*, 15(8) :687–704, 2000.*

[127] *Shirazi H. M. Anomaly intrusion detection system using information theory, k-nn and kmc algorithms.* Australian Journal of Basic and Applied Sciences, *3 (3) :2581–2597, 2009.*

[128] *G. Di Caro M. Dorigo and L. M. Gambardella. Ant algorithms for discrete optimization.* Artificial Life, *5 (2) :137–172, 1999.*

[129] *S. Markovich M. Lindenbaum and D. Rusakov. Selective sampling for nearest neighbor classifiers.* Machine Learning, *54 (2) :125 – 152, 2004.*

[130] *Chan P. Mahoney M. Learning nonstationary models of normal network traffiic for detecting novel attacks. In* the eighth ACM SIGKDD international conference on knowledge discovery and data mining, Edmonton, Canada. 37685, *2002.*

[131] *Christensen M. Zerkle D. Manganaris, S. and K. Hermiz. A data mining analysis of rtid alarms.* Computer Networks, *34(4) :571–577, 2000.*

[132] *Zerkle D. Manganaris S., Christensen M. and Hermiz K. A data mining analysis of rtid alarms.* Computer Networks : The International Journal of Computer and Telecommunications Networking, *34(4) :571577, 2000.*

[133] *Todd Ellis Ivan Krsul Mark Ceosbie, Bryn Dole and Eugene Spafford. Idiot user guide. Technical report, The COAST Project, Dept of Computer Science, Purdu University, West Lafayette, IN, USA, 4 September 1996.*

[134] *Mary E.Hanna Michael M.Sebring, Eric Shellhouse and R. Alan Whitehurst. Experts systems in intrusion detection : A case study.* Proceeding of the 11th National Computer Security Conference, Baltimore, Maryland, *. :74–81, 1988.*

[135] *A. Miloud-Aouidate and A. R. Baba-Ali. An efficient ant colony instance selection algorithm for knn classification. has been accepted for publication in the International Journal of Applied Metaheuristic Computing 4(3), (Mai 2013).*

[136] *A. Miloud-Aouidate and A. R. Baba-Ali. Ids false alarm reduction using a knn-memetic algorithm. has been accepted for publication in the International Journal of Metaheuristics. Accepted (June 2013).*

[137] *A. Miloud-Aouidate and A. R. Baba-Ali. Ant colony prototype reduction algorithm for knn classification.* 2012 IEEE 15th International Conference on Computational Science and Engineering, *pages 289–294, 2012.*

[138] *A. Miloud-Aouidate and A. R. Baba-Ali. A hybrid knn-ant colony optimization algorithm for prototype selection.* ICONIP 2012, Part III, *LNCS 7665 :307–314, 2012.*

[139] *Le Monde.fr. Les etats-unis pourraient répliquer militairement à une potentielle cyberattaque.* AFP, *31 Mai :22 :58, 2011.*

[140] *Debar H. Morin B., Mé L. and Ducasse M. M2d2 : A formal data model for ids alert correlation.* In Recent Advances in Intrusion Detection, Lecture Notes in Computer Science, *2516 :115–137, 2002.*

[141] P. Moscato. On evolution, search, optimization, genetic algorithms and martial arts : towards memetic algorithms. Technical report, Technical Report C3P 826, Pasadena, CA, 1989.

[142] Pablo Moscato. On evolution, search, optimization, genetic algorithms and martial arts - towards memetic algorithms. Technical report, Pasadena : California Institute of Technologie, 1989.

[143] Janoski G.H. Mukkamala S. Intrusion detection : support vector machines and neural networks. In the IEEE international joint conference on neural networks, Honolulu, USA.*, 2002.*

[144] Jankowski N. and Grochowski M. Comparison of instances seletion algorithms i. algorithms survey. International Conference on Artificial Intelligence and Soft Computing*, 1 :598–603, 2004.*

[145] J. Smith N. Krasnogor. A tutorial for competent memetic algorithms : model, taxonomy, and design issues. IEEE Trans. Evol. Comput*, 9 N. 5 :474–488, 2005.*

[146] Goil S. & Choudhary A. N. Nagesh, H. S. A scalable parallel subspace clustering algorithm for massive data sets. In International Conference on Parallel Processing*, 2000.*

[147] Snyder J. Newman D. and Thayer R. Crying wolf : False alarms hide attacks. Technical report, http ://www.nwfusion.com/techinsider/2002/0624security1.html, 24th June, 2002.

[148] Goulermas J.Y. & Wu Q.H. Nikolaidis K. A class boundary preserving algorithm for data condensation. Pattern Recognition*, 44 :704–715, 2011.*

[149] Le nouvel Observateur. Cyber-attaques : Washington soupçonne des pirates iraniens, selon un haut responsable. 12 Octobre :20, 2012.

[150] Anderson J. P. Computer security threat monitoring and surveillance. Technical report, Technical report, James P. Anderson Co., 1980.

[151] James P.Anderson. Computer security threat monitoring and surveillance. Technical report, Technical report, Contract 79F6400, James P.Anderson Co,Box 42, Fort Washington, PA, 19034, February 26, revised April 15 1980.

[152] Odile PAPINI. Logique et sécurité, détection dintrusionlsis. Master's thesis, Université de Toulon et du Var, 2005.

[153] H.S. Lopes Parpinelli, R.S. and A. A. Freitas. Data mining with and ant colony optimization algorithm. IEEE Transactions on Evolutionary Computing*, 6 (4) :321–332, 2002.*

[154] Vern Paxson. Bro :a system for detecting network intruders in real-time. Computer Networks*, 31(23-24) :2435–2463, 1999.*

[155] P.Hart. The condensed nearest neighbor rule. IEEE Transactions on Information Theory*, 14 :515–516, 1968.*

[156] Tadeusz Pietraszek. Alert Classification to reduce false positives in intrusion detection*. PhD thesis, Albert-Ludwigs-University of Freiburg, Germany, 2006.*

[157] Tadeusz Pietraszek. Alert Classification to reduce false positives in intrusion detection*. PhD thesis, Institut fur Informatik, Albert-Ludwigs-Universitat Freiburg, Germany, 2006.*

[158] Philip A Porras and Peter G Neumann. Emerald : Event monitoring enabling responses to anomalous live disturbances. Proceeding of the 20th NAtional Information Systems Security Conference, Baltimore, Maryland,USA*, . :353–365, 7-10 October 1997.*

[159] Philip A Porras and Alfonso Valdes. Live traffic analisys of tcp/ip gateways. Proceeding of the 1998 ISOC Symposium on Network and Distributed System Security,*, San Diego :California, 11-13 March 1998.*

[160] Thomas H. Ptacek and Timothy N. Newsham. Insertion, evasion and denial of service : Eluding network intrusion detection. Technical report, Secure Networks Inc, 1998.

[161] Thomas H. Ptacek and Timothy N. Newsham. Insertion, evasion and denial of service : Eluding network intrusion detection. Technical report, Secure Networks Inc., 1998.

[162] Dawkins R. The Selfish Gene*. Oxford University Press, 1976.*

[163] F.J. Ferri E. Vidal R.A.Mollinedaa. An efficient prototype merging strategy for the condensed 1-nn rule through class-conditional hierarchical clustering. Pattern Recognition*, 35 :2771– 2782, 2002.*

[164] C. R. Reeves and D. R. Bush. Using genetic algorithms for training data selection in RBF networks, in Instance Selection and Construction, pp. 339356. *Norwell, MA : Kluwer, 2001.*

[165] A.Maccabe R.Heady, G.Luger and M.Servilla. The architecture of a network level intrusion detection system. Technical report, University of New Mexico, Departement of Computer Science, August 1990.

[166] Seth Webster Richard Lippmann and Douglas Stetson. The effect of identifying vulnerabilities and patching software on the utility of network intrusion detection. In Recent Advances in Intrusion Detection (RAID2002),Springer-Verlag*, 2516 LNCS :307–326, 2002.*

[167] InfoSec Reading Room. Intrusion detection systems :definition, need and challenges. SANS Institut*, 2001.*

[168] Axelsson S. The base-rate fallacy and its implications for the intrusion detection. In Proceedings of the 6th ACM Conference on Computer and Communications Security, pages, Kent Ridge Digital Labs*, Singapore :17, 1999.*

[169] F. Ghedjati et A. Hamzaoui S. Khalouli. An ant colony system algorithm for the hybrid flow-shop scheduling problem. IGI Global, International Journal of Applied Metaheuristic Computing (IJAMC)*, 2(1) :29–44, 2011.*

[170] Francisco Herrera Salvador García, José Ramón Canob. A memetic algorithm for evolutionary prototype selection : Ascaling up approach. Pattern Recognition*, 41 :2693– 2709, 2008.*

[171] Umesh Shankar and Vern Paxson. Active mapping : Resisting nids evasion without altering traffic. In Proceedings of the 2003 IEEE Symposium on Security and Privacy*, Oakland, CA :4462, 2001.*

[172] Chatterjee S. & Zhang A. Sheikholeslami, G. Wavecluster : A multi-resolution clustering approach for very large spatial databases. In the 24th International Conference of Very Large Data Bases (VLDB). 428-439*, 1998.*

[173] Matzner S. Sinclair C., Pierce L. An application of machine learning to network intrusion detection. In the 5th annual computer security applications conference, Phoenix, USA. 3717.*, 1999.*

[174] Robin Sommer and Vern Paxson. Enhancing byte-level network intrusion detection signatures with context. In Proceedings of the 10th ACM Conference on Computer and Communication Security*, Washington, DC :262–271, 2003.*

[175] R. Crawford M.Dilger J.Frank J.Hoagland K Levitt C.Wee R.Yip S.Stani Ford Chen, S.Cheung and D.Zerkle. Grids-a graph based intrusion detection system for large networks. Proceeding of the 19th National Information Systems Security Conference*, 1996.*

[176] Porrar P. Kahn C. Schnackenberg D. Feiertag R. et al Staniford-Chen S., Tung B. The common intrusion detection framework-data formats. Technical report, Internet draft : draft-staniford-cidf-dataformats-00.txt, 1998.

[177] Daniel M Teal Steven R Snapp, Stephen E.Smaha and Tim Garance. The dids (distributed intrusion detection system) prototype. Proceeding of the Summer USENIX Conference, San Antonio, Texas*, . :227–233, 8-12 June 1992.*

[178] Suguna and Dr. K. Thanushkodi. An improved k-nearest neighbor classification using genetic algorithm. International Journal of Computer Science Issues*, 7 :18–21, 2010.*

[179] S. Sumathi and Surekha P. Computational Intelligence Paradigms : Theory and Applications using MATLAB*. CRC Press, 2010.*

[180] M. Narasimha Murty T. Ravindra Babu. Comparison of genetic algorithm based prototype selection schemes. Pattern Recognition*, 34 :523–525, 2001.*

[181] M. Tavallaee. A detailed analysis of the kdd cup 99 data set. In in Proceedings of the Second IEEE international conference on Computational intelligence for security and de-fense applications, 53-58*, 2009.*

[182] Kenaza Tayeb. Détection d'intrusion coopérative basée sur la fusion de données. Master's thesis, Institut National de formation en Informatique / ALGER, 2006.

[183] Karl Levitt Biswanath Mukherjee Jeff Wood Todd Heberlein, Gihan Dias and David Wolber. A network security monitor. Proceeding of the 1990 IEEE Symposium on Research in Security and Privacy.Soc Press, Los Alamitos*, CA :USA, 1990.*

[184] H S Vaccaro and G E Liepins. Detection of anomalous computer session activity. Proceeding of the 1989 IEEE Symposium on Security and Privacy. Oakland, California*, . :280–289, 1-3 May 1989.*

[185] Ariadna Fuertes Vicente Ceveron. Parallel random search and tabu search for the minimal consistent subset selection problem. Lecture Notes Comp.Sci., vol. 1518*, 1518 :248–259, 1999.*

[186] B. Shi W. Yi and W. Zhang'ou. A fast knn algorithm applied to web text categorization. Journal of The China Society for Scientific and Technical In-formation*, 26(1) :60–64, 2007.*

[187] DYLAN WALSH. Cyberstalkers threaten pipeline security. http ://green.blogs.nytimes.com/2013/01/10/cyberstalkers-threaten-pipeline-security/ ?partner=rss&emc=rss*, 1(47), 2013.*

[188] Yang J. & Muntz R. R. Wang, W. Sting : A statistical information grid approach to spatial data mining. In the 23rd International Conference on Very Large Data Bases, Morgan Kaufmann. 186-195*, 1997.*

[189] Sal Stolfo Wenke Lee and Kui Mok. A data mining framework for building intrusion detection models. In the IEEE Symposium on Security and Privacy, Oakland, CA*, 1999.*

[190] Randall Wilson and Tony R. Martinez. Reduction techniques for instance-based learning algorithms. Machine Learning*, 38-3 :257–286, 2000.*

[191] Ian H. Witten and Eibe Frank. DATA MINING : Practical MAchine Learning Tools and Techniques*. Morgan Kaufmann, 2005.*

[192] Wang X. A fast exact k-nearest neighbors algorithm for high dimensional search using k-means clustering and triangle inequality. In The 2011 International Joint Conference on Neural Networks. pp. 1293-1299*, 2011.*

[193] Ke Li Xiang Cheng, Bing-Xiang Liu and Jun Yan. Intrusion detection system based on knn-mars. World Congress on Software Engineering*, 1 :392–396, 2009.*

[194] Li Y. Optimizing network anomaly detection scheme using instance selection mechanism. Global Telecommunications Conference, *GLOBECOM :17, 2009.*

[195] Wang Y. and Wang Z-O. A fast knn algorithm applied to web text categorization. International Conference on Machine Learning and Cybernetics, *6 :3436–3441, 2007.*

[196] Yiming Yang. An evaluation of statistical approaches to text categorization. Journal of Information Retrieval, *1 :67–88, 1999.*

[197] Chandru Sargor Shyhtsun Felix Wu Y.Frank Jou, Fenming Gong and Cleaveland W Rance. Architecture design of a scalable intrusion detection system for the emerging network infrastructure. Technical report, Department of Computer Science, North Carolina State University, Releigh, N.C, USA, April 1997.

[198] Krasimir G. Iankiev Yingquan Wu and Venu Govindaraju. Improved k-nearest neighbor classification. Pattern Recognition, *35 :2311–2318, 2002.*

[199] W. Yu and W. Zheng'ou. A fast knn algorithm for text categorization. Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, Hong Kong, *. :3436–3441, 2007.*