وزارة التعليم العالي و البحث العلمي

BADJI MOKHTAR-ANNABA UNIVERSITY
UNIVERSITE BADJI MOKHTAR-ANNABA

جامعة باجي مختار– عنابـــة

Faculté des Sciences de L'Ingéniorat

Département d'Informatique

THESE

Présentée en vue de l'obtention
du diplôme de Doctorat en sciences

Approche hybride pour l'apprentissage automatique incrémental

Option : Intelligence Artificielle

par
M^me^ Chefrour Aida

Directeur de Thèse
Mme Labiba SOUICI-MESLATI Prof. Université Badji Mokhtar-Annaba

DEVANT LE JURY

Président :

Mme Halima BAHI Prof. Université Badji Mokhtar-Annaba

Examinateurs :

Mr Imed BOUCHRIKA MCA Université Med Cherif Messaadia- Souk Ahras
Mr Chawki DJEDDI MCA Université Larbi Tébessi- Tébessa
Mr Brahim FAROU MCA Université 8 Mai 1945- Guelma

Année 2019-2020

REMERCIEMENTS

Louanges à Allah, qui m'a donné la force, la patience et la volonté d'arriver au terme de ce travail.

Je tiens à exprimer mes profonds remerciements à ma directrice de thèse **Professeur Souici-Meslati Labiba**, pour son encadrement, ses conseils, ses orientations, ses encouragements et son soutien permanent tant au niveau des connaissances qu'au niveau humain. Je la remercie pour ses efforts considérables et pour le temps qu'elle a consacré à l'encadrement de ce travail de recherche. **Merci beaucoup Pr Souici-Meslati Labiba.**

Je tiens, également, à remercier les membres du jury qui m'ont fait l'honneur de bien vouloir évaluer mon travail, et plus précisément :

Madame **Halima BAHI**, Professeur à l'université Badji Mokhtar -Annaba pour l'honneur qu'elle m'a fait, en acceptant de présider ce jury.

Monsieur **Imed BOUCHRIKA**, Professeur à l'université Mohamed Chérif Messaadia -Souk Ahras-, pour avoir accepté de juger le présent travail.

Monsieur **Chawki DJEDDI**, MCA à l'université Larbi Tébessi -Tébessa-, d'avoir accepté de faire partie de ce jury.

Monsieur **Brahim FAROU**, MCA à l'université 8 Mai 1945 -Guelma-, pour avoir accepté d'accorder de son temps pour examiner ce travail.

Je voudrais aussi profiter de cette occasion pour remercier tous ceux qui ont contribué, de près ou de loin, à l'élaboration de ce travail, en particulier ma mère, mon père, mon mari, mon frère et mes sœurs.

DEDICACES

A mon cher père,

La source et l'origine de tous mes succès, pour son grand amour, son encouragement.

A ma chère mère,

La lumière de ma vie, qui n'a cessé de prier

pour que j'atteigne mes objectifs dans la vie.

A mon mari,

Mon compagnon de route, Pour son affection,

sa patience et son encouragement.

A mon fils Mohamed Wael

A ma fille Nouran

A mon frère

A mes soeurs

Qui sont toujours à mes côtés, prêts à m'aider.

A tous ceux qui me sont chers…

Je dédie les fruits de ce modeste travail

RÉSUMÉ

Le travail présenté dans ce document se situe dans le cadre général de l'apprentissage automatique et, plus précisément, celui de l'apprentissage incrémental qui propose une alternative d'évolutivité et d'adaptation dynamique pour intégrer de nouvelles connaissances, de nouvelles données ou pour restructurer des problèmes déjà partiellement appris. L'objectif principal de l'apprentissage incrémental est de disposer d'un système capable d'apprendre de nouvelles informations (nouvelles données, nouvelles classes...) sans pour autant oublier les connaissances déjà acquises.

Suite à notre étude des concepts et méthodes de l'apprentissage incrémental ainsi que des travaux relatifs à l'apprentissage incrémental supervisé, notamment dans le domaine de la reconnaissance de formes, nous avons constaté le manque de propositions d'hybridation et de combinaison de ces méthodes. Dans ce cadre, nous comparons et combinons deux algorithmes d'apprentissage supervisé le SVM incrémental et le réseau neuronal incrémental Learn++. Le système proposé, que nous avons nommé ISVM-Learn++, a montré des bonnes capacités d'apprentissage incrémental sur plusieurs jeux de données et a permis d'obtenir des résultats satisfaisants.

Après une étude approfondie des méthodes d'apprentissage non supervisé ou clustering, notamment celles proposées dans un cadre incrémental, nous nous sommes intéressés particulièrement au DBSCAN (Density-Based Spatial Clustering of Applications with Noise). Nous proposons AMF-IDBSCAN, une version améliorée de cet algorithme qui construit de manière incrémentale les clusters de formes et de tailles différentes dans de grands ensembles de données et élimine la présence de bruit et de valeurs aberrantes. L'algorithme AMF-IDBSCAN proposé utilise le canopy clustering pour pré-regrouper les ensembles de données afin de réduire le volume de données, applique un DBSCAN incrémental pour regrouper les exemples d'apprentissage et la technique AMF (Adaptive Median Filtering) pour réduire les données aberrantes ou bruits en les remplaçant par les médianes choisies. Les résultats expérimentaux obtenus, sur plusieurs bases de données, montrent que notre algorithme donne de bons résultats par rapport au DBSCAN et à certaines de ses extensions proposées dans la littérature.

Mots clés: Apprentissage incrémental; Apprentissage supervisé ; Clustering; DBSCAN; AMF-IDBSCAN; ISVM-Learn++.

ABSTRACT

Our research work is in the general framework of machine learning and, more specifically, incremental learning which offers an alternative scalability and dynamic adaptation to integrate new knowledge, new data or to restructure already partially learned problems. Therefore, the main objective of incremental learning is to have a system capable of learning new information (new data, new classes ...) without forgetting the knowledge already acquired.

After a detailed study of the concepts and methods of incremental learning as well as work relating to supervised incremental learning, in particular in the field of pattern recognition, we noted the lack of proposals for hybridisation and combination of these methods. In this framework, we compare and combine two supervised learning algorithms for the incremental SVM and the incremental neural network Learn ++. The proposed system, which we named ISVM-Learn ++, has shown good incremental learning capabilities on several datasets and has achieved satisfactory results.

After study on the unsupervised learning methods or clustering, those proposed in an incremental framework, we were particularly interested in DBSCAN (Density-Based Spatial Clustering of Applications with Noise). We offer AMF-IDBSCAN, an improved version of this algorithm that incrementally builds clusters of different shapes and sizes in large data sets and eliminates the presence of noise and outliers. The proposed AMF-IDBSCAN algorithm uses canopy clustering to pre-cluster the data sets to reduce the volume of data, applies an incremental DBSCAN to group the learning examples and the AMF (Adaptive Median Filtering) technique to reduce the outliers or noises by replacing them with the chosen medians. The experimental results obtained, on several databases, show that our algorithm gives good results compared to DBSCAN and to some of its extensions proposed in the literature.

Keywords: Incremental learning; supervised learning; clustering; dbscan; AMF-IDBSCAN; ISVM-Learn ++.

الملخص

العمل المقدم في هاته الوثيقة يندرج في الإطار العام للتعلم الآلي وبالتحديد التعلم الإضافي الذي يوفر قابلية بديلة للتكيف والتكيف الديناميكي لدمج المعرفة أو البيانات الجديدة أو لإعادة هيكلة المشكلات التي سبق تعلمها جزئيًا. لذلك فإن الهدف الرئيسي للتعلم التدريجي هو أن يكون لديك نظام قادر على تعلم معلومات جديدة (بيانات جديدة، فصول جديدة....) دون أن ينسى المعرفة المكتسبة بالفعل.

بعد دراسة مفصلة لمفاهيم وأساليب التعلم التدريجي وكذلك العمل المتعلق به الخاضع للإشراف وخاصة في مجال التعرف على الأنماط لاحظنا عدم وجود مقترحات لتهجين هذه الأساليب ومزيجها. في هذا الإطار نقوم بمقارنة ودمج خوارزميتين خاضعتين للإشراف التدريجي SVM والشبكة العصبية الإضافية..++ Learn. أظهر النظام المقترح الذي أطلقنا عليه اسم++ ISVM-Learn، قدرات تعليمية تدريجية جيدة وحقق نتائج مرضية على العديد من مجموعات البيانات.

بعد دراسة أساليب التعلم غير الخاضعة للرقابة أو ألمجموعات تلك المقترحة في إطار تدريجي انصب اهتمامنا بشكل خاص بالتجميع المكاني القائم على الكثافة للتطبيقات ذات الضوضاء. اقترحنا AMF-IDBSCAN، كنسخة محسنة من هذه الخوارزمية التي تصنع بشكل تدريجي مجموعات من الأشكال والأحجام المختلفة انطلاقا من مجموعات البيانات الكبيرة، وتزيل وجود الضوضاء والقيم المتطرفة. تستخدم خوارزمية AMF-IDBSCAN المقترحة مجموعات المظلات لتجميع مجموعات البيانات مسبقًا لتقليل حجمها وتطبق ايضا خوارزمية DBSCAN التزايدي لتجميع أمثلة التعلم وتقنيةAMF للتصفية الوسيطة التكيفية للحد من القيم المتطرفة أو الضوضاء من خلال استبدالها بالقيم الوسيطة المختارة. أجريت التجارب على عدة قواعد بيانات عامة، وأظهرت نتائج الخوارزمية المقترحة والتي تم تقييمها اعطت نتائج جيدة مقارنةً بـ DBSCAN وبعض امتداداتها المقترحة.

الكلمات المفتاحية: التعلم الآلي، التعلم التدريجي، DBSCAN ،++ISVM-Learn،AMF-IDBSCAN ، التعرف على الشكل.

TABLE DES MATIERES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ALGORITHMES

Introduction générale et problématique

Apprendre, acquérir des connaissances ou des compétences par l'expérience, les études ou l'apprentissage est essentiel à la survie et à la prospérité des hommes, en tant qu'individus, société et espèce. Un bébé apprend à signaler ses besoins immédiats, un jeune enfant apprend à parler, à comprendre une langue, à éviter les dangers, un adulte apprend à conduire...Le grand nombre de situations et les natures intrinsèquement non déterministes suggèrent que l'apprentissage est inévitable. Notre mémoire à court et à long terme est constamment agrégée et réorganisée pour répondre aux différents besoins et critères (temps, risque…) de la multitude de situations nécessitant une prise de décision, intuitive ou profonde. L'environnement est en constante évolution et, dans une certaine mesure, ce qui a bien fonctionné dans le passé (ou dans un cadre quelque peu différent) pourrait ne pas fonctionner aussi bien s'il est confronté à nouveau. En ce sens, pour atteindre son objectif ultime, l'apprentissage doit être robuste, durable et solide.

L'ordinateur occupe une place très importante dans cette société, mais il y a encore tellement de choses qu'un humain réussit mieux, malgré sa capacité limitée de stockage et de calcul. L'un des domaines d'étude les plus intrigants est probablement l'apprentissage. L'une des définitions de l'apprentissage est la suivante [SIM 83]:

"*Learning denotes changes in the system that is adaptive in the sense that they enable the system to do the same task (or tasks drawn from a population of similar tasks) more effectively the next time*."

L'apprentissage artificiel ou automatique (machine learning) est l'application de méthodes de calcul sous-jacentes à la prise de décision basée sur l'expérience. Ce ne sont pas tous les types de problèmes qui nécessitent un apprentissage automatique. Les problèmes de tri dans un environnement statique, par exemple, ont des algorithmes efficaces dédiés, simplement des traductions en langage informatique de recettes qui pourraient être autrement être interprétées et exécutées par des humains (si le temps, par exemple, n'était pas une préoccupation). Cependant, dans de nombreux contextes, ces algorithmes sont difficiles à déterminer. Des relations floues entre la représentation des entrées (par exemple, une image bitmap) et les concepts sous-jacents aux décisions de sortie requises (par exemple, «Chien ou chat?»), l'incertitude inhérente aux données, au modèle réel et à l'environnement en général. Il est extrêmement difficile de concevoir une telle recette comme un algorithme. C'est là que les méthodes d'apprentissage automatique permettent de gagner du temps, visant à imiter les conséquences de la généralisation de l'apprentissage humain (et parfois des structures biologiques sous-jacentes associées), tout en tenant compte des différentes incertitudes sous-jacentes. L'apprentissage automatique a été appliqué avec succès au cours des dernières décennies pour permettre la prise de décision autonome par des robots, le diagnostic médical, l'optimisation de réseau, et de nombreuses autres tâches modernes. La récente prévalence d'Internet et l'ampleur et l'abondance des données associées provenant d'un nombre de capteurs en croissance exponentielle offrent de nouvelles possibilités d'application de l'apprentissage automatique. Certains, tels que ceux utilisés pour les résultats de recherche, la

publicité et la correspondance de contenu, déjà établies comme la pierre angulaire de l'économie de l'Internet, et d'autres, tels que les systèmes de recommandation fondés sur une analyse sémantique, promettent de révolutionner notre expérience de la vie quotidienne grâce à des services connexes de plus en plus personnalisés.

Les objectifs de l'apprentissage automatique sont de fournir une plus grande précision de solution, une plus grande couverture des problèmes, une plus grande économie dans l'obtention de solutions et une plus grande simplicité dans la représentation des connaissances [MCD 89].

Les techniques de classification représentent un sujet très actif dans l'apprentissage automatique. On les retrouve fréquemment dans de nombreux domaines d'application et deviennent l'outil de base de presque toutes les tâches de reconnaissance de formes. Plusieurs approches structurelles et statistiques ont été proposées pour construire des systèmes de classification à partir de données. Traditionnellement, le système de classification est formé à l'aide d'un jeu de données d'apprentissage sous la supervision d'un expert qui contrôle et optimise le processus d'apprentissage. La performance du système est fondamentalement liée à l'algorithme d'apprentissage et au jeu de données d'apprentissage utilisé. Ce dernier contient des échantillons étiquetés des différentes classes qui doivent être reconnus par le système. Dans presque tous les algorithmes d'apprentissage, l'ensemble de données d'apprentissage est visité plusieurs fois afin d'améliorer les performances de classification, qui sont généralement mesurées à l'aide d'un ensemble de données de test séparé. L'expert peut modifier les paramètres et le réglage de l'algorithme d'apprentissage et relancer le processus d'apprentissage jusqu'à obtenir une performance acceptable. Ensuite, le système de classification est remis à l'utilisateur final pour être utilisé dans des contextes applicatifs réels. Le rôle du classifieur est de suggérer une étiquette pour chaque échantillon non étiqueté fourni par l'application. En règle générale, aucun algorithme d'apprentissage n'est disponible du côté utilisateur.

La principale faiblesse du paradigme de conception dans la classification statique est que la base de connaissances est limitée par le jeu de données d'apprentissage disponible du côté expert et ne peut pas être étendue sur la base des données fournies du côté utilisateur. La nature statique du classifieur du côté utilisateur entraîne deux inconvénients principaux.

Le premier inconvénient est que le classifieur ne peut pas être adapté en fonction des données du côté utilisateur qui représentent plus précisément le contexte réel spécifique que l'ensemble de données d'apprentissage du côté expert. Il est généralement très difficile d'obtenir un ensemble exhaustif de données d'apprentissage pouvant couvrir toutes les caractéristiques du système réel dans tous les contextes possibles, en particulier dans des environnements incertains et imprécis, tels que les systèmes de reconnaissance faciale, vocale ou d'écriture manuscrite. Ainsi, il peut être très utile d'enrichir la base de connaissances du classifieur par de nouveaux échantillons des classes apprises obtenues à partir d'un environnement d'exploitation très spécifique.

Le deuxième inconvénient est que le classifieur ne peut pas prendre en compte de nouvelles classes qui n'étaient pas représentées dans le jeu de données d'apprentissage du côté expert. L'ensemble de classes reconnaissables par le classifieur est défini du côté expert et ne peut pas être modifié par l'application. Bien que l'apprentissage devienne impraticable si un très grand ensemble de classes est utilisé, il est toujours difficile, dans de nombreux contextes applicatifs, de prédire toutes les classes possibles requises par l'utilisateur. De plus, les besoins des utilisateurs

peuvent évoluer et le classifieur doit faire face à ces nouveaux besoins et intégrer de nouvelles classes à sa base de connaissances.

Ces inconvénients augmentent le besoin de nouveaux types de systèmes de classification capables d'apprendre, de s'adapter et d'évoluer de manière continue tout au long de la vie. Ces systèmes de classification seront appelés «systèmes évolutifs». Un algorithme d'apprentissage incrémental est utilisé pour apprendre des échantillons de données fournis par l'utilisateur après l'envoi d'un signal de validation ou de correction afin de confirmer ou de modifier l'étiquette proposée par le classifieur. Contrairement à l'apprentissage traditionnel, il n'y a pas de séparation entre la phase d'apprentissage (ou de conception) et la phase d'exploitation dans les systèmes de classification en évolution. L'une des principales caractéristiques des classifieurs en évolution est que les échantillons entrants peuvent introduire de nouvelles classes invisibles apprises par le classifieur sans détruire sa base de connaissances, ni oublier les classes existantes. Dans les systèmes de classification en évolution, le processus d'apprentissage peut soit commencer à partir de l'utilisateur, de sorte qu'aucune phase d'apprentissage initiale n'est requise et que le système n'apprenne que les classes requises par l'utilisateur, ou bien une phase d'apprentissage initiale est effectuée par un expert.

Contexte de recherche et Objectifs

La plupart des méthodes reconnues efficaces dans un contexte d'apprentissage statique ne proposent aucune alternative d'évolutivité et d'adaptation dynamique pour intégrer de nouvelles connaissances, de nouvelles données ou pour restructurer des problèmes déjà partiellement appris. Dans ce cas, une approche classique est de fusionner les anciennes et les nouvelles données et de procéder à un réapprentissage complet du système. Les difficultés d'une telle démarche sont bien entendu la nécessité de disposer des anciennes données d'apprentissage, les temps d'apprentissage prohibitifs et la nécessité de redéfinir les nouveaux paramètres du modèle adopté.

L'adaptabilité ou la capacité d'évolution représente l'une des limitations les plus fondamentales des techniques d'apprentissage qui sont actuellement relativement efficaces dans le cas statique. De ce fait, les méthodes classiques sont souvent inefficaces pour répondre aux nouveaux besoins des applications actuelles où des flux continus de gros volumes de données sont disponibles.

Une alternative à cette problématique est de disposer d'un système d'apprentissage incrémental capable d'apprendre de nouvelles informations (nouvelles données, nouvelles classes...) sans pour autant oublier les connaissances déjà acquises.

L'apprentissage adaptatif ou incrémental concerne des problèmes complexes, dynamiques et évolutifs, avec des données de natures et d'origines différentes, hétérogènes et bruitées. Dans le cadre du traitement des documents écrits, par exemple, il est intéressant d'envisager ce type d'apprentissage aussi bien pour la reconnaissance de mots dans des vocabulaires évolutifs, la recherche d'informations dans des textes ou l'indexation de documents par le contenu.

Le problème de l'apprentissage continu, adaptatif ou incrémental représente l'une des préoccupations majeures de la communauté de l'apprentissage automatique et constitue un champ de recherche ouvert qui a fait l'objet plusieurs types de travaux.

Le travail de recherche envisagé, dans le cadre de cette thèse, se situe dans le cadre général de l'apprentissage automatique. Il se concentre principalement sur les problèmes posés au niveau des systèmes évolutifs qui sont censés avoir la capacité d'intégrer (dans un système déjà entraîné) de nouvelles connaissances: nouvelles données d'apprentissage, nouvelles classes, fusionner/fractionner un ensemble de classes…

Au début de notre travail de recherche, nous nous sommes fixés un ensemble d'objectifs, qui se sont progressivement affinés, que nous avons essayé d'atteindre à travers notre étude bibliographique ainsi que les contributions décrites dans cette thèse :

- Etude des concepts, méthodes et algorithmes d'apprentissage supervisé ou non supervisé, et en particulier, incrémental [CHE 13].
- Réalisation d'une étude bibliographique comparative des méthodes supervisées d'apprentissage incrémental dans le domaine de la reconnaissance de formes [CHE 19a] qui représente le centre d'intérêt de nos travaux antérieurs [CHE 14].
- Proposition d'une approche hybride d'apprentissage incrémental, l'idée d'hybridation s'est progressivement transformée, au fil de notre avancement, en fusion ou combinaison de classifieurs pour l'apprentissage incrémental supervisé.
- Mise en œuvre et expérimentations dans le but de montrer la faisabilité de l'approche proposée et d'en évaluer les performances [CHE 19b].
- Réalisation d'une étude bibliographique des méthodes non supervisées d'apprentissage incrémental dans le but de proposer un algorithme de clustering incrémental.
- Proposition de l'extension un algorithme de clustering incrémental (basés DBSCAN) et son évaluation expérimentale comparative [CHE 19c].

Description du contenu de la thèse

Cette thèse est organisée en quatre chapitres, les deux premiers sont dédiés aux concepts et méthodes relatifs à l'apprentissage automatique classique (statique) et incrémental. Les deux derniers chapitres comprennent une description des travaux entrepris durant notre parcours de thèse.

Chapitre 1. ***Généralités sur l'apprentissage automatique***

Ce chapitre présente les principaux concepts et algorithmes de l'apprentissage automatique. Nous commençons par un bref aperçu sur la notion d'apprentissage, ses catégories, son cycle de vie et ses avantages. Le reste du chapitre est consacré à une description de méthodes d'apprentissage statique ou classique. Nous y présentons des méthodes d'apprentissage supervisé classique telles que les réseaux de neurones supervisés à couches, les K plus proches voisins, les SVM, les arbres de décision ainsi que les méthodes de clustering classiques telles que: DBSCAN, k-means, STING,....., en plus des mesures d'évaluation correspondantes. Nous terminons ce chapitre en citant les principaux problèmes des algorithmes d'apprentissage classique, pour introduire l'intérêt de l'apprentissage incrémental.

Chapitre 2. ***L'apprentissage incrémental : concepts et algorithmes***

Ce chapitre est dédié aux concepts de base de l'apprentissage incrémental (Incremental Learning ou IL), sa définition, son principe, ses caractéristiques, ses catégories ainsi que les principales méthodes et algorithmes d'apprentissage incrémental supervisé et non supervisé.

Chapitre 3. ***Contributions dans l'apprentissage supervisé incrémental en reconnaissance de formes***

Ce chapitre présente une partie de nos contributions dans le cadre de l'apprentissage automatique incrémental, consacrée à l'apprentissage incrémental supervisé. Ce chapitre est scindé en deux parties: la première contient une synthèse d'un ensemble de travaux de l'apprentissage supervisé incrémental dans le domaine de la reconnaissance des formes. La deuxième partie de ce chapitre est consacrée à la présentation, la description et l'évaluation expérimentale de notre proposition de combinaison ISVM-Learn++ de classifieurs supervisés incrémentaux, qui combine (fusionne) le SVM incrémental et le réseau de neurones incrémental Learn++.

Chapitre 4. ***Contributions dans le clustering incrémental***

Ce chapitre présente la deuxième partie de nos contributions dans le cadre de l'apprentissage incrémental, plus précisément, le clustering ou apprentissage non supervisé. En effet, il est dédié à la description détaillée ainsi qu'à l'évaluation de notre proposition d'une version améliorée de l'algorithme de clustering incrémental DBSCAN (Density-Based Spatial Clustering of Applications with Noise) que nous avons intitulée AMF-IDBSCAN et qui est basée sur le canopy clustering et la technique de filtrage médian adaptatif AMF (Adaptative Median Filtering).

A la fin de cette thèse, nous émettons nos conclusions sur les recherches que nous avons entreprises, depuis quelques années, dans le domaine de l'apprentissage automatique incrémental. Nous présentons aussi plusieurs perspectives envisageables pour faire évoluer les propositions que nous avons présentées dans cette thèse.

CHAPITRE 1

GENERALITES SUR L'APPRENTISSAGE AUTOMATIQUE

Ce chapitre présente principalement les algorithmes de l'apprentissage automatique. Nous commençons par un bref aperçu sur la notion d'apprentissage, ses catégories, son cycle de vie et ses avantages. Le reste du chapitre est consacré à une description de méthodes d'apprentissage classiques. Nous y présentons des méthodes d'apprentissage supervisé classiques telles que les réseaux de neurones supervisés à couches, les K plus proches voisins, les SVM, les arbres de décision,.....les méthodes de clustering classiques telles que: DBSCAN, k-means, STING,....., ainsi que les mesures d'évaluation correspondantes. Nous terminons ce chapitre en citant les principaux problèmes des algorithmes d'apprentissage classiques.

1.1. Introduction

Apprendre, acquérir des connaissances ou des compétences par l'expérience, les études ou l'apprentissage, est essentiel à la survie et à la prospérité de l'homme, en tant qu'individus, société et espèce. Un bébé apprend à signaler ses besoins immédiats, un jeune enfant apprend à parler, à comprendre une langue, à éviter les dangers, un adulte apprend à conduire..... Le grand nombre de situations et les natures intrinsèquement non déterministes suggèrent que l'apprentissage est inévitable. Nous apprenons donc, par exemple, par transfert, en utilisant notre mémoire, à court et à long terme qui est constamment agrégée et réorganisée pour répondre aux différents besoins et critères (par exemple: temps, risque) de la multitude de situations nécessitant une prise de décision, intuitive ou profonde. L'environnement en constante évolution, (du fait de nos actions, de notre nature statistique ou autre) doit être considéré comme un apprentissage. Dans une certaine mesure, ce qui a bien fonctionné dans le passé (ou dans un cadre quelque peu différent) pourrait ne pas fonctionner aussi bien s'il est confronté à nouveau. En ce sens, pour atteindre son objectif ultime, l'apprentissage doit être robuste, durable et solide.

L'apprentissage désigne le processus d'augmentation de l'efficacité et la productivité de l'activité intellectuelle et comportementale en se basant sur l'expérience. Il est caractérisé par la quantité d'informations acquise afin d'accomplir une tâche déterminée d'un sujet particulier.

Le concept d'apprentissage peut être décrit de nombreuses manières, notamment l'acquisition de nouvelles connaissances, l'amélioration des connaissances existantes, la représentation des connaissances, l'organisation des connaissances et la découverte de faits par le biais d'expériences.

L'apprentissage humain[1] est un processus adapté à l'individu pour fournir des réponses adéquates à certaines situations. C'est l'acquisition de savoir-faire, c'est-à-dire le processus d'acquisition des

[1] http://campusport.univ-lille2.fr/UV2S/ress_tice/co/chp2_1_notion.html

pratiques, des connaissances, des compétences, d'attitudes ou des valeurs culturelles, par l'observation, l'imitation, l'essai et la répétition,….

Cependant, l'Intelligence Artificielle (IA) a pour but de reconstituer à l'aide des moyens artificiels (par exemple, des programmes informatiques) des raisonnements et des actions intelligentes, et ce, par apprentissage. On parle, d'apprentissage automatique (artificiel) qui utilise la machine pour répondre aux questions et non plus de l'apprentissage humain (naturel).

L'apprentissage automatique cherche à répondre à la question suivante:[MIT 06]

"Comment pouvons-nous construire des systèmes informatiques qui s'améliorent automatiquement avec l'expérience, et quelles sont les règles fondamentales qui régissent tous les processus d'apprentissage?"

Mitchell [MIT 06] a formulé une définition opérationnelle de l'apprentissage automatique:

"Les choses apprennent quand elles changent leur comportement de manière à améliorer leurs performances", on peut penser à apprendre en termes de performance (et non de connaissances), ce qui est plutôt abstrait et mesurable dans le cas de programmes informatiques. Cette performance peut être mesurée soit en termes d'analyse de la complexité des algorithmes, soit en termes de fonctionnalité, le choix dépendant principalement du domaine d'application.

1.2. Concepts de base

1.2.1. Définitions

Dans cette section, nous présentons un bref aperçu sur l'apprentissage automatique afin de faciliter la discussion des algorithmes dans les parties suivantes de cette thèse:

Définition1: [SAM 59]

En 1959, Samuel a défini le Machine Learning comme "*un domaine d'étude qui donne à l'ordinateur la capacité d'apprendre sans être explicitement programmé*".

Définition2: [SIM 83]

L'apprentissage automatique (Machine Learning), est l'une des tâches les plus étudiées de nos jours, Il est une partie très importante de l'Intelligence Artificielle et doit être une des principales caractéristiques des systèmes intelligents. Par apprentissage, nous pouvons exploiter et construire des modèles de la réalité en se basant sur des expériences, soit en créant un modèle complètement, soit en modernisant un modèle partiellement construit.

"*L'apprentissage automatique désigne l'ensemble des changements dans un système qui lui permettant de réaliser une même tâche, ou des tâches similaires, de manière plus efficace ou plus efficiente au cours du temps*"

Définition3: [MIT 83]

En général, une tâche d'apprentissage automatique peut être définie formellement en termes de trois éléments, à savoir: l'expérience d'apprentissage E, les tâches T et l'élément de performance P. Mitchell et al. [MIT 83] définissent une tâche d'apprentissage plus précisément comme suit: Un

programme informatique est réputé apprendre de l'expérience E par rapport à une classe de tâches T et à la mesure de performance P, si les performances aux tâches T, mesurées par P, s'améliorent avec l'expérience E. Cette représentation d'une tâche d'apprentissage automatique définit clairement les exigences. Cela donne une idée sur ce qu'est le problème de l'apprentissage automatique et de ses objectifs d'apprentissage. Il indique également comment ces objectifs peuvent être mesurés afin que l'efficacité de la tâche puisse être décidée.

Définition4: [MIT 97]

L'apprentissage automatique (ou apprentissage artificiel) est, l'étude des algorithmes et des méthodes qui permettent aux programmes de s'améliorer automatiquement par expérience.

Définition5: [WEI 10]

Un système d'apprentissage automatique construit des modèles à partir de données. Le processus de construction de ces modèles est appelé apprentissage ou formation. Les données utilisées lors de la formation s'appellent donc données d'apprentissage.

L'apprentissage automatique (AA) en tant que domaine de recherche s'intéresse donc à l'étude des problèmes à traiter par l'AA, à la mise au point de modèles d'apprentissage automatique et à des méthodes d'apprentissage de ces modèles à partir de données.

Définition6: [MER 12]

La notion d'apprentissage automatique englobe toute méthode permettant de construire un modèle de la réalité à partir de données, soit en améliorant un modèle partiel ou moins général, soit en créant complètement le modèle. Il existe deux tendances principales en apprentissage, celle issue de l'intelligence artificielle et qualifiée de symbolique, et celle issue des statistiques et qualifiée de numérique.

Si on simule l'apprentissage par un système, on trouve:

Les entrées sont : des données, des formes (images, sons, chiffres, lettres,…..); les sorties sont: des classes des formes homogènes; le traitement est : la caractérisation des classes des formes de manière à bien distinguer les familles homogènes de formes.

L'apprentissage automatique consiste à trouver les ressemblances entre les formes d'une même classe et les dissemblances entre les formes des classes différentes.

Actuellement, l'apprentissage automatique permet de réaliser plusieurs tâches différentes telles que [BEL 16]:

- Analyse et traitement automatique des images;
- Interprétation et reconnaissance de formes et de la parole humaine;
- Recherche d'information (moteur internet…);
- Découverte des relations entre les données dans les grandes bases de données;
- Exploitation des connaissances qui ne peuvent pas être explicitées par le cerveau humain;
- Aide des experts humains dans la prise de décisions complexes;

Il y a deux façons d'apprendre: soit le système se modifie lui même pour exploiter ses propres connaissances plus efficacement, soit le système acquiert de nouvelles connaissances grâce à des sources externes [MUS 01].

1.2.2. Catégories d'apprentissage automatique

Les tâches d'apprentissage automatique sont généralement classées en trois grandes catégories, en fonction de la nature du signal d'apprentissage ou du retour d'informations disponible pour un système d'apprentissage [RUS 16]:

- **L'apprentissage supervisé**: consiste à déduire une fonction à partir de données d'apprentissage étiquetées. Les exemples d'apprentissage sont un ensemble de données d'apprentissage, chacun étant une paire composée d'un objet d'entrée (généralement un vecteur) et d'une valeur de sortie souhaitée (également appelé signal de supervision). Un algorithme d'apprentissage supervisé analyse les données d'apprentissage et produit une fonction inférée, qui peut être utilisée pour introduire de nouveaux exemples. Un scénario optimal permet à l'algorithme de déterminer correctement les étiquettes de classes pour des instances invisibles. Cela nécessite que l'algorithme d'apprentissage utilise les données d'apprentissage pour généraliser de manière raisonnable les situations invisibles.
- **L'apprentissage non supervisé:** est l'approche d'apprentissage automatique consistant à inférer une fonction permettant de décrire une structure cachée à partir de données non étiquetées. Comme les exemples donnés à l'apprenant ne sont pas étiquetés, il n'y a pas d'erreur ni de récompense pour évaluer une solution potentielle, ce qui distingue l'apprentissage non supervisé de l'apprentissage supervisé et de l'apprentissage par renforcement.
- **L'apprentissage par renforcement**: est un domaine d'apprentissage inspiré par la psychologie comportementaliste, qui s'intéresse à la façon dont les agents doivent agir dans un environnement afin de maximiser la notion de récompense cumulative. La théorie des jeux est étroitement liée à ce problème, en particulier à la notion de maximiser sa récompense. Cette approche d'apprentissage automatique diffère de l'apprentissage supervisé standard, car les paires d'entrée / sortie correctes ne sont jamais présentées et les actions sous-optimales ne sont pas explicitement corrigées.
- **L'apprentissage semi-supervisé**: se situe entre l'apprentissage supervisé et l'apprentissage non supervisé, en ce sens que l'enseignant donne un signal d'apprentissage incomplet: un ensemble de données d'apprentissage avec une ou plusieurs des sorties cibles manquantes. Le raisonnement qui sous-tend les cas d'apprentissage spécifiques observés à des cas tests spécifiques s'appelle la transduction. Dans l'apprentissage semi-supervisé, il existe un cas de ce principe où l'ensemble des instances de problème est appelé temps d'apprentissage, sauf qu'une partie des cibles est manquante.

1.2.3. Cycle de vie d'une tâche d'apprentissage automatique

Le cycle de vie d'une tâche d'apprentissage automatique suit généralement le processus décrit à la figure 1.1:

1. Choisir un algorithme d'apprentissage;
2. Entraîner l'algorithme en utilisant un ensemble d'instances (appelé ensemble d'apprentissage);

3. Évaluer les performances en exécutant l'algorithme sur un autre ensemble d'instances (appelé ensemble de test).

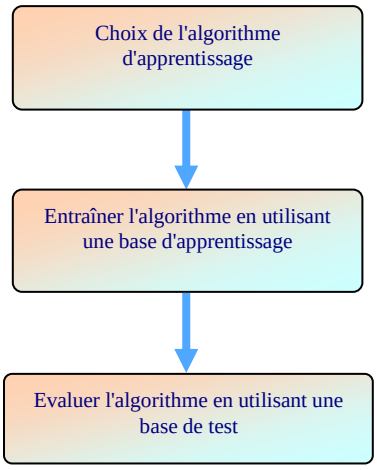

Figure 1.1. Flux d'apprentissage automatique

Selon la nature des connaissances à acquérir, différents types d'algorithmes peuvent être choisis à différents moments. De plus, le type d'entrées et de sorties joue également un rôle déterminant dans le choix d'un algorithme. Celles-ci incluent des algorithmes des règles d'association, des algorithmes basés sur des modèles statistiques, des approches de division, des algorithmes de couverture, des algorithmes de régression, des algorithmes basés sur des modèles linéaires, des algorithmes d'apprentissage par instance et des algorithmes de classification.

1.2.4. Dilemme précision/généralisation

La précision est définie par un écart entre une valeur mesurée ou prédite et une valeur réelle.

La généralisation: est la capacité de reconnaître de nouveaux exemples jamais vus auparavant.

Apprendre avec trop de précision conduit à un "surapprentissage", comme l'apprentissage par cœur, pour lequel des détails insignifiants (dû au bruit) sont appris.

Apprendre avec trop peu de précision, conduit à une sur-généralisation telle que le modèle s'applique même quand l'utilisateur ne le désire pas.

Il existe des mesures de généralisation, et l'utilisateur peut fixer le seuil de généralisation qu'il juge optimal.

1.2.5. Avantages de l'apprentissage automatique

Le domaine de l'apprentissage automatique s'est révélé extrêmement utile dans les domaines liés au génie logiciel [MIT 90]:

1. Problèmes de Data Mining dans lesquels des bases de données volumineuses peuvent contenir de précieuses régularités implicites pouvant être découvertes automatiquement;
2. Domaines dans lesquels les humains n'auraient peut-être pas les connaissances nécessaires pour développer des algorithmes efficaces;

3. Domaines dans lesquels le programme doit s'adapter aux conditions dynamiques.

1.3. Apprentissage supervisé (supervised learning)

L'apprentissage supervisé est une catégorie de techniques d'apprentissage automatique qui permettent à un système d'apprendre et de modéliser des données et des connaissances et de réaliser des tâches à partir d'une base d'exemples d'apprentissage déjà traités et fournis à priori au système.

Chaque exemple de l'ensemble d'apprentissage étant un couple (entrée, sortie). L'apprentissage supervisé contient les méthodes de classification de données (on connaît l'entrée et l'on veut déterminer la sortie) et les méthodes de régression (on connaît la sortie et l'on veut retrouver l'entrée).

L'objectif de la classification supervisée est principalement de définir des règles permettant de classer des individus dans des classes à partir de variables illustratives caractérisant ces individus (en se basant sur des classes prédites) [CHE 13].

L'apprentissage supervisé contient un ensemble des méthodes telles que :

- Les arbres de décision ;
- Les réseaux neuronaux supervisés à couches;
- La méthode des K-plus proches voisins (KPPV).
- Les Séparateurs à Vaste Marge (SVM)....

Exemple :

Soit un exemple de Quinlan [Qui93] de la classification supervisée concernant la possibilité de jouer un match, où il peut s'agir par exemple de déterminer si un nouvel exemple rencontré fait partie de la classe "OUI" ou bien la classe "NON", en se basant sur des caractéristiques de la journée (ensoleillement, température, humidité et vent) et sur les classes déjà connues (exemples d'apprentissage). Le tableau 1.1 présente le problème dans ce cadre, à chaque exemple est associée la classe à laquelle il appartient. L'objectif est alors d'être capable d'estimer la classe la plus appropriée à un nouvel exemple rencontré (l'exemple numéro 5 dans le tableau 1.1):

Journée	**Ensoleillement**	**Température (F)**	**Humidité**	**Vent**	**Classe : jouer ?**
1	Soleil	75	70	Oui	Oui
2	Soleil	80	90	Oui	Non
3	Soleil	85	85	Non	Non
4	Couvert	72	90	Oui	Oui
5 (nouvel exemple)	Soleil	69	70	Non	??

Tableau 1.1. Exemple d'application de la classification supervisée

Dans la sous section suivante, nous détaillerons les principaux éléments relatifs à différents algorithmes d'apprentissage supervisé: définition, processus d'apprentissage, avantages et inconvénients.

1.3.1. Arbres de décision

Les arbres de décision (AD) représentent l'une des méthodes d'apprentissage automatique supervisé symbolique les plus connues dans le domaine de l'intelligence artificielle. Ils sont parmi les techniques de classification les plus anciennes et les plus intuitives. Leur modélisation repose sur une représentation (structuration) graphique d'un ensemble de règles, qui les rendent aisément interprétables, rapides, apparemment simples, et donnant souvent des résultats convenables. C'est pourquoi cette approche a connu un vif succès dans différents domaines comme la fouille de données, ou encore les systèmes d'expertise (médicale, financière,… etc.) ainsi que l'aide au diagnostic médical où le médecin doit pouvoir interpréter les raisons du diagnostic.

Un arbre de décision (decision tree) est un enchaînement (une structure) hiérarchique de règles logiques et en forme d'arbre, sa structure est construite grâce à des méthodes d'apprentissage par induction à partir d'exemples, en cherchant à chaque niveau, l'attribut le plus discriminant pour classifier un exemple. L'arbre ainsi obtenu représente une fonction (qui a une traduction immédiate en terme de règles de décision mutuellement exclusives) qui fait la classification d'exemples, en s'appuyant sur les connaissances induites à partir d'une base d'apprentissage. En raison de cela, ils sont aussi appelés arbres d'induction *(Induction decision tree*s) [NEM 10].

Dans un arbre de décision :

- Une feuille indique une classe.
- Un nœud spécifie un test que doit subir un certain attribut, chaque branche sortant de ce nœud correspond à une valeur possible de l'attribut en question.

L'apprentissage dans les arbres de décision consiste à construire l'arbre à partir de l'ensemble d'apprentissage et à partitionner cet ensemble en sous-ensembles qui deviennent de plus en plus petits jusqu'à contenir à la fin des exemples relatifs à une seule classe, alors les tests sur l'ensemble d'apprentissage forment les branches de l'arbre et chaque classe (sous-ensemble contenant des exemples homogènes) forme la feuille.

Le résultat de l'apprentissage est l'arbre construit, le classement d'un nouvel exemple se fait par le parcours de l'arbre jusqu'à trouver la classe convenable.

La figure (1.2) donne un exemple d'arbre de décision, pour le classement d'un ensemble d'exemples avec un test d'appartenance à une classe. Il existe deux cas : les exemples dit positifs: ceux qui appartiennent à la classe, et les cas négatifs: ceux qui n'y appartiennent pas.

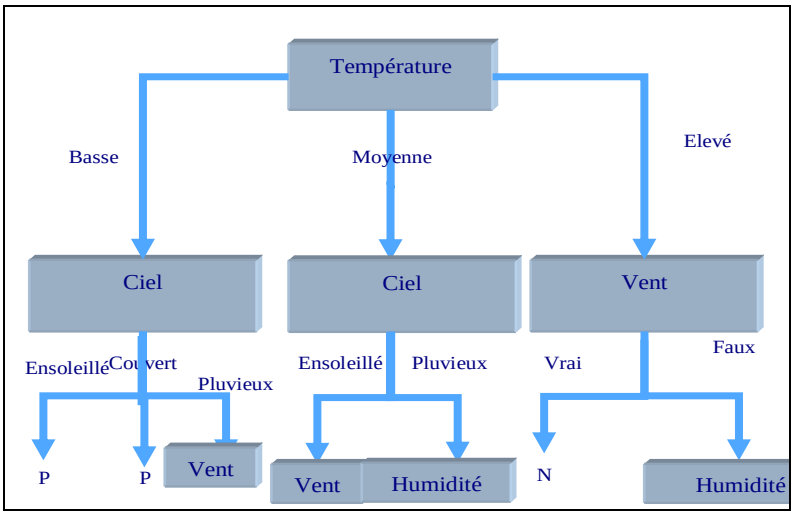

Figure 1.2. Exemple d'arbre de décision simple

Si (température=basse ***et*** le ciel=ensoleillé) ***ou***

(Température=basse ***et*** le ciel=couvert)

Alors **classe=P**

L'apprentissage se fait à partir d'une base d'apprentissage d'exemples, qui possède un certain nombre d'attributs significatifs: température, ciel, vent et humidité. Chaque exemple intègre des valeurs à chaque attribut et il est associé à une classe particulière (méthode d'apprentissage supervisé).

Le principe de construction des arbres est:

On a la base d'apprentissage contenant l'ensemble des attributs non sélectionnés; on eu choisit d'abord un parmi eux et on crée un nœud portant un test sur cet attribut. Si tous les exemples positifs ou négatifs appartenant à une même classe alors on crée une feuille correspondante à cette classe, reliée à l'attribut précédent par un arc étiqueté par la valeur de cet attribut, sinon, si tous les exemples de la classe d'équivalence considérée ne sont pas dans la même classe, on réitère ce processus en enlevant l'attribut précédemment considéré des attributs à sélectionner.

Quinlan [QUI 86] a proposé une technique de choix d'attribut fondée sur la théorie de l'information de Shannon. L'idée consiste à appliquer une méthode qui permet de minimiser l'entropie et de maximiser le gain

On remarque que les arbres de décision possèdent les avantages suivants :

- Leur lisibilité, car ils peuvent être traduits sous forme de règles.
- Leur utilisation est très importante dans le domaine de classification d'images, la reconnaissance d'écriture manuscrite,……
- Leur capacité à trouver les variables discriminantes dans un important volume de données.

Comme toute méthode d'apprentissage, les arbres de décision présentent aussi certains inconvénients

- La difficulté de manipulation de variables quantitatives (valeurs continues). La discrétisation des données reste une solution à considérer, mais elle ne résout pas tous les problèmes de traitement d'informations non symboliques.
- Le choix des attributs à considérer dans les divisions (nœuds de l'arbre) ne garantit pas une solution toujours parfaite. De plus, la méthode la plus utilisée, fondée sur la théorie de l'information de Shannon, impose de fortes contraintes aux méthodes incrémentales (on est obligé de tout reconstruire).

Afin d'éviter ces inconvénients, les chercheurs ont proposé d'utiliser les arbres de décision incrémentaux [NEM 10].

Algorithme 1.1 (Construction d'un arbre de décision)
Arbre←arbre vide; nœud courant← racine **Répéter** Décider si le nœud courant est terminal **Si** le nœud terminal alors lui affecter une classe **Sinon** sélectionner un test et créer autant de nœuds fils qu'il y a de réponse au test Passer au nœud suivant (s'il existe) **Jusqu'à** obtenir un arbre de décision

1.3.2. K plus proches voisins (k-ppv)

La méthode des k plus proches voisins (KPPV en bref, nearest neighbor en anglais*)* est une méthode dédiée à la classification qui peut être étendue à des tâches d'estimation. La méthode KPPV est une méthode de raisonnement à partir de cas. Elle part de l'idée de prendre des décisions en recherchant un ou des cas similaires déjà résolus en mémoire. Contrairement aux autres méthodes de classification (arbres de décision, réseaux de neurones, …), il n'y a pas d'étape d'apprentissage consistant en la construction d'un modèle à partir d'un échantillon d'apprentissage. C'est l'échantillon d'apprentissage, associé à une fonction de distance et d'une fonction de choix de la classe en fonction des classes des voisins les plus proches, qui constitue le modèle.

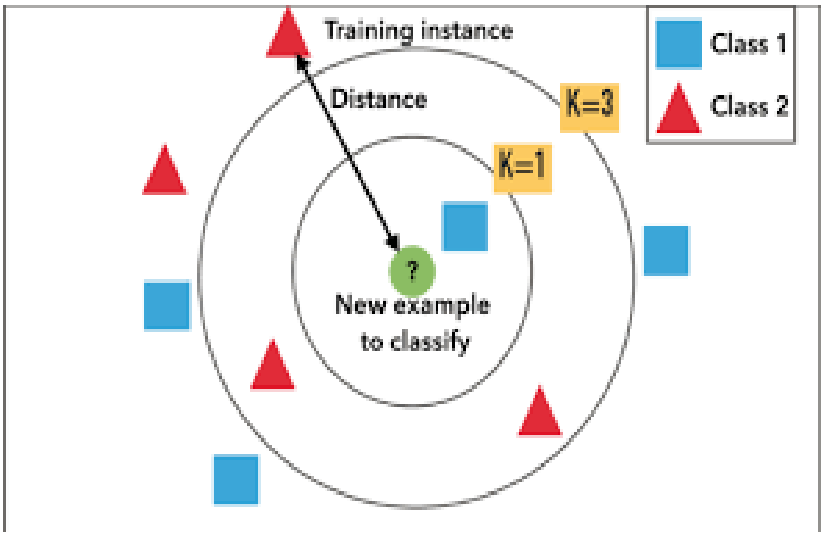

Figure 1.3. Exemple de k-ppv [2]

Algorithme 1.2 (KPPV) [LAD 19]

Entrées :

- Un ensemble de données D.
- Une fonction de définition de distance d
- Un nombre entier $K \geq 1$

Sortie : La prédiction y de la nouvelle observation$X$.

1. Calculer toutes les distances de l'observation X avec les autres observations du jeu de données D à l'aide de la fonction de définition de distance d

2. Retenir les Kobservations du jeu de données D les proches de X

3. Prendre les valeurs de y des K observations retenues :

   - Dans le cas de la régression, calculer la moyenne (ou la médiane) de y retenues
   - Si on effectue une classification, y est la classe majoritaire de ses voisins

4. Retourner la valeur calculée dans l'étape 3 comme étant la valeur qui a été prédite par KNN pour l'observation X.

L'avantage de cette approche est qu'elle ne nécessite pas une phase d'apprentissage, puisque, pour chaque nouvel exemple, la distance euclidienne est calculée entre les exemples de la base d'apprentissage et cet exemple. Alors il est très simple à mettre en place.

L'inconvénient de cette méthode réside au niveau de la performance, elle est lente surtout dans le calcul de distances entre la nouvelle donnée à classer et chaque exemple de la base.

[2] https://informatic-ar.com/k-nearest-neighbor-algorithm/

1.3.3. Les réseaux neuronaux supervisés à couches

Un réseau de neurones (RN) est un ensemble des neurones interconnectés par des poids, qui traitent l'information d'une source externe. Les RN sont des méthodes non symboliques ou adaptatives, ce sont des modèles d'entrées/sortie basés sur un caractère des neurones biologiques. Le but de cette modélisation est de reproduire les capacités de cerveau humain à interpoler ou à classifier et à mémoriser les connaissances de façon expérimentale et de les rendre disponibles pour l'utilisation.

Dans le RN :

- La connaissance est acquise en utilisant le processus d'apprentissage.
- La mémorisation des connaissances se faite par les poids des connections entre les neurones.

Les deux points précédents expliquent la ressemblance entre le RN et le cerveau humain.

La topologie de modèle est définie par la forme des connexions entre les neurones qui composent le réseau. Parmi les topologies les plus connues des réseaux à couches, nous identifions:

Le réseau mono couche: sa structure est que les neurones d'entrée soient entièrement connectés aux neurones de sortie par une couche modifiable de poids.

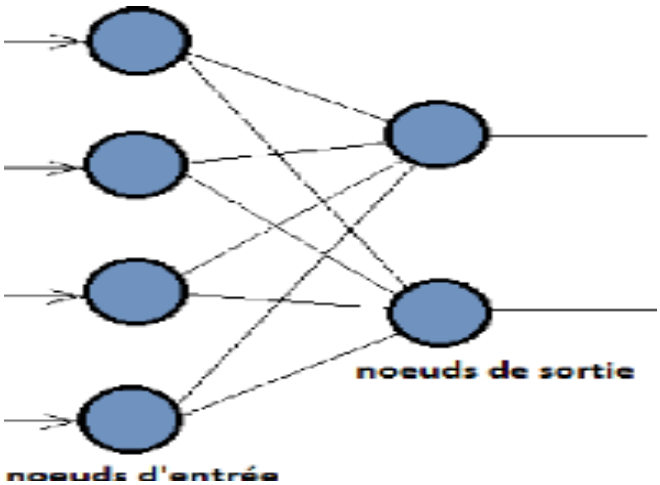

Figure 1.4. Réseau monocouche [BIS 11]

Le perceptron multicouche (PMC): est le réseau le plus utilisé. Il s'agit d'un réseau feedforward (non bouclé) composé de couches successives.

L'idée consiste à regrouper les neurones par couches intermédiaires. Chaque neurone d'une couche est connecté à tous les neurones de la couche suivante. L'ensemble des neurones dont la tâche est de recevoir l'information de l'extérieur est appelé couche d'entrée. Les informations sont transmises ensuite aux neurones intermédiaires, l'ensemble de neurones intermédiaires est appelé couche cachée (il existe une ou plusieurs couche (s) cachée(s)). Ces dernières vont effectuer certains traitements puis envoyer les résultats vers une dernière couche appelée couche de sortie.

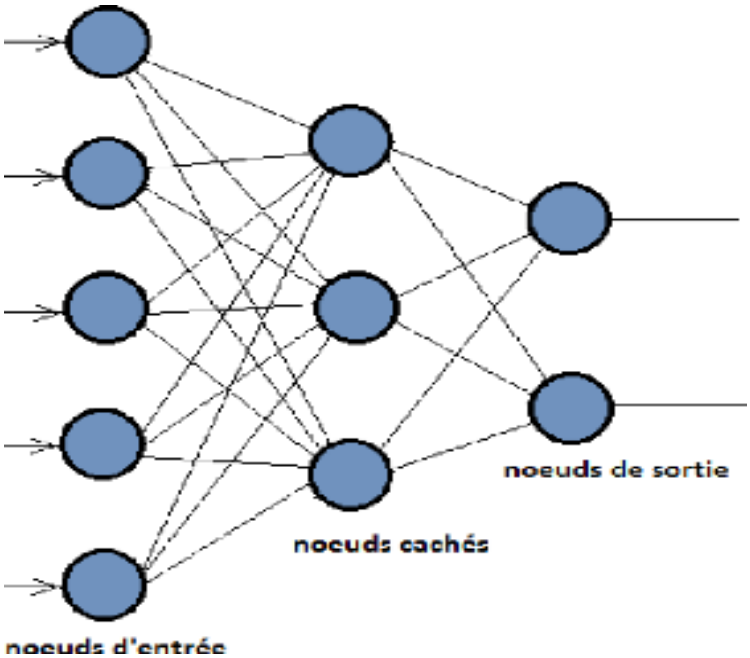

Figure 1.5. Perceptron multicouches [BIS 11]

1.3.4. Séparateurs à large marge (Support Vector Machines ou SVM)

Les SVM sont une méthode basée sur des techniques d'apprentissage supervisé introduite par Cortes et Vapnik [COR 95]. L'idée de cette méthode est de séparer les donnés d'apprentissage, en cherchant le séparateur qui maximise la marge, autrement dit, l'objectif : est de trouver le critère de la maximisation de la marge qui lui donne un excellent pouvoir de généralisation. La motivation principale des SVM est la recherche des limites probabilistes qui permettent de réduire au maximum les erreurs de classification.

Le but de cette méthode d'apprentissage est de modéliser les probabilités d'appartenance d'un objet à une classe, mais cette approche consiste à déterminer les frontières entre les classes (frontière de décision linéaire), autrement dit, à déterminer les surfaces discriminantes. Elle est basée sur l'utilisation de fonctions dites noyaux qui permettent une séparation optimale entre deux classes. Cette méthode repose sur l'existence d'un séparateur linéaire dans l'espace de représentation.

La figure 1.6 montre un exemple de séparation linéaire entre deux classes. Les données de l'ensemble d'apprentissage les plus proches de l'hyperplan qui séparent les deux classes sont appelés : les vecteurs de support. SVM cherche l'hyperplan optimal qui maximise la marge, entre les points de deux classes et cet hyperplan.

Les vecteurs de support désignent les points les plus proches de l'hyperplan optimal H.

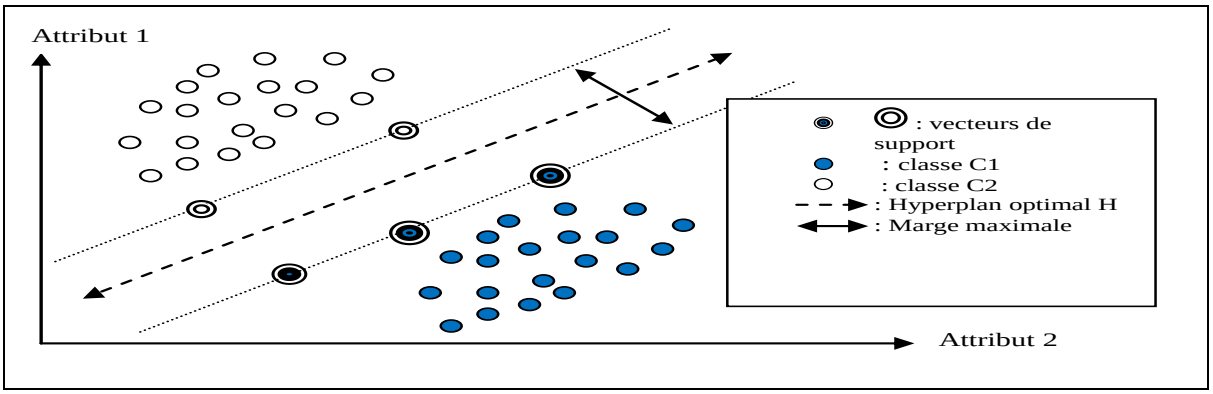

Figure 1.6. Principe de fonctionnement de la méthode SVM dans le cas de classes linéairement séparables [MAR 13]

La méthode SVM détermine les surfaces discriminantes par la transformation de l'espace de représentation initial, dont la séparation est non linéaire, en un espace de dimension plus élevée, dont la séparation devient linéaire; cette transformation est réalisée par la fonction noyau (polynomiale, gaussienne,…).

Pour bien éclaircir la transformation non linéaire en une transformation linéaire, prenons le problème XOR dans un espace de représentation de 2 classes. La séparation de la classe C1 est représentée par les points (0,1) et (1,0), de la classe C2 par les points (0,0) et (1,1), cette séparation est non linéaire comme le montre la figure 1.7 (partie gauche).

On doit effectuer la transformation, on utilise la fonction noyaux de type polynomiale suivante : $(x,y)\rightarrow(x,y,x\times y)$ qui fait passer d'un espace de dimension de 2 à un espace de dimension 3, on obtient un problème de 3 dimensions linéairement séparable, pour la classe C1 : $(0,1)\rightarrow(0,1,0)$, $(1,0)\rightarrow(1,0,0)$, pour la classe C2 : $(0,0)\rightarrow(0,0,0)$,$(1,1)\rightarrow(1,1,1)$, comme le montre la figure 1.7 (partie droite).

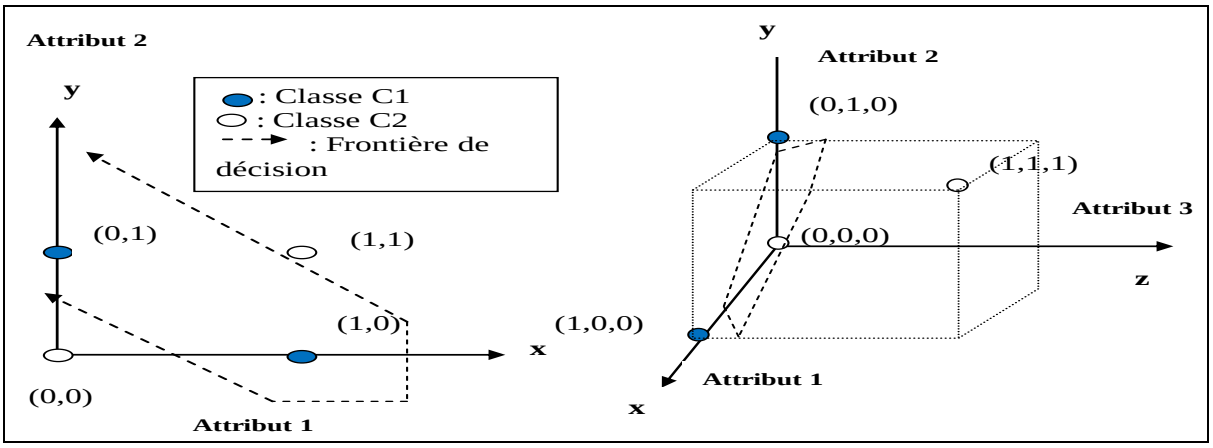

Figure 1.7. Partie gauche : Séparation linéaire pour le problème XOR, Partie droite : Séparation linéaire du problème XOR grâce à la transformation de l'espace de représentation par la fonction à noyau polynomiale [MAR 13]

L'avantage de cette méthode est qu'elle est adaptée pour la classification des données évolutives.

Comme limite de cette méthode, on a l'impossibilité d'intégrer un mécanisme pour la division des classes en plus du temps de calcul qui peut augmenter considérablement quand il y a un grand nombre de points dans l'ensemble d'apprentissage.

Généralement, les SVM sont une méthode d'apprentissage supervisé qui permet de prendre une décision binaire, c'est-à-dire : la classification d'une donnée se fait dans un espace de décision contenant 2 classes au maximum. Lorsque plusieurs classes sont présentes, la classification est traitée par plusieurs SVMs. Il existe deux stratégies [MAR 13] :

- La 1ère stratégie est appelée ***un contre tous*** : on décompose le problème initial en un ensemble de sous problèmes à deux classes, puis on affecte un SVM standard à chacun de ces sous problèmes. Pour cela, on cherche à distinguer les données d'une classe de celles de toutes les autres classes. Pour N classes, on obtient N frontières de séparation.
- La 2ème stratégie est appelée ***un contre un*** : elle consiste à construire autant de SVM qu'il y a des paires de classes, on obtient N(N-1)/2 hyperplans séparateurs. Quand on traite une nouvelle donnée, chaque hyperplan vote pour sa classe préférée, et la classe majoritaire après le vote sera affectée à cette donnée.

1.3.5. Évaluation de l'apprentissage supervisé

L'apprentissage étant un processus inductif, on ne peut pas prouver, au sens de la logique déductive, la validité du résultat d'un apprentissage. On peut néanmoins évaluer de manière approchée la qualité d'une hypothèse. Deux grandes approches sont possibles pour évaluer cette qualité : estimer théoriquement *a priori* un majorant de l'erreur des hypothèses en fonction du nombre d'exemples et des biais utilisés, ou estimer empiriquement *a posteriori* la qualité d'une hypothèse apprise [MUS 01].

1.3.5.1. Evaluation théorique

Déterminer *a priori* la qualité d'un apprentissage est le domaine de la *théorie de l'apprentissage*. Les travaux dans ce domaine visent à déterminer le nombre d'exemples nécessaires pour qu'un apprentissage soit correct. Plus précisément, la théorie PAC (Probably Approximatly Correct) recherche, pour un biais donné, combien d'exemples sont suffisants pour assurer, avec une probabilité donnée, que le résultat d'un apprentissage ait au plus une erreur donnée. Ce nombre d'exemples est appelé la *complexité d'échantillonnage* [MUS 01].

Dans le cadre de l'apprentissage avec un biais de représentation définissant un espace des hypothèses de taille finie, l'idée de départ est la suivante : si l'espace des hypothèses explorées par un algorithme d'apprentissage est petit, il y a peu de chances qu'un algorithme trouve, *par hasard,* dans cet espace, une hypothèse cohérente. De plus, plus les exemples sont nombreux, moins il y a de chance de trouver par hasard une hypothèse cohérente avec ces exemples. Donc, plus l'espace est grand, plus il faut utiliser de nombreux exemples pour s'assurer que l'hypothèse cohérente trouvée ne l'ait pas été du fait du hasard mais du fait que le biais d'apprentissage est adapté au concept à apprendre [SHA 90]. Cette idée peut être étendue au cas des biais de représentation de taille infinie grâce à la notion de dimension de Vapnik et Chervonenkis [VAP 15].

De nombreuses théories ont été développées à partir de ces notions. Elles permettent de définir des bornes sur le nombre suffisant d'exemples pour assurer une certaine qualité au résultat d'un

problème d'apprentissage donné. Cependant, ces bornes sont très pessimistes en pratique, elles sont trop exigeantes. Pour assurer statistiquement une certaine qualité du résultat de l'apprentissage, l'évaluation théorique présuppose en particulier de nombreuses indépendances entre les attributs d'un exemple, et entre chaque attribut et la classe d'un exemple. Or, en pratique, cette indépendance est faible. Les bornes données dans ces théories sont donc basées sur l'analyse du problème dans le pire des cas et tendent à ne pas être utilisables en pratique. La théorie de l'apprentissage permet de montrer que plus l'espace des hypothèses est grand, plus l'apprentissage est difficile, et plus il faut utiliser de nombreux exemples. Elle permet également de guider certaines approches de l'apprentissage. Mais elle ne permet pas de spécifier en pratique un nombre d'exemples minimal pour un problème, ni d'évaluer le résultat d'un l'apprentissage [MUS 01].

1.3.5.2. Evaluation empirique

- ✓ **Erreur réelle, apparente et estimée:** l'évaluation théorique recherche des conditions sur le problème d'apprentissage pour assurer d'obtenir une hypothèse probablement correcte. L'évaluation empirique cherche, quant à elle, à estimer le taux d'erreur d'une hypothèse apprise (h). Ce taux d'erreur, appelé *erreur réelle*, est le pourcentage d'erreur que l'hypothèse h effectue sur l'ensemble E des exemples possibles. Puisque l'on ne connaît pas la classe de tous les exemples de E (d'où le besoin d'apprentissage inductif), cet erreur est, en théorie, inconnue et on cherche à l'estimer. Le taux d'erreur sur les exemples ayant servi à la fabrication de (h), appelé *erreur apparente*, est un estimateur peu fiable et en général très optimiste de l'erreur réelle. Ceci est particulièrement le cas quand l'espace des hypothèses considéré est vaste et que les hypothèses peuvent facilement trop couvrir les exemples d'apprentissage. La validation empirique a pour but d'approcher l'erreur réelle par une *erreur estimée* grâce à diverses méthodes décrites ci-après [MUS 01].
- ✓ **Apprentissage et test:** la manière la plus simple pour estimer l'erreur réelle d'une hypothèse apprise est de l'appliquer sur un jeu d'*exemples tests* totalement indépendants des *exemples d'apprentissage* ayant servi à fabriquer l'hypothèse. L'indépendance des exemples test et des exemples d'apprentissage est un pré-requis nécessaire à la validité de l'approche. Quand les exemples disponibles sont nombreux, cette technique d'estimation de l'erreur est simple et fiable. Mais des difficultés apparaissent quand les exemples disponibles sont peu nombreux. Un premier problème réside dans l'apprentissage: si peu d'exemples sont disponibles il peut être dommageable pour la qualité de l'hypothèse apprise d'ignorer 1/3 des exemples. A l'extrême, si une classe est particulièrement peu représentée elle risque d'être complètement absente des données d'apprentissage, et il devient impossible d'apprendre à classer des exemples dans cette classe. Un deuxième problème réside dans l'évaluation de la qualité de l'hypothèse apprise : si peu d'exemples sont utilisés pour estimer l'erreur, celle-ci peut être éloignée de l'erreur réelle. On se retrouve ici dans un problème d'échantillonnage où la variance de l'erreur estimée est forte lorsque les échantillons (exemples tests) couvrent trop peu l'espace des possibles [MUS 01].
- ✓ **Validation croisée:** la validation croisée est une technique souvent utilisée pour remédier aux problèmes dus au manque de données rencontrés avec la technique "apprentissage et test". Il serait plus précis d'utiliser le terme d'estimateur biaisé. Nous utilisons dans cette partie le terme de fiabilité pour éviter les confusions avec le *biais d'apprentissage.*

- On évalue le taux d'erreur de classification de h_i sur la partie P_i.

- Le taux d'erreur estimé est la moyenne des taux d'erreur des k tests.

La validation croisée est donc une évaluation peu biaisée utile dans le cas où les exemples sont peu nombreux. Mais elle est une évaluation indirecte de l'hypothèse apprise ce qui diminue la confiance que l'on peut lui apporter par rapport à la validation "apprentissage et test" effectuée sur un nombre important d'exemples [MUS 01].

- ✓ **Fiabilité de l'évaluation empirique:** que ce soit en validation croisée ou en évaluation "apprentissage et test", le principe pour assurer une certaine fiabilité de la mesure de l'erreur estimée est de séparer les exemples d'apprentissage et les exemples tests. Il existe néanmoins de nombreux risques d'utiliser plus ou moins directement les exemples tests pendant l'apprentissage, et donc de remettre en cause cette fiabilité.

  Ces risques liés à la fiabilité de l'évaluation sont clairement identifiés par les concepteurs d'algorithmes d'apprentissage. Ils le sont moins par les utilisateurs de ces algorithmes qui effectuent alors des validations peu fiables. En effet, les algorithmes d'apprentissage sont souvent paramétrables, par exemple de manière à fournir des hypothèses plus ou moins détaillées selon les besoins de l'utilisateur. Il est courant qu'un utilisateur utilise alors un algorithme avec différents paramètres pour retenir l'hypothèse qui minimise l'erreur estimée, par exemple par validation croisée. Il n'est alors pas fiable de considérer ensuite que cette erreur minimale est l'estimateur de la qualité de l'apprentissage, puisque les exemples tests ont été utilisés pendant l'apprentissage. Il en est de même si un utilisateur expérimente plusieurs algorithmes différents et utilise le même jeu de test pour choisir l'algorithme et valider ses résultats. Un même risque existe enfin si l'utilisateur expérimente différentes représentations des exemples (en éliminant ou ajoutant des attributs par exemple) pour en retenir la plus efficace.

  De ce fait, en pratique il est très rare que l'erreur d'une hypothèse soit réellement estimée sur des exemples tests entièrement indépendants des exemples ayant servi à construire cette hypothèse. De plus, toutes les techniques d'évaluation empirique de l'apprentissage s'appuient sur l'hypothèse que les exemples sont choisis aléatoirement dans l'espace de tous les exemples possibles, ce qui est rarement le cas en pratique. La fiabilité des estimations est ainsi souvent réduite [MUS 01].

1.3.6. Synthèse des méthodes d'apprentissage supervisé

Plusieurs méthodes sont proposées pour le problème général d'apprentissage supervisé. Elles diffèrent par les mesures de proximité qu'elles utilisent, la nature des données traitées et les objectifs finaux d'apprentissage supervisé. Chacune de ces méthodes possède des points forts et des points faibles. Le tableau 1.2 synthétise ces critères pour les quatre méthodes décrites dans les sections 1.3.1 à 1.3.4

Méthode	Paramètres d'entrée	Type de donnée	Points forts	Points faibles
Arbre de décision	**-Ensemble d'individus X**	**-Symboliques et numériques**	-La simplicité de compréhension et d'interprétation. -Peu de préparation des données. -Le modèle peut gérer à la fois des valeurs numériques et catégorielles. -Possibilité de valider un modèle à l'aide de tests statistiques.	-Peut mener à des arbres de décision très complexes. -Certains concepts sont difficiles.
Réseaux neuronaux supervisés à couches	**-Ensemble d'individus X**	**-Numériques**	-Capacité de représenter n'importe quelle fonction. -Résistance au bruit ou au manque de fiabilité des données. -Comportement acceptable même en cas de faible quantité de données.	-Absence de méthode systématique. -La connaissance acquise par un réseau de neurone est codée par les valeurs des poids qui sont inintelligibles pour l'utilisateur.
K-ppv	**-Nombre de classes K**	**-Numériques**	Simplicité. Bonnes performances en général	-Paramétrage difficile. -Impossibilité d'interprétation d'un classement proposé. -Nécessité de garder sous la main la base de données. -Lenteur en classement -Sensibilité à la dimensionnalité.
SVM	**-Ensemble d'individus X**	**-Numériques**	-Fondements mathématiques solides. - Décision rapide.	- Temps de calcul élevé. - Nécessité d'utiliser l'approche un-contre-un.

Tableau 1.2. Synthèse des méthodes d'apprentissage supervisé

1.4. Apprentissage non supervisé (unsupervised learning, clustering)

Le partitionnement de données (en anglais « clustering »), est un sujet de recherche en apprentissage émanant d'une problématique plus générale, à savoir la classification. C'est une des techniques statistiques largement utilisées dans la fouille de données. Il se place dans un cadre d'apprentissage non supervisé, qui tente d'obtenir des informations sans aucune connaissance préalable, ce qui n'est pas le cas de l'apprentissage supervisé.

Dans cette section, nous présentons, dans la première partie, une introduction au clustering qui est organisée en plusieurs sous sections. Nous présentons dans la première quelques définitions essentielles pour comprendre la notion de clustering, nous rappelons également ses principales étapes.

Dans la deuxième sous section, nous introduisons la notion de similarité (proximité). Cette notion est fondamentale pour la majorité des méthodes de clustering. Nous donnons quelques mesures de similarité utilisées selon le type des variables : numériques et catégorielles.

La troisième sous section est dédiée à l'introduction des principales méthodes de clustering, alors que la section 1.4.4 est consacrée aux mesures d'évaluation du clustering.

1.4.1. Notions de base et étapes du clustering

Clustering, apprentissage non supervisé ou regroupement automatique sont des expressions similaires auxquelles nous devons nous familiariser ; elles convergent toutes dans la même direction et font référence à des méthodes qui se singularisent par : un regroupement des objets d'un ensemble de données, en groupes homogènes inconnus initialement, en fonction de leur similarité.

Plusieurs définitions de clustering ont vu le jour dans ces dernières années, les plus référencées et aussi les plus synthétiques sont présentées dans ce qui suit:

Définition1: **[FOU 11]**

Le clustering aussi connu sous nom (Segmentation) est un regroupement en classes homogènes qui consiste à représenter un nuage des points d'un espace quelconque sous forme d'un ensemble de groupes appelés ***Clusters***.

Un « cluster » est donc une collection d'objets qui sont « similaires »entre eux et qui sont « dissemblables » par rapport aux objets appartenant à d'autres groupes.

Définition2: [DUB 11]

Le clustering a pour but d'organiser une collection de données, d'exemples, de points dans des clusters (des ensembles) de manière à ce que l'affirmation suivante soit vérifiée :

Des points d'un même cluster sont plus similaires/proches, les uns des autres que des points appartenant à des clusters différents.

Cette notion de similarité peut être exprimée de beaucoup de manières différentes, celle-ci dépendant généralement du sujet de l'étude, des hypothèses spécifiques au domaine et de la connaissance initiale du problème. Le clustering est généralement utilisé lorsqu'aucune information n'est disponible en ce qui concerne l'appartenance des données à une classe

prédéfinie. Pour cette raison, le clustering est traditionnellement vu comme une partie de l'apprentissage non-supervisé.

Définition3: (mathématique)

Mathématiquement, on a un ensemble X de N données décrites chacune par leurs P attributs. Le clustering consiste à créer une partition ou une décomposition de cet ensemble en sous parties (clusters) telles que :

- Les données appartenant au même groupe se ressemblent;
- Les données appartenant à deux groupes différents soient peu ressemblantes.

L'objectif principal de clustering est : de maximiser la similarité intra-clusters et de minimiser la similarité inter-clusters.

Exemple :

On répète le même exemple du tableau 1.1 mais, dans le cas de clustering, on remarque qu'il n'y a aucune information a priori de classe associée aux exemples.

Numéro	Ensoleillement	Température (F)	Humidité	Vent
1	Soleil	75	70	Oui
2	Soleil	80	90	Oui
3	Soleil	85	85	Non
4	Couvert	72	90	Oui
5 (nouvel exemple)	Soleil	69	70	Non

Tableau 1.3. Exemple d'application de la classification non supervisée

Un algorithme de clustering dit « efficace » s'il répondre aux critères suivants (exigences) [KOU 11] :

- Evolutivité de données : les méthodes de clustering doivent être évolutives afin de répondre aux BDD (bases de données) de grand volume de données;
- Traitement des différents types d'attributs : l'ensemble de données à traiter peut contenir des types de données complexes aux plusieurs types, en simultané. Cependant, la plupart des algorithmes de clustering traitent aisément le type simple (numérique);
- Découverte des clusters en forme arbitraire : les algorithmes qui s'appuient sur des mesures de distance pour effectuer des regroupements obtiennent, au travers de leur recherche, des amas de points de forme sphérique généralement de taille et de densité relativement similaires. Il est primordial pour un "bon" algorithme d'effectuer une détection sur les résultats obtenus et nous confirmer la découverte d'une forme arbitraire;
- Exigences minimales pour la connaissance du domaine afin de déterminer les paramètres d'entrées : les algorithmes de clustering utilisent des informations sous forme de paramètres d'entrée, et nécessitent l'intervention humaine pour préciser ces paramètres (nombre de clusters par exemple). Nous obtiendrons, au final, des résultats que nous ne pourrons pas qualifier de partiaux et généraux même s'ils reproduisent fidèlement les souhaits formulés en amont par l'usager. De ce fait, il est conseillé de réduire, de façon maximale, l'intervention de l'utilisateur dans le fonctionnement de l'algorithme afin de ne pas ternir la qualité des résultats, et de conserver leur pertinence et leur précision;

- Capacité de composer avec le bruit et les valeurs manquantes, traiter des dimensionnalités élevées et assurer l'intelligibilité et la convivialité.

Le processus de clustering se divise en trois étapes majeures (voir figure 1.8) : (1) la préparation de données (prétraitement), (2) le choix de l'algorithme de clustering et (3) l'exploitation et l'interprétation des résultats de l'algorithme :

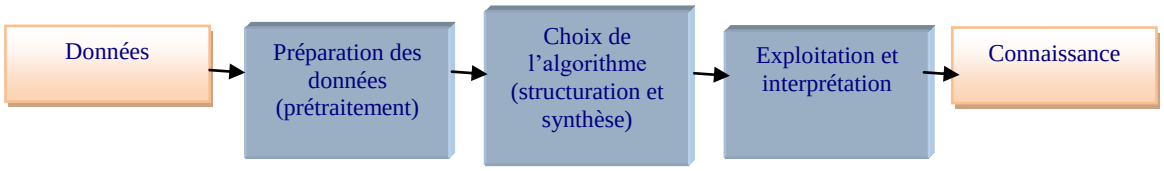

Figure 1.8. Principales étapes du clustering

Nous décrivons ci-après les détails de ces trois étapes [BEN 16]:

1.4.1.1. La préparation des données

Cette étape considère deux éléments : les variables et la mesure de leur proximité.

Les variables :

Les objets sont décrits par des variables, appelées aussi attributs, descripteurs ou traits, elles sont de différentes natures [BAR 96]:

- *Variables quantitatives*: continues (exemple: la taille d'une personne), discrètes (exemple: le nombre de personnes) ou sous forme d'intervalles (exemple: la période de vie d'une personne);
- *Variables qualitatives*: non-ordonnées (exemple: la "couleur" des cheveux) ou ordonnées (exemple: la taille : "petit", "moyen", "grand", etc.) ;
- *Variables structurées*: par exemple la forme d'un objet (polygone, parallélogramme, rectangle, ovale, cercle, etc.).

L'étape de préparation de données consiste à normaliser (sélectionner et/ou pondérer) les variables pour créer des nouvelles variables, afin de distinguer les objets à traiter. En effet, les variables ne sont pas toutes pertinentes : certaines peuvent être redondantes et d'autres non pertinentes pour la tâche ciblée. Ce problème de sélection de variables a été largement étudié en classification supervisée mais reste très ouvert pour l'apprentissage non supervisé.

Le problème de sélection des variables dans le cas non supervisé est dû à la difficulté d'évaluation de la pertinence des variables. Cette difficulté peut être contournée en effectuant une première étape de clustering à partir de l'ensemble des variables, puis de considérer chaque cluster comme une classe et de réitérer ce processus (comme dans l'algorithme des K-moyennes). De fait, cette technique se place parmi des approches dites « enveloppe », qui utilise l'algorithme de clustering pour choisir un sous ensemble de variables réellement discriminantes.

La proximité des variables:

La plupart des algorithmes de clustering utilisent une mesure de proximité entre les objets à traiter. La proximité se présente, dans le clustering, sous forme d'indice, de distance, de mesure de similarité... son choix est déterminant, il a un effet considérable sur le résultat obtenu.

En effet, deux mesures différentes peuvent induire deux schémas de classification différents. Chaque domaine d'application possédant ses propres données, il possède également sa propre notion de "proximité" ; il faut concevoir alors une mesure différente pour chaque domaine d'application, permettant de retranscrire, au mieux, les différences (entre les objets) qui semblent importantes pour un problème donné.

Si on prend par exemple le clustering des données textuelles (mots, documents, etc.), il utilise la proximité basée sur des co-occurrences [BAR 96]. Par contre, dans le cas des données spatiales, ce sont les espaces métriques qui sont étudiés en utilisant des mesures de distances traditionnelles (distance Euclidienne, de Manhattan,…).

1.4.1.2. Le choix de l'algorithme

Le choix de l'algorithme de clustering doit donner lieu à une analyse globale du problème: quelle est la nature de données (qualitative et quantitative)? Quelle est la nature des clusters attendus (nombre, forme, densité, etc.) ? Quelle est le type de schéma attendu (partition, arbre hiérarchique,…) ? Le choix de l'algorithme dépend de la réponse à ces questions.

La taille des données :

La quantité de données à traiter est un facteur de décision très important. Pour traiter les données de très grande taille, les algorithmes linéaires sont très souvent utilisés tels que : l'algorithme des k-moyennes, les nuées dynamiques. En revanche, lorsque l'on souhaite organiser quelques milliers, voire quelques centaines d'objets, on utilise des méthodes plus complexes.

La nature des données

Les algorithmes de clustering s'appuient généralement sur une matrice de similarité ou dissimilarité. Cette matrice est obtenue à partir des descriptions des données, la nature de ces données (quantitatives ou qualitatives) détermine le choix de la mesure de similarité/dissimilarité utilisée.

La forme des clusters

Certaines méthodes de clustering aboutissent à des clusters de formes spécifiques. Par exemple, les méthodes hiérarchiques ou de partitionnement qui consistent à déterminer des centroïdes/ médoïdes aboutissent le plus souvent à des clusters de forme convexe. Par contre, des algorithmes tels que CHAMELEON ou plus simplement l'algorithme ascendant hiérarchique, construisent des clusters de formes variées. La question est alors de savoir s'il est raisonnable de formuler à priori, une hypothèse sur la forme des clusters ? On peut citer quelques facteurs qui entrent dans la décision du choix de l'algorithme : la capacité de l'algorithme à obtenir à la fois des clusters contenant beaucoup d'objets ou peu, voire très peu d'objets, ainsi que la densité qui traduit le degré de proximité entre les objets d'un même cluster ou entre les différents clusters.

Le type de résultats attendus

La sortie d'un algorithme de clustering peut être, par exemple, une partition, une fonction ou encore un dendrogramme (arbre hiérarchique). De même, chaque cluster obtenu peut être défini soit par l'ensemble des objets qui le composent, soit par une description relative aux variables initiales.

Une nouvelle fois, le choix de la méthode de clustering devra être effectué en fonction du type de résultat souhaité et donc de l'exploitation envisagée de ce résultat.

1.4.1.3. L'exploitation des clusters

Les aides à l'interprétation sont des techniques (ou des calculs) qui permettent d'éprouver le bien fondé du résultat obtenu par une quelconque méthode de traitement de données. Elles permettent surtout de rendre raison de la formation des classes qui ne sont que des intermédiaires de calcul et qui nécessitent un contrôle par la suite [DUD 12].

Ces techniques sont un élément essentiel dans le processus du partitionnement de données. Leur dépouillement suggère, par une critique méticuleuse des données, de construire ou de recoder, à nouveau, la technique utilisée pour la réitération de l'analyse.

Cependant, la tentation est grande, pour un non-spécialiste, de considérer comme "acquis" le résultat d'un processus de clustering. Autrement dit, les clusters obtenus ne sont généralement ni remis en cause, ni évalués en termes de disposition relative, dispersion, orientation, séparation, densité ou stabilité [BAR 96].

Pourtant, il est sans aucun doute utile de distinguer les clusters pertinents obtenus, des autres. De même, cette étape d'analyse permet d'envisager le recours à une autre approche de clustering plus adaptée.

Deux situations sont possibles: soit la tâche de clustering s'inscrit dans un traitement global d'apprentissage, soit les clusters générés constituent un résultat final. Dans le premier cas, l'analyse des clusters (mesures statistiques de qualité) peut aider à orienter le traitement suivant. La description des clusters n'est pas nécessaire dans cette situation.

En revanche, dans le cas où le clustering constitue, à lui seul, un processus global de découverte des clusters, l'exploitation de ces derniers pour une application donnée passe par leur description. Lorsque les objets se présentent sous forme d'une matrice de (dis) similarité, il existe peu de méthodes pour décrire les classes (médoïdes, k-objets représentatifs et mesures de cohésion). Lorsque les objets sont décrits par un ensemble de variables, on peut avoir recours à des méthodes de description des clusters (descriptions conceptuelles).

1.4.2. Mesures de similarité

Pour mesurer la ressemblance entre deux objets (points, images, classes, phonème,...), il faut pouvoir mesurer la similarité (ou la dissimilarité) entre eux. C'est une partie importante de la définition d'une méthode de clustering, qui consiste à définir et formaliser une mesure de similarité adaptée aux caractéristiques des données.

Nous allons décrire maintenant les principales mesures utilisées pour évaluer et prouver la similarité entre les objets [KOU 11] :

La distance euclidienne: (aussi appelée la distance à vol d'oiseau) la mesure de la distance la plus courante est la distance euclidienne ou la distance au carré euclidienne.

$$d^2(x_1, x_2) = \sum_i (x_{1i} - x_{2i})^2 = (x_1 - x_2)(x_1 - x_2)' \text{.....(1.1)}$$

La distance de Manhattan: (appelée aussi taxi-distance)

$$d^2(x_1, x_2) = \sum_i |x_{1i} - x_{2i}| \text{.....(1.2)}$$

La distance de Mahalanobis: corrige les données pour les différentes échelles et des corrélations dans les variables. L'angle entre deux vecteurs peut être utilisé comme mesure de distance quand le regroupement des données est de haute dimension.

$$d^2(x_1, x_2) = (x_1 - x_2)C^{-1}(x_1 - x_2)'.....(1.3)$$

$$(C = covariance)$$

La distance de Sebestyen: pour donner un poids différent aux attributs

$$d^2(x_1, x_2) = (x_1 - x_2)W(x_1 - x_2)'.....(1.4)$$

$$(W = matrice\ diagonale\ de\ pondération)$$

La distance de Hamming: mesure le nombre minimum de substitutions nécessaires pour changer un membre dans un autre. Elle permet ainsi, de quantifier la différence entre deux séquences de symboles, généralement utilisée dans le cas des valeurs discrètes (vecteurs)

$$d(a, b) = \sum_{i=0}^{n-1}(a_i \oplus b_i).....(1.5)$$

Exemple : Considérons les suites binaires suivantes :

a= (0 0 0 1 1 1 1) et b= (1 1 0 1 0 1 1) alors d =1+1+0+0+1+0+0

La distance entre a et b est égale à 3 car 3 bits diffèrent.

Distances entre distributions: La similarité entre distributions consiste à déterminer si deux distributions peuvent être issues de la même distribution de probabilités. Le test statistique du X2 chi-square) permet de décider si deux vecteurs $\vec{x}$et $\vec{y}$ sont engendrés par la même distribution. La version symétrique du test est :

$$x^2(\vec{x}, \vec{y}) = \sum_{i=1}^{n} \frac{(x_i, y_i)^2}{x_i + y_i}.....(1.6)$$

Pour les données de grandes dimensions, il y a une distance spécifique très utilisée:

La métrique Minkowski : c'est la mesure la plus populaire, pour les données dimensionnelles,

$$d_p(x_i, x_j) = (\sum_{k=1}^{d} |x_{ik} - x_{jk}|^p)^{\frac{1}{p}} \qquad (1.7)$$

Où d est la dimensionnalité des données. La distance euclidienne est un cas particulier où p = 2, alors que Manhattan p = 1. Néanmoins, il n'existe pas de directives générales théoriques pour la sélection d'une mesure à une application donnée. Une autre question, est de savoir comment mesurer la distance entre 2 classes D (C_1 ; C_2) ? Pour cela, certaines fonctions permettent de mesurer cette distance comme :

Plus proche voisin :

$$\min (d(i, j), i \in C_1, j \in C_2).....(1.8)$$

Diamètre maximum :

$$\max (d(i, j), i \in C_1, j \in C_2).....(1.9)$$

Distance moyenne :

$$\frac{\sum_{i,j} d(i,j)}{n_1 n_2} \qquad (1.10)$$

Distance des centres de gravité :

$$d(\mu_1, \mu_2).....(1.11)$$

Distance de Ward :

$$\sqrt{\frac{n_1 n_2}{n_1+n_2} d(\mu_1, \mu_2)} \text{.....(1.12)}$$

1.4.3. Méthodes de clustering

Dans cette partie, nous présenterons un panorama des méthodes de clustering les plus connues et qui font référence à l'existence de groupes ou classes de données. Les méthodes de clustering d'un ensemble de données peuvent être divisées en quatre familles comme présenté dans la figure 1.9: les approches hiérarchiques, les approches par partitionnement, les approches basées sur la densité et les approches basées sur la grille.

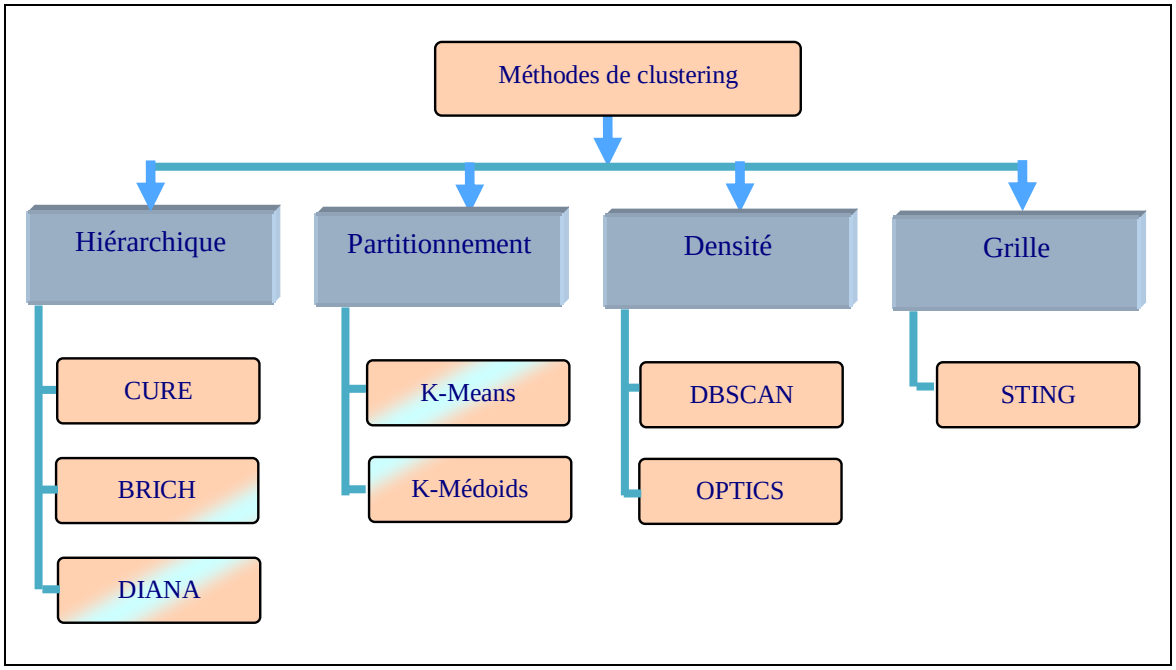

Figure 1.9. Classification des méthodes de clustering

1.4.3.1. Méthodes hiérarchiques

Les méthodes hiérarchiques produisent une séquence de partitions emboîtées d'hétérogénéités croissantes de la plus fine à la plus grossière, conduisent à des résultats sous forme d'arbre hiérarchique indicé (dendrogramme) représentant la proximité entre les données sous forme d'une hiérarchie. Il y a deux genres de méthodes de classification hiérarchiques : la Classification Ascendante Hiérarchique CAH (agglomérative) et la Classification Descendante Hiérarchique CDH (divisive). Les deux modes de construction aboutissent à une classification hiérarchique indicée qui n'est pas forcément la même. Les avantages des méthodes hiérarchiques sont la flexibilité au niveau de granularité, la facilité de manipuler toute forme de similitude ou de distance et l'applicabilité à tout type d'attribut. Par contre, le critère de coupure de l'arbre est souvent vague, la complexité de temps d'au moins $O(n^2)$, où n est le nombre d'objets, la forme de clusters est souvent convexe ne prenant pas en compte les points aberrants.

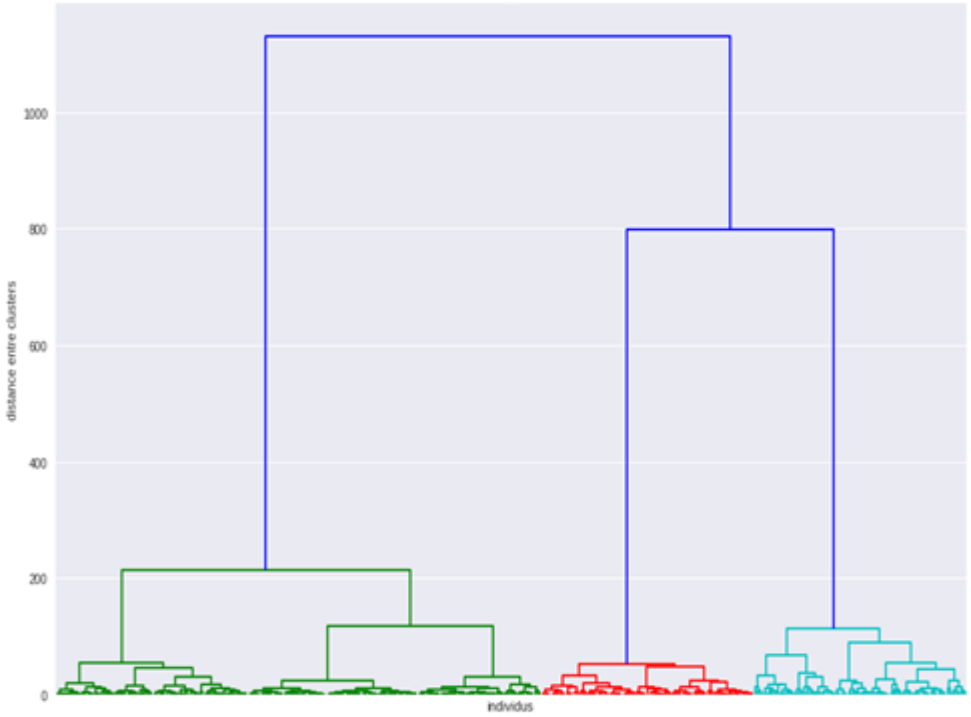

Figure 1.10. Dendrogramme d'un ensemble de données [GUE 19]

Nous allons détailler maintenant les deux sous approches hiérarchiques :

1.4.3.1.1. La Classification Ascendante Hiérarchique (CAH)

On commence en considérant chaque objet comme une classe et on essaye de fusionner deux ou plusieurs classes appropriées selon une mesure de similarité pour former une nouvelle classe. Le processus est réitéré jusqu'à ce que tous les objets se trouvent dans un même cluster. Cette classification génère un arbre que l'on peut couper à différents niveaux pour obtenir un nombre de clusters plus ou moins grand.

L'algorithme de CAH est le suivant :

Algorithme 1.3 (CAH)
1. Les clusters initiaux contiennent les objets eux-mêmes; 2. On calcule la distance entre les clusters; 3. On fusionne les deux clusters les plus proches pour obtenir un seul cluster; 4. Le processus reprend en 2 jusqu'à n'avoir qu'un seul cluster, qui contient tous les objets.

Pour calculer la distance entre les clusters, il existe trois stratégies :

- La stratégie *lien simple* (Single-link), la distance entre deux clusters est représentée par la distance minimale entre toutes les paires de données (paire composée d'un élément de chaque cluster).
- La stratégie *lien complet* (Complete-link), la distance entre deux clusters est représentée par la distance maximale entre toutes les paires de données de deux clusters;
- La stratégie *lien moyen* (Average-link), la distance entre deux clusters est la moyenne des distances entre toutes les paires d'objets de deux clusters.

D'autres méthodes de classification hiérarchique ont été développées, afin d'éviter la majorité des problèmes décrits ci-dessus (section 1.4.3.1), et notamment pour fournir des partitions en clusters de forme et taille arbitraires. Parmi celles-ci citons CURE [GUH 00], BIRCH [ZHA 96] et CHAMELEON [GEO 99].

- L'algorithme CURE (Clustering Using REpresentatives) a été proposé par Gupta et Rastogi [GUH 00]. L'idée est de représenter un cluster par un ensemble de points qui le représentant, il utilise tous les points de données et aussi un centroïde. CURE détermine un nombre constant d'objets représentatifs de ce cluster, il est très efficace pour les grandes bases, c'est une méthode qui maîtrise les outliers et les données aberrantes, elle identifie les clusters ayant une forme arbitraire ainsi que la différence des tailles des clusters.
- L'algorithme BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) a été développé par Zhang et al. [ZHA 96] pour traiter les bases de données de tailles importantes. L'idée principale est de résumer les données avant de procéder au clustering. BIRCH s'articule autour de deux étapes :

  Etape (1) : construction en mémoire d'un arbre de sous-classes et de leur vecteur caractéristiques (CF-tree) résumant les données volumineuses sur disque.

  Etape (2) : application en mémoire d'une technique classique de clustering sur les sous-classes les plus fines de l'arbre déterminé en (1).
- L'algorithme CHAMELEON (Hierarchical Clustering Algorithm Using Dynamic Modeling) a été développé par George [GEO 99]. Cette méthode mesure la similarité entre deux clusters en se basant sur un modèle dynamique, elle vise à créer un ensemble initial important de petits clusters puis les fusionner selon une nouvelle mesure. CHAMELEON procède en deux étapes :
- Une première étape consiste à partitionner l'ensemble des objets en petits clusters homogènes, puis une deuxième phase agglomérative permet d'aboutir à une hiérarchie. Pour cela, CHAMELEON s'appuie sur un formalisme de représentation basé sur les graphes.

1.4.3.1.2. La Classification Descendante Hiérarchique (CDH)

La Classification Descendante Hiérarchique (CDH) considère tous les objets comme un seul cluster puis divise successivement les clusters en clusters plus raffinés. Le processus réitère jusqu'à ce que chaque cluster contient un seul objet ou bien si l'on atteint un nombre de clusters désiré. On a besoin d'un critère pour choisir le groupe à diviser : le plus large/le plus dense. Pour diviser un groupe en 2 ; il y a un grand nombre de possibilités à tester : $(2^{n-1} - 1)$, donc on doit utiliser une heuristique pour réduire le nombre de possibilités. Dans le cas agglomératif, chaque fusion de 2 clusters parmi n, offre seulement $\frac{n(n-1)}{2}$ possibilités.

Pour éviter d'explorer toutes les possibilités de divisions, l'algorithme DIANA (DIvisive ANAlysis) [KAU 09], recherche d'abord l'objet le plus « atypique » (aberrant) du cluster avant de lui agréger éventuellement d'autres objets proches de façon à distinguer deux sous-clusters. Il procède comme suit :

- Il commence avec un seul cluster qui contient l'ensemble des n objets.
- A chaque étape, le cluster avec le plus grand diamètre est sélectionné et doit être divisé (éclaté) en deux clusters. Ici le diamètre d'un cluster est défini comme étant la distance maximale ou de dissimilarité (c'est à dire similarité minimum) parmi tous les objets dans le cluster. Un objet au sein de ce cluster ayant la plus grande dissimilarité moyenne à d'autres objets dans le cluster identifié. Cet objet initie un « nouveau cluster ». Un objet

au sein de ce cluster est réaffecté au nouveau cluster s'il est plus proche du nouveau cluster que du vieux cluster. Par conséquent, à la fin de l'étape, le cluster est divisé en deux nouveaux clusters.

- L'étape ci-dessus est répétée jusqu'à ce que n clusters soient formés.

1.4.3.2. Méthodes par partitionnement

Les méthodes par partitionnement cherchent la meilleure partition en k clusters disjoints de données, le nombre de clusters k étant fixé a priori. Les approches par partitionnement utilisent un processus itératif fonction du nombre k qui consiste à affecter chaque objet au cluster le plus proche au sens d'une distance ou d'un indice de similarité en optimisant une certaine fonction objective traduisant le fait que les objets doivent être similaires au sein d'un même cluster, et dissimilaires d'un cluster à un autre. Les clusters de la partition finale, pris deux à deux, sont d'intersection vide et chacun est représenté par un noyau (un ou des objets de la population, ou un point de l'espace). La plupart des approches supposent un nombre prédéfini des classes.

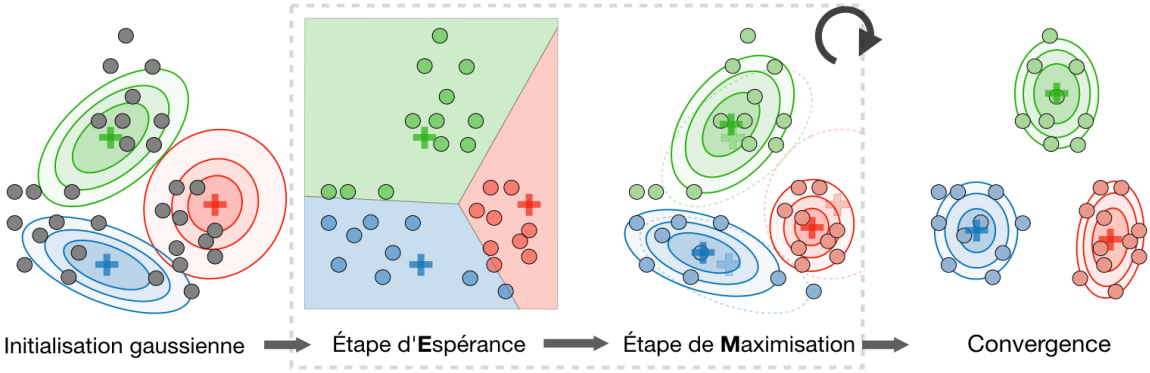

Figure 1.11. Clustering par partitionnement [GUE 19]

Les algorithmes de partitionnement sont divisés en trois grandes sous familles: les méthodes des k-moyennes (k-means), les méthodes des k-médianes (k-medoids) et les méthodes des nuées dynamiques, selon la définition des représentants des classes.

1.4.3.2.1. Les méthodes des K-moyennes

L'algorithme des k-moyennes (k-means) des centres mobiles [TJH 09] est sans aucun doute la méthode de partitionnement la plus connue et la plus utilisée dans divers domaines d'application. Il est utilisé pour regrouper les éléments d'un ensemble de données en k clusters autour d'un centre de gravité (centroïde). En général on ne connait pas le nombre de classes que contient l'ensemble de données.

Le principe est le suivant :

Algorithme 1.4 (K-moyennes)
A l'étape 1: on choisit k individus comme centres initiaux des classes (on tire au sort, ou l'on prend les k premiers, ou l'on prend 1 sur n/k).
A l'étape 2: on calcule les distances entre chaque individu et chaque centre c_i, de l'étape précédente, et on affecte chaque individu au centre le plus proche, ce qui définit k classes.
A l'étape 3: on remplace les k centres c_i par les barycentres des k classes définit à l'étape 2

(ces barycentres ne sont pas forcément des individus de la population).

A l'étape 4: on regarde si les centres sont restés suffisamment stables (en comparant leur déplacement aux distances entre centres initiaux) ou si un nombre fixé d'itérations a été atteint:

Si oui, on arrête (en général, après au moins une dizaine d'itérations).

Sinon, on revient à l'étape2.

L'avantage de l'algorithme k-means réside dans sa complexité linéaire $O(nkt)$; (n: nombre de données, k: nombre de classes, t: nombre de cycles d'exécutions). Mais il est sensible au bruit et requière le nombre de clusters désirés comme paramètre ce qui risque de pâtir la qualité d'une classification si le nombre de clusters fixé d'avance ne correspond pas à la configuration véritable du nuage des objets. En plus K-means ne peut pas classifier les données non-numériques pour lesquelles il est impossible de calculer la moyenne pour générer les centres des clusters.

1.4.3.2.2. Les méthodes des K-médianes

Les k-médianes présentent l'avantage d'être plus robuste aux outliers que les k-means. Cette robustesse vient du principe de l'algorithme : un médiode est le représentant d'un cluster, choisi comme son objet le plus central, ce dont on s'assure en permutant systématiquement un représentant et un autre objet de la population choisis au hasard, pour voir si la qualité de la classification croit, c'est-à-dire si la somme des distances de tous les objets à leurs représentants décroit. L'algorithme s'arrête lorsque plus aucune permutation n'améliore la qualité. Il y'a plusieurs versions des méthodes de k-médianes: PAM [KAU 09], CLARA [KAU 09], CLARANS [HAN 01 et NG 02]. Les k-médianes sont moins sensibles aux points bruits que le k-means et peuvent manipulés des données aussi bien numériques que catégorielles. Mais comme K-means, ils nécessitent la fixation du nombre de classes mais ils ont une complexité quadratique ($O(k(n-k)^2)$).

L'algorithme CLARANS a été développé par Ng et al. [NG 02], son idée est, d'utiliser l'échantillonnage aléatoire. Son algorithme consiste en une recherche stochastique, et la représentation des clusters choisis est un ensemble de K médoïdes.

L'algorithme de k-médianes est le suivant :

Algorithme 1.5 (K-médianes) [BEN 14]

Choisir arbitrairement k médoïdes

Répéter {Affecter chaque objet restant au médoïde le plus proche,

Choisir aléatoirement un non médoïde O_r;

Pour chaque médoïde O_j

Calculer le coût TC du remplacement de O_j par O_r;

Si $TC < 0$ alors remplacer O_j par O_r et

Calculer les nouvelles classes

} jusqu'à ce qu'il n'y ait plus de changement.

Les inconvénients de cette méthode sont: choix de K, sensible aux exceptions et aux données bruitées, difficulté de définition de la moyenne des objets des clusters.

1.4.3.3. Les nuées dynamiques

La méthode des nuées dynamiques développée par Diday [DID 71] se distingue principalement des approches précédentes par le mode de représentation des clusters appelé aussi noyau. Ce dernier peut être son centre de gravité (dans ce cas nous retrouvons l'approche des k-moyennes), un ensemble d'objets (l'approche des k-médianes avec un seul individu), une distance, une loi de probabilité, etc.

L'algorithme des nuées dynamiques cherche à optimiser un critère objectif mesurant l'adéquation entre une partition et un mode de représentation des classes de cette partition. En pratique, l'algorithme converge lorsque ce critère à optimiser cesse de décroitre de façon sensible, ou lorsqu'un nombre fixé d'itérations est atteint. Afin de minimiser ce critère, l'algorithme des nuées dynamiques utilise principalement une étape de représentation suivie d'une étape d'affectation de façon itérative jusqu'à la convergence qui donne une solution localement optimale au problème posé. La méthode des nuées dynamiques comme les approches précédentes de classification automatique par partitionnement fournit une solution dépendante de la configuration initiale qui est généralement faite par tirage au hasard.

1.4.3.4. Les approches par densité

Dans ces approches, les clusters sont considérés comme des régions en haute densité, ils sont séparés par des régions de faible densité. La densité est représentée par le nombre d'objets dans le voisinage. L'idée de ces méthodes est de définir un cluster étant un ensemble d'objets dense selon un critère de voisinage et de connectivité. Ces approches sont fondées sur les concepts de densité, noyau, accessibilité et connectivité.

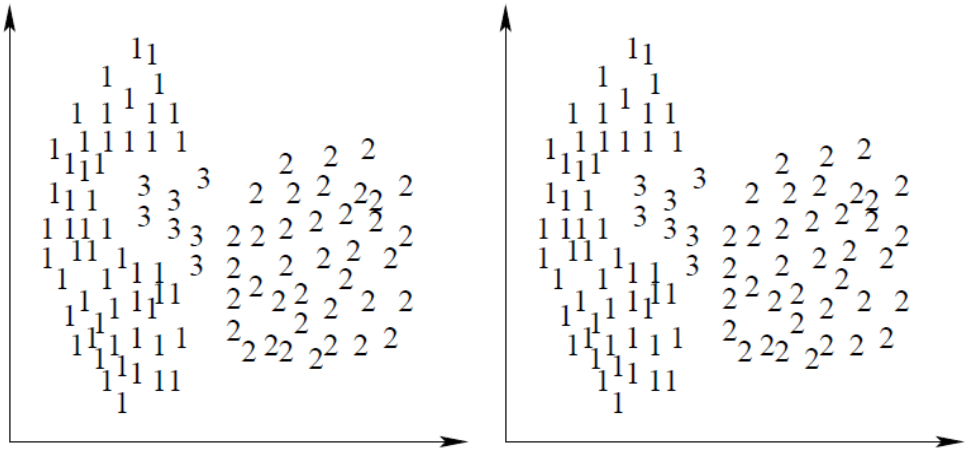

Figure 1.12. Clustering par densité [GUE 19]

Nous allons présenter dans ce qui suit les deux approches par densité :

Les algorithmes les plus connus de ce type sont DBSCAN (basé sur la connectivité de densité) et OPTICS (fonction de densité).

DBSCAN (Density Based Spatial Clustering of Applications with Noise) est un algorithme, très populaire en classification utilise itérativement deux étapes pour découvrir les groupes:

1. Choisit aléatoirement un point dense;

2. Former un groupe à partir de tous les points accessibles depuis ce point selon un certain seuil de densité.

Pour ce faire, deux paramètres doivent êtres définis:

1. Le paramètre de voisinage (ε ou eps) qui définit la distance maximale du groupe;
2. Le seuil de densité ($MinPts$) qui définit le nombre minimal de points dans un voisinage.

Un point qui a au moins $MinPts$ points dans son voisinage de rayon ε est marqué comme un **core point** (point noyau ou central). On a appelle p un **border point** (point bordure), si le nombre de point dans son voisinage est inférieur à $MinPts$. *ε-voisinage* de p: les points qui sont dans un voisinage de ε

A partir de ces deux paramètres, l'algorithme manipule les notions suivantes:

- Un point q est ***directement densité-accessible*** depuis un point p si q est dans l'ε-voisinage de p et p est un point central.
- Un point p est ***densité-accessible*** depuis un point q s'il existe une chaîne $p_1,...,p_n$, avec $p_1=q$ et $p_n=p$, tel que p_{i+1} est directement accessible depuis p_i pour tout i dans $[2,n]$.
- Un point p est ***densité-connecté*** à q s'il existe un point o qui soit densité-accessible à p et q.
- Un ***cluster*** C est un ensemble non vide de points D satisfaisant les conditions suivantes:
    1. *Maximalité* : pour tous points p et q, si p est dans C et si q est densité-accessible depuis p, alors q est aussi dans C
    2. *Connectivité* : pour tous points p et q dans C, p et q sont densité-connectés.
- ***Bruit***: tous les points qui ne sont pas directement densité-accessible depuis au moins un point central.

La figure 1.13 est une illustration de l'algorithme DBSCAN:

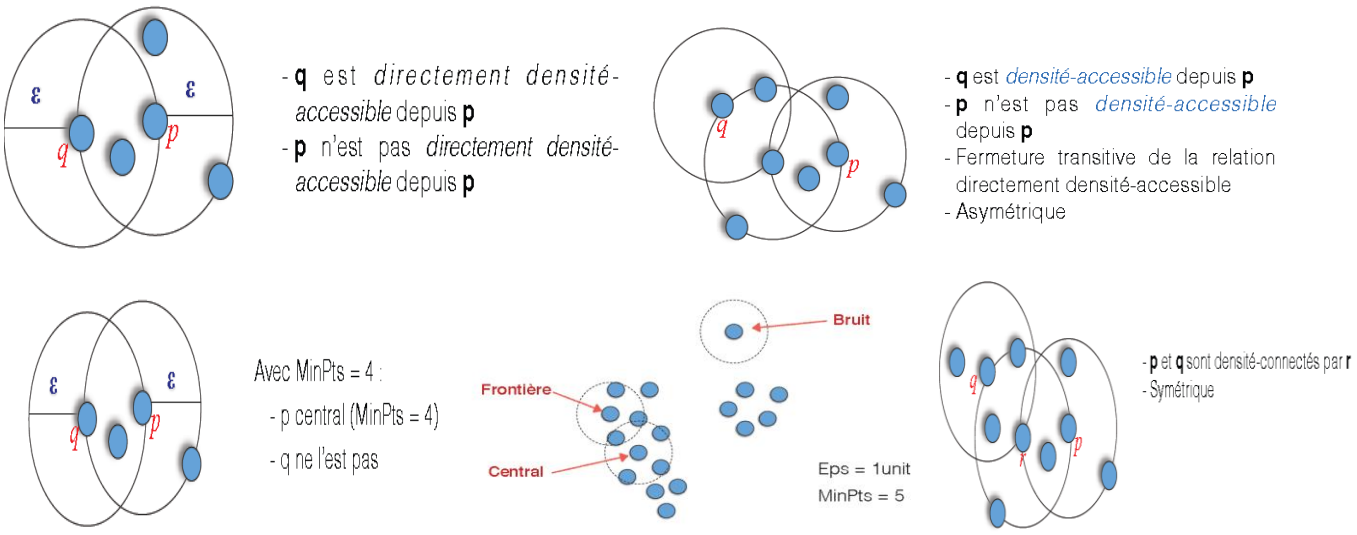

Figure 1.13. Illustration de DBSCAN[3]

La démarche générale se résume comme suit:

[3] https://medium.com/@bluedme_tech/les-techniques-de-d%C3%A9tection-danomalies-183ecd4449ff

Algorithme 1.6 (DBSCAN)
• Choisir un point p • Retrouver tous les points densité-accessibles depuis **p** par rapport aux paramètres utilisateurs ε et $MinPts$ • Si p est central, un cluster est créé avec tous ces points • Si p est bordure, aucun point n'est densité-accessible depuis lui et DBSCAN passe à un autre point non traité de la base de données • On continue le processus jusqu'à ce que tous les points aient été traités

DBSCAN a l'avantage de trouver lui-même une évaluation du nombre de clusters, et celles-ci peuvent avoir des formes arbitraires. Il permet également de gérer tout type de données et de bien tenir compte des données aberrantes (bruites), qui ne sont pas affectées aux clusters identifiés. Cependant, l'algorithme requiert l'entrée des paramètres eps et $MinPts$, et l'expérience montre que les résultats obtenus sont très sensibles aux choix de ces paramètres.

OPTICS (Ordering Points To Identify Clustering Structure) est un algorithme basé sur la densité proposé par Ankerst et al. en 1999 [ANK 99]. OPTICS ne produit pas explicitement un clustering de la base de données, mais il crée un ordre augmenté de la base de données représentant sa structure de clustering basée sur la densité. Cet ordre basé sur la densité ordonné peut être utilisé pour extraire des informations de base de clustering (tels que les centres des clusters ou les clusters de forme arbitraires) ainsi que de fournir la structure du clustering.

Par rapport à l'algorithme DBSCAN, les objets doivent être traités dans un ordre précis. Cet ordre sélectionne un objet qui est accessible par densité par rapport à la plus faible valeur de eps ce qui permet au cluster ayant une densité élevée (petite valeur de eps) d'être déterminer en premier.

Sur la base de cette idée, deux valeurs doivent être stockées pour chaque objet distance-noyau et distance- accessibilité:

- La distance-noyau d'un objet p est la valeur la plus petite eps qui rend $\{p\}$ un point noyau. Si p n'est pas un point-noyau, la distance-noyau de p est indéfini.
- La distance-accessibilité d'un objet q par rapport à un autre objet p est le maximum entre la distance-noyau de p et de la distance euclidienne entre p et q. Si p n'est pas un point-noyau, la distance-accessibilité entre p et q est indéfini [HAN 11].

Son avantage par rapport aux algorithmes de classification proposés dans la littérature, est qu'il ne se limite pas à un paramétrage global. En effet, le cluster ordonné contient des informations qui sont équivalentes au clustering basé sur la densité pour différents paramètres. Donc, il constitue une base polyvalente à la fois pour un clustering automatique et interactif. Par contre, il est impossible d'appliquer OPTICS dans sa forme actuelle à une base de données contenant plusieurs millions d'objets de grande dimension parce que ce dernier ne supporte pas des structures d'indexation pour soutenir efficacement les requêtes de plages hyper-sphère. La complexité d'OPTICS est de l'ordre de $O(n \log n)$.

1.4.3.5. Les approches par grille

L'idée de ces méthodes est qu'on divise l'espace de données en cellules. Donc ce type d'algorithme est conçu pour des données spatiales. Une cellule peut être un cube, une région, un hyper rectangle. En fait, elle est un produit cartésien de sous intervalles d'attributs de données. Avec une telle représentation des données, au lieu de faire la classification dans l'espace de données, on la fait dans l'espace spatial en utilisant des informations statistiques des points dans la cellule. Les méthodes de ce type sont hiérarchiques ou de partitionnement. Les algorithmes les plus connus sont STING, CLIQUE, WaveCluster.

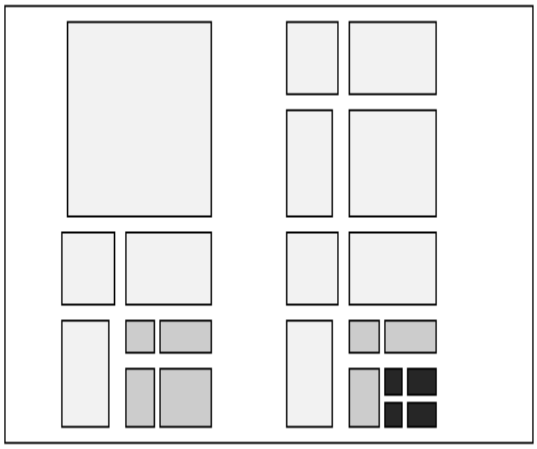

Figure 1.14. Clustering par grille [GUE 19]

WaveCluster (WaveletCBased Clustering) [SHE 98], est un algorithme est basé sur une méthode de transformation des signaux appelée "ondelettes". L'algorithme considère l'espace des attributs comme un espace d-dimensionnel et procède à un découpage de cet espace tel que chaque dimension i est découpée en m_i intervalles de façon à obtenir une grille de $\prod i\, m_i$ cellules où chaque objet est affecté à une cellule. La répartition des objets sur les cellules constitue un signal d-dimensionnel auquel est appliquée la méthode des ondelettes permettant ainsi de former les clusters.

Algorithme 1.7 (WaveCluster)
Entrée: X : Un ensemble de données multidimensionnelles.
Sortie: Un ensemble de clusters
Début
1. Découper l'espace des attributs en cellules.
2. Affecter chaque objet à une cellule.
3. Appliquer la transformation par ondelettes.
4. Rechercher les composants connectés (clusters)
5. Affecter les objets aux clusters.
Fin

Les algorithmes de clustering basés sur les grilles dépendent fortement de la taille de la grille. En effet, WaveCluster est de complexité $\boldsymbol{O(n)}$ par rapport au nombre d'objets dans le cas

bidimensionnel mais cette complexité augmente de façon exponentielle avec le nombre d'attributs.

1.4.4. Evaluation des algorithmes de clustering

L'objectif principal de la validation de clusters est d'évaluer le résultat de clustering afin de trouver le meilleur partitionnement du jeu de données. Historiquement, une série de mesures ont été proposées pour évaluer les résultats du clustering en fonction de différents critères [DAL 09].

En particulier, toute mesure d'évaluation ne doit pas prendre en compte les valeurs absolues des étiquettes de cluster mais plutôt si, dans le cluster, les séparations de données similaires sont bien définies par rapport à un ensemble de classes prises comme référence ou satisfaire certaines hypothèses de telle manière que les membres appartenant à la même classe se ressemblent davantage que les membres de classes différentes, selon une métrique de similarité.

1.4.4.1. Indices de validation internes

Appelée aussi évaluation non supervisée. Dans ce type d'évaluation, l'information interne au clustering, telle que la matrice de similarité, est utilisée comme référence. Elle mesure le degré de correspondance entre le résultat du clustering et l'information contenue dans l'ensemble de données. Le meilleur clustering est celui qui conserve le maximum d'information relativement à l'information contenue dans la matrice de similarité [BAR 15]. Ce type d'approche permet aussi d'évaluer la compacité et la séparabilité des clusters. De nombreuses mesures existent, nous présentons dans ce qui suit les plus connues.

- ***L'indice de Dunn***

L'indice de Dunn [DUN 73] est basé sur l'identification de clusters denses et bien séparés. Il est défini par le rapport entre la plus petite dissimilarité interclasse d_{min} (entre deux individus de deux classes différentes) et la plus grande dissimilarité intra-classe s_{max} (entre deux individus de la même classe).

$$\mathrm{Dunn(P)} = \frac{\mathrm{d_{min}}}{\mathrm{s_{max}}} \text{.....(1.13)}$$

Alors le but principal de cet indice est de maximiser la dissimilarité interclasse et de minimiser la dissimilarité intra-classe. L'objectif est donc de maximiser l'indice de Dunn.

- ***L'indice de Dunn généralisé***

La conséquence de non robustesse de l'indice de Dunn a introduit un nouvel indice qui s'appelle: indice de *Dunn généralisé* [BEZ 92]. En effet il dépend uniquement d'un nombre réduit d'individus de la population, et des liens établis entre eux.

Or, il est sensible à tout changement qui intervient dans la structure des clusters ainsi qu'aux objets aberrants. Les modifications apportées à cet indice interviennent dans le calcul de la dissimilarité *interclasse* et *intra-classe*.

L'indice de *Dunn généralisé* est plus approprié pour l'évaluation de la qualité d'une partition donnée, car il fournit un bon compromis entre la *maximisation* de la *dissimilarité interclasse* et la *minimisation* de la *dissimilarité intra-classe* de la partition.

$$Dunn\_gen(P) = \frac{\min_i\{\min_{j\neq i}\{d_a(C_i,C_j)\}\}}{\max_h s_a(C_h)} \text{.....(1.14)}$$

La meilleure classification correspond la partition P qui produit la plus grande valeur de $\mathrm{Dunn\_gen(P)}$.

- ***L'indice de Davies-Bouldin***

L'indice de *DaviesBouldin* [BEZ 92] est basé sur la minimisation du rapport des *dispersions intra-classe* et de la *séparation interclasse*. Il est calculé comme suit:

$$DB(P) = \frac{1}{k}\sum_{i=1}^{k} \max_{\substack{1\le j\le k \\ j\neq i}} \{\frac{s_a(C_i)+s_a(C_j)}{d_a(C_i,C_j)}\} \text{.....(1.15)}$$

Si le ratio est plus faible alors les classes seront compactes et éloignées les unes des autres. Par conséquence, la partition de meilleure qualité sera celle qui minimisera l'indice de *DaviesBouldin*.

- ***L'indice de Silhouette***

L'indice de silhouette est défini par [ROU 89] pour tout individu X_i de l'ensemble X par la formule suivante :

$$\forall X_i \in X; s(X_i) = \frac{b(X_i)-a(X_i)}{\max(a(X_i),b(X_i))} \text{.....(1.16)}$$

Où:

$\mathrm{a(X_i)}$ Est la dissimilarité moyenne entre l'individu $\mathrm{X_i}$ et tous les autres individus de la classe à la quelle il appartient $\mathrm{C(X_i)}$.

$$\forall X_i \in X; a(X_i) = \frac{1}{|C(X_i)|-1}\sum_{\substack{X_j\in C(X_i) \\ X_j\neq X_i}} d(X_i,X_j) \text{.....(1.17)}$$

$\mathrm{b(X_i)}$ Est le minimum des dissimilarités moyennes entre l'individu $\mathrm{X_i}$ et tous les autres individus des classes de la partition P différente de $\mathrm{C(X_i)}$.

$$\forall X_i \in X; b(X_i) = \min_{\substack{C\in P \\ C\neq C(X_i)}} d(X_i,C) \text{.....(1.18)}$$

Où
$$d(X_i,C) = \frac{1}{|C|}\sum_{X_j\in C} d(X_i,X_j) \text{.....(1.19)}$$

L'indice de silhouette est borné : $-1 \le s(X_i) \le +1$. Lorsque $s(X_i)$ est proche de 1, X_i est dit *bien classé* dans$C(X_i)$. Quant $s(X_i)$ est proche de 0, alors, X_i se situe entre deux classes. Si $s(X_i)$est proche de -1, $X_i$est dit *mal classé* dans$C(X_i)$ et doit être rattaché à un autre cluster le plus proche.

Chaque classe est aussi représentée par une silhouette qui montre quels objets sont correctement classés à l'intérieur de cette classe et lesquels n'ont simplement qu'une position intermédiaire. Pour une classe C_i donnée, son indice de silhouette est défini par la moyenne des indices de silhouette des individus qui lui appartiennent :

$$\forall C_i \in P; s(C_i) = \frac{\sum_{X_j\in C_i} s(X_j)}{|C_i|} \text{.....(1.20)}$$

L'indice de silhouette global de la partition Pest donné par la moyenne globale des largeurs de silhouettes dans les différentes classes C_i qui composent la partition :

$$s(P) = \frac{\sum_{C_i\in P} s(C_i)}{k} \text{.....(1.21)}$$

La meilleure partition retenue est alors celle qui permet d'obtenir une silhouette globale maximale.

1.4.4.2. Indices de validation externes

Connue aussi sous le nom de l'approche d'évaluation supervisée. Elle consiste à mesurer l'adéquation entre résultat d'un schéma de clustering et un partitionnement connu de l'ensemble des données [BAR 15].

L'approche de validation externe compare le clustering généré par l'algorithme de classification avec un autre est considérée comme le "meilleur clustering", de sorte qu'il correspond à une mesure de l'erreur, directement ou indirectement.

Dans le cas d'indice externes dans lesquels les résultats des clustering sont comparés à une référence "idéale" de solution de clustering, il est extrêmement important d'évaluer la qualité du clustering. Cependant, il n'existe pas de mesure standardisée pour comparer les clusters [ROM 14]. Voici quelques-unes des mesures les plus couramment utilisées:

- ***Indice de Rand ajusté***

Indice de Rand ajusté (ARI) [HUB 85] est un indice qui mesure les correspondances entre les étiquettes des classes calculées par l'algorithme de clustering et les étiquettes de classe du cluster "idéal". ARI est une valeur comprise entre -1≤ARI≤1 et plus elle est élevée, plus la similarité entre les résultats du groupement et la vérité de terrain est grande.

L'indice aléatoire RI, sans ajustement, peut alors être exprimé par:

$$RI = \frac{a+b}{A_2^n} \dots\dots\dots\dots\dots\dots (1.22)$$

Tel que: A^2 est le nombre total de paires possibles dans le jeu de données, sans ordre.

Cependant, l'indice ne garantit pas que les assignations d'étiquettes aléatoires atteindront une valeur proche de zéro, en particulier si le nombre de clusters est du même ordre que le nombre d'échantillons.

Pour contrer cet effet, les E [RI] attendus des assignations étiquetées peuvent être actualisé de manière aléatoire, afin d'obtenir un indice ARI:

$$ARI = \frac{RI - E[RI]}{max(RI) - E[RI]} \dots\dots\dots\dots\dots (1.23)$$

- ***Informations mutuelles normalisées (NMI)***

Dans la théorie de l'information, Mutual Information (MI) [VIN 09] ou trans-information quantifie la valeur des informations partagées entre deux variables aléatoires, c'est-à-dire mesure la réduction de l'incertitude (entropie) d'une variable aléatoire, en raison de la connaissance, de la valeur d'une autre variable aléatoire B.

Si la vérité sur le terrain de l'affectation des classes et des classes prédites par l'algorithme de classification est connue, le MI peut être calculé comme une fonction qui mesure les coïncidences entre ces paramètres, sans prendre en compte les permutations.

$$MI(A,B) = H(A) + H(B) - H(A,B) = \sum_{i=1}^{c}\sum_{j=1}^{k}\frac{n_{i\,j}}{n} log\frac{n_{i\,j}n}{n_i n_j} \dots\dots\dots (1.22)$$

$H(\mathrm{A})$ et $H(\mathrm{B})$ sont les entropies de chaque variable indépendante, alors que $H(A,B)$ est l'entropie qui est commune aux clusters. L'entropie d'un cluster est définie comme la valeur attendue de l'information contenue si elle est vue comme une variable aléatoire.

Deux versions standardisées et bien connues de cette mesure sont: NMI (***Normalized Mutual Information)*** et AMI (***Adjusted Mutual Information)***. L'information mutuelle normalisée,

NMI est utilisée plus fréquemment dans la littérature, alors que la AMI a été proposée plus récemment et est normalisée contre le hasard.

La quantité d'informations statistiques partagées par les deux variables aléatoires représentant l'affectation de cluster et l'étiquette de classe sous-jacente. En supposant que l'entrée nij indique le nombre d'éléments appartenant au cluster et à la classe j. Alors, NMI est calculé comme suit:

$$NMI(A,B) = \frac{MI(A,B)}{\sqrt{H(A)H(B)}} = \frac{\sum_{i=1}^{c}\sum_{j=1}^{k}\frac{n_{ij}}{n}log\frac{n_{ij}n}{n_i n_j}}{\sqrt{\left(\sum_{i=1}^{c}\frac{-n_i}{n}log\frac{n_i}{n}\right)\left(\sum_{j=1}^{k}\frac{-n_j}{n}log\frac{n_j}{n}\right)}} \text{.....(1.23)}$$

Tel que $n_{i=\sum_{i=1}^{k} n_{ij}, n_j}$ =$\sum_{i=1}^{c} n_{ij}, n, c, k$ indique le nombre total d'objets, le nombre de clusters et le nombre de classes, respectivement. Sur la base de la connaissance préalable du nombre de classes, en général, le nombre de clusters est fixé égal au nombre réel de classes, c'est-à-dire que c= k.

- ***Information mutuelles ajustées (AMI)***

Adjusted Mutual Information(AMI) [VIN 09] ou l'information mutuelle ajustée est la valeur attendue de l'information mutuelle et peut être calculée à l'aide de l'équation suivante, proposée par Vinh et al:

$$\mathrm{AMI(A,B)} = \frac{\mathrm{MI-E[MI]}}{\max(\mathrm{H(A),H(B)})-\mathrm{E[MI]}} = \frac{\mathrm{MI-E[MI]}}{\sqrt{\mathrm{H(A)H(B)}}-\mathrm{E[MI]}} \text{.....(1.24)}$$

La valeur de AMI est calculée en utilisant une forme similaire à celle d'indice. Le domaine de cette fonction est le même pour toutes les valeurs normalisées de MI [VIN 09]:

$$-1 \leq AMI \leq 1 \text{.....(1.25)}$$

- ***F-mesure***

La F-mesure est une fonction utilisée souvent dans la littérature pour évaluer les algorithmes de clustering. Elle compare la qualité de clustering en tenant compte des classes correctes connues pour un jeu de données. La F-mesure est une mesure populaire qui combine la précision et le rappel. Les critères précision, rappel et F -mesure varient entre 0 et 1. Plus la valeur de ces critères est élevée plus le critère est meilleur [VAN 04].

$$F - mesure = \frac{2\times Précision\times Rappel}{Précision+Rappel} \text{.....(1.26)}$$

Le tableau 1.4 représente la matrice de confusion pour le cas d'un problème à deux classes: positive et négative. La matrice de confusion est utilisée pour visualiser, pour chaque classe de modèle, les vraies classifications versus les classifications prédites.

	Classe prédite positive	**Classe prédite négative**
Vraie positive	Vraie positive (VP)	Fausse négative (FN)
Vraie négative	Fausse positive (FP)	Vraie négative (VN)

Tableau 1.4. Matrice de confusion d'un problème a deux classes

VP: est un résultat où le modèle prédit correctement la classe positive

FP: est un résultat où le modèle prédit incorrectement la classe positive

FN: est un résultat où le modèle prédit incorrectement la classe négative.

VN: est un résultat où le modèle prédit correctement la classe négative.

Le rappel: représente le nombre de données classées correctement, comme positives par exemple, au regard du nombre de données positives qui existent dans le corpus de données.

$$Rappel = \frac{VP}{VP+FN} \text{.....(1.27)}$$

La précision: représente le nombre de données classées correctement, comme positives par exemple, par rapport au nombre de données totales reconnues comme positives.

$$Précision = \frac{VP}{VP+FP} \text{.....(1.28)}$$

1.4.4.3 Indices de validation relatifs

Les indices de validation internes évaluent uniquement le résultat de la mise en cluster effectué par l'algorithme de mise en cluster, ils ne peuvent pas percevoir la stabilité de l'algorithme par rapport aux variations des données ou à la cohérence des résultats en cas de redondance. Les indices de validation relatifs sont une famille d'indices plus complexes que les internes qui essaient de mesurer la cohérence d'un algorithme en comparant les clusters obtenus, par le même algorithme, dans des conditions différentes. Ces indices tentent souvent d'exploiter la redondance dans les données et, comme dans les indices de validation internes, ils ne nécessitent pas d'informations supplémentaires pour le processus de mise en cluster. Dans ce type d'indices de validation, on trouve [DAL 09]:

- ***L'indice FOM (Figure Of Merit)*** où, pour mesurer la cohérence, il est supposé que la redondance est incorporée dans les données de l'échantillon.
- ***Indice de stabilité*** qui mesure la capacité d'un ensemble de données, précédemment regroupé, à prédire le clustering d'un autre ensemble de données échantillonnées à partir de la même source. Pour ce faire, diviser l'ensemble de données que l'on souhaite regrouper en deux parties. L'algorithme de clustering à utiliser est appliqué à la première partie, et les étiquettes obtenues sur ces points servent à former un classifieur qui divise tout l'espace. L'indice de stabilité dépend du nombre de clusters et doit donc être normalisé lorsqu'il est utilisé pour le modèle de sélection. Un autre problème qui affecte l'indice de stabilité est le choix de la règle de classification, qui peut fortement influencer les résultats.

1.4.5. Limites des algorithmes de clustering

Malgré l'existence d'un grand nombre de méthodes de clustering ainsi que leur utilisation avec succès dans de nombreux domaines, le clustering pose encore de nombreux problèmes. Ces problèmes sont liés d'une part au manque de précision dans la définition de ce qu'est réellement un cluster mais également dans la difficulté de définir une mesure de similarité entre objets ou encore dans la définition d'une fonction objective pour un problème donné. Jain et Dubes [JAI 88] ont listé un ensemble de questions qu'il est nécessaire de se poser lorsqu'on entreprend d'effectuer une tâche de clustering. Cette liste de questions met en avant la multiplicité et surtout la nature différente des paramètres à prendre en compte dans ce type d'approche:

- Qu'est-ce qu'un cluster?
- Quels attributs doivent être utilisés?

- Quels algorithme de clustering doit être utilisé? Le résultat de l'algorithme de clustering peut être interprété de différentes façons**.**
- Quel est le nombre de clusters présents? Beaucoup d'algorithmes de clustering exigent la spécification du nombre de clusters à produire en entrée de l'ensemble de données, avant l'exécution de l'algorithme.
- Comment déterminer la similarité entre deux objets? La mesure de la distance n'existe pas, nous devons la «définir», ce qui n'est pas toujours facile, surtout dans des espaces multidimensionnels parce que l'efficacité de la méthode dépend de la définition de «distance» utilisée.

1.4.6. Caractéristiques des méthodes de clustering

Quel que soit le type de la classification, il y a trois éléments permettent de caractériser les différentes méthodes :

1. La classification se déroule séquentiellement en regroupant les observations les plus "semblables" ou similaires (méthodes hiérarchiques) ou elle regroupe en k groupes toutes les observations simultanément (méthodes non-hiérarchiques).
2. Le critère de "ressemblance" (dissimilarité) entre deux observations.
3. Le critère de "ressemblance" entre deux groupes ou entre une observation et un groupe.

Ces trois éléments permettent de définir le déroulement ainsi que le type de la méthode, le deuxième et le troisième caractère ont un point primordial dans la performance et la qualité du résultat attendu d'une méthode, car il y aura certainement une différence de calcul (précision) entre le fait d'utiliser la distance euclidienne au lieu de la distance de Hamming (c'est à dire que la distance utilisée est prise en considération afin d'améliorer les résultats) [GOW 86].

1.4.7. Synthèse des algorithmes de clustering

Plusieurs méthodes sont proposées pour le problème général du clustering comme mentionné dans le tableau 1.5. Ils diffèrent par les mesures de proximité qu'ils utilisent, la nature des données qu'ils traitent et les objectifs finaux du clustering. Chacune de ces méthodes possède ses points forts et ses points faibles. Donc le choix de l'une de ces méthodes de clustering se fait selon les objectifs attendus (hiérarchie, grille, densité...), ainsi que la nature des individus à partitionner.

Méthode		Paramètres d'entrée	Type de données	Points forts	Points faibles
Hiérarchique	*AGNES*	Ensemble d'individus X	Numériques	-Simple -Donne le meilleur résultat dans certains cas	-Sensible au bruit et les données aberrantes **-**Absence d'une fonction objective
	DIANA	Ensemble d'individus X	Numériques	-Eviter l'exploration de toutes les possibilités de divisions	-Sensible au bruit et les données aberrantes **-**Absence d'une fonction objective
Partitionnement	*K-moyenne*	Nombre de classes K	Numériques	-Simple -Compréhensibles -Applicable à des données de grande taille	-Le nombre des clusters doit être fixé au départ, -Ne détecte pas les données bruitées -Le résultat dépend du tirage initial des centres des clusters
	PAM	Nombre de classes	Numériques	-Simple et compréhensible	-Ne traite pas les points aberrants -Avoir une certaine tolérance au bruit
Densité	*DBSCAN*	-Rayon d'un cluster; -Nombre minimum d'individus dans un cluster.	Numériques	**-**Traite les points aberrants -Peut avoir des clusters aux formes arbitraires	-Les résultats obtenus sont très sensibles au choix des paramètres d'entrée
	DENCLURE	-Rayon d'un cluster; -Nombre minimum d'individus	Numériques	**-**Permet de trouver des clusters aux formes arbitraires **-**Traite les bruits -Le nombre de clusters n'est pas fixé a priori	-Le choix du paramètre est difficile
Conceptuelles	*COBWEB*	-Le nœud courant -La hiérarchie de concepts -le nouvelle individu à incorporer	Catégoriques	**-**Incrémentale	**-**Sensible à l'ordre d'entrée des individus -La difficulté de traiter les attributs numériques -Influence de l'ordre dans lequel les objets sont intégrés à la hiérarchie

Par Grille	*STING*	-Nombre de cellules au niveau le plus bas **-**Nombre d'individus dans une cellule	Spatial	-Traite les données aberrantes -Analyse rapide des données	-Peut impliquer la perte de précision dans le traitement des requêtes -Les clusters peuvent être de mauvaise qualité en débit d'une vitesse d'exécution élevée
	WaveCluster	-Wavelets, le nombre de cellules de la grille pour chaque dimension, -Le nombre d'applications de la Transformation des ondelettes	Spatial	-Une bonne qualité des classes -Il manipule avec succès les données atypiques -Abilité de traiter des données spatiales de dimensions relativement grandes	**-**Se limite aux données multidimensionnelles qui sont décrites par des attributs numériques -Le problème de sous partitionnement -Le problème de sur partitionnement
Par probabilité	*EM*	-Le nombre de clusters -Le seuil ϵ de la log vraisemblance	Numériques	-Bien adaptée au traitement de données cachées	-Présente une complexité en temps linéaire par rapport au nombre d'individus à classer - N'assure pas que la solution retournée soit l'optimum global

Tableau 1.5. Synthèse des méthodes de clustering [BEN 16]

1.5. Problèmes des algorithmes d'apprentissage classique

- La volumétrie de données est l'une des principales difficultés rencontrées. Si les données peuvent être toutes chargées en mémoire alors l'algorithme d'apprentissage automatique classique à tout moment, accès à toutes les données. Cette hypothèse correspond au cas le plus simple sur laquelle repose la plupart des algorithmes. Mais parfois la quantité de données peut être très importante, et il est impossible de tous les charger en mémoire. Alors, il faut donc concevoir un algorithme qui puisse générer un modèle sans avoir besoin que toutes les données soient en mémoire [SAL 12];
- Si les exemples ne sont pas entièrement chargeables en mémoire ou arrivent de manière continue, la complexité calculatoire de l'algorithme d'apprentissage est supérieure à une complexité dite quasi-linéaire et le temps d'accès et de lecture des données deviennent prohibitif. Alors l'apprentissage rapide devient très difficile à réaliser [MAR 13];
- Lorsque des nouvelles classes qui n'étaient pas représentées dans le jeu de données d'apprentissage arrivent et un très grand ensemble de classes est utilisé. Les algorithmes classiques d'apprentissage deviennent inutilisables et il est toujours difficile, dans de nombreux contextes applicatifs, de prédire toutes les classes possibles requises par l'utilisateur;
- Les algorithmes d'apprentissage statique ne proposent aucune alternative d'évolutivité et d'adaptation dynamique pour intégrer de nouvelles connaissances, de nouvelles données ou pour restructurer des problèmes déjà partiellement appris;
- Il n'est pas garanti que les algorithmes de machine learning (ML) fonctionnent toujours dans tous les cas. Parfois, ML échouera. Il est donc nécessaire de comprendre le problème pour appliquer correctement l'algorithme ML. Certains algorithmes ML nécessitent beaucoup de données, tels que des algorithmes d'apprentissage incrémental ou en profondeur (Deep Learning);
- Apprendre quand et comment utiliser un algorithme d'apprentissage automatique spécifique est important. Il existe des algorithmes d'apprentissage simples et complexes, il est donc important de choisir judicieusement entre eux.

1.6. Conclusion

Ce chapitre a été dédié à une introduction à l'apprentissage automatique. Dans les premières sections, nous avons d'abord abordé quelques concepts de base puis nous avons présenté, de manière relativement succincte, les principales notions et méthodes classiques d'apprentissage supervisé et non supervisé.

Nous terminons le chapitre par la discussion de certains problèmes relatifs à ce type de méthodes. Ceci nous a permis de prendre conscience des éléments sur lesquels nous devrions agir en recherchant des solutions à ces problèmes.

CHAPITRE 2

L'APPRENTISSAGE INCREMENTAL: CONCEPTS ET ALGORITHMES

Dans le domaine de la reconnaissance de formes, l'apprentissage automatique consiste à utiliser des données brutes et prendre une décision en fonction de la catégorie ou de la classe de la forme [DUD 12]. Une fois qu'un nouvel exemple (non étiqueté) est fourni, nous souhaitons affecter une identité aux nouvelles données sans l'aide d'un humain. L'objectif ultime de l'intelligence artificielle est de développer un algorithme capable d'imiter une intelligence semblable à celle du cerveau; dans le contexte de l'apprentissage automatique, on parle alors d'apprentissage adaptatif.

Le processus d'identification des formes est crucial dans le processus de pensée de l'homme et a permis l'évolution de nos systèmes neuronaux et cognitifs qui permettent l'apprentissage de nouvelles connaissances lorsqu'elles nous sont présentées. La plasticité neuronale est l'apprentissage et le développement de nos systèmes neuronaux en réponse à une nouvelle expérience [PIN 03].

La plasticité permet l'acquisition de nouvelles informations. Il semble donc intuitif d'intégrer une forme de plasticité dans la machine à apprendre si on souhaite qu'elle réagisse de manière similaire à une intelligence humaine. La stabilité neuronale consiste à conserver les informations précédemment apprises et constitue une autre qualité précieuse pour une machine à apprendre. Par conséquent, si nous voulons que la machine d'apprentissage, ait des propriétés d'intelligence similaires à celles du cerveau, il est impératif de conserver les connaissances existantes, le cas échéant, et d'apprendre de nouvelles connaissances disponibles au fil du temps. C'est le cas de l'apprentissage incrémental…

Ce chapitre est dédié aux concepts de base de l'apprentissage incrémental, son principe, ses caractéristiques, ses catégories ainsi que les principales méthodes et algorithmes d'apprentissage incrémental supervisé et non supervisé.

2.1. Définitions de l'apprentissage incrémental

L'apprentissage automatique est une discipline bien établie qui a été étudiée au cours des dernières décennies par de nombreux chercheurs. Un grand nombre de schémas d'apprentissage et d'algorithmes ont été proposés et leurs bonnes performances ont été prouvées dans différents domaines d'application du monde réel. Néanmoins, le type d'apprentissage qui nous intéresse dans cette thèse présente de nouvelles caractéristiques qui le différencient des algorithmes d'apprentissage statique bien connus (tels que SVM, PMC, AD,...). Dans cette section, nous tentons de définir l'apprentissage incrémental.

Définition 1: [ALM 11]

En apprentissage incrémental, l'ensemble de données d'apprentissage complet n'est pas nécessairement disponible. Le système doit apprendre chaque échantillon de données séparément, sans connaître ni les prochains échantillons de données d'apprentissage ni les anciens.

Supposons que $x_i, i = 1, 2, .., n$ représentent les échantillons de données d'apprentissage, M désigne le système appris et f désigne un algorithme d'apprentissage donné. L'apprentissage incrémental peut être simplement défini comme suit:

$$M_i = f(M_{i-1}, x_i) \dots \dots \dots \dots \dots \dots (2.1)$$

Un mode d'apprentissage incrémental est parfois utilisé pour apprendre des jeux de données volumineux où il est impossible de le propager plusieurs fois au cours du processus d'apprentissage.

Définition 2: [BOU 16]

L'apprentissage incrémental est le fait d'apprendre en parcourant les données une par une, sans prendre en considération le jeu de données d'apprentissage dans son ensemble.

Définition 3: [DI 05]

Un apprenant L est incrémental si L entre une expérience de formation à la fois ne retransforme aucune expérience antérieure et ne conserve qu'une seule structure de connaissances en mémoire.

Définition 4: [HEB 08]

L'apprentissage incrémentale est la possibilité d'améliorer les performances d'un classifieur en reprenant une phase d'apprentissage lorsque des exemples supplémentaires apportant un complément d'information deviennent disponibles.

Définition 5: [MAR 13]

L'apprentissage incrémental est le fait de ne charger qu'un petit bloc de données en mémoire à la fois, de construire un modèle partiel et de le mettre à jour en chargeant consécutivement des blocs de données. Cela permet de traiter de très grandes quantités de données sans difficulté de mémoire sur une machine standard. Un algorithme incrémental peut donner des résultats approximatifs ou identiques à ceux obtenus par un algorithme d'apprentissage chargeant une seule fois l'ensemble entier des données.

Définition 6: [MUH 09]

Un algorithme d'apprentissage est incrémental si, pour une séquence d'ensembles de données d'apprentissage (ou instances), il produit une séquence d'hypothèses, où l'hypothèse courante décrit toutes les données qui ont été vues précédemment, mais dépend seulement des hypothèses antérieures et les données d'apprentissage courantes. D'où, un algorithme d'apprentissage incrémental doit apprendre la nouvelle information, et retient la connaissance acquise précédemment, sans avoir accès aux données vues précédemment.

Définition 7: [SAL 10]

L'apprentissage incrémental correspond à un système capable de recevoir et d'intégrer de nouvelles données sans devoir réaliser un apprentissage complet. Un algorithme d'apprentissage est incrémental si, pour n'importe quels exemples, il est capable de produire

des hypothèses telles que l'hypothèse suivante ne dépend que de l'hypothèse courante et de l'exemple courant.

Définition 8: [SHI 18]

L'apprentissage incrémental est un paradigme d'apprentissage automatique dans lequel le processus d'apprentissage ajuste ce qui a été appris en fonction des nouveaux exemples lorsque de nouvelles données arrivent. Le système doit être capable d'apprendre de manière incrémentale à partir de flux de données dynamiques à grande échelle, et de constituer au fil du temps une base de connaissances favorisant l'apprentissage et la prise de décision future.

Définition 9: [VIN 16]

L'apprentissage incrémental (aussi séquentiel ou en ligne/ online learning), est le processus par lequel une entité accroît ses connaissances au cours du temps, en même temps qu'elle les utilise.

Définition 10: [ZIA 16]

Dans ce type d'apprentissage, le réapprentissage commence à partir du niveau de base, les anciennes données sont remplacées par la combinaison des anciennes (si les anciennes données seraient disponibles à ce moment-là) et des nouvelles données. Il s'agit d'un phénomène connu sous le nom d'oubli ou d'interférence catastrophique [FRE 99]. Il est très courant que les données d'apprentissage soient disponibles par petits lots pendant de petites périodes. De nouvelles instances de classes peuvent également être introduites dans le nouvel ensemble de données. Les informations doivent être tirées des nouvelles données ainsi que des données précédentes.

Donc, il doit y avoir une solution dans laquelle nous pouvons intégrer de nouvelles données d'apprentissage sans oublier les données précédentes.

2.2. Compromis plasticité/stabilité

Le dilemme plasticité/stabilité traité par la théorie de la résonance adaptative [CAR 91] désigne l'équilibre à trouver la capacité d'un système évolutif à intégrer de la nouveauté dans son modèle, sans pour autant oublier la connaissance existante et compromettre sa stabilité.

Formellement, un algorithme d'apprentissage est totalement stable s'il garde en mémoire les connaissances acquises sans oubli catastrophique. Cependant, il n'est pas nécessaire de prendre en compte de nouvelles connaissances. Au contraire, un algorithme d'apprentissage est complètement plastique s'il est capable d'apprendre continuellement de nouvelles connaissances sans qu'il soit nécessaire de préserver les connaissances précédemment acquises.

Alors le dilemme vise à accueillir de nouvelles données (plasticité) sans oublier (stabilité) en générant des éléments de connaissance au fil du temps, chaque fois que les nouvelles données véhiculent de nouveaux éléments de connaissance.

L'objectif de l'apprentissage incrémental est de faire un compromis entre les extrémités stabilité et plasticité du spectre d'apprentissage, comme illustré à la figure 2.1.

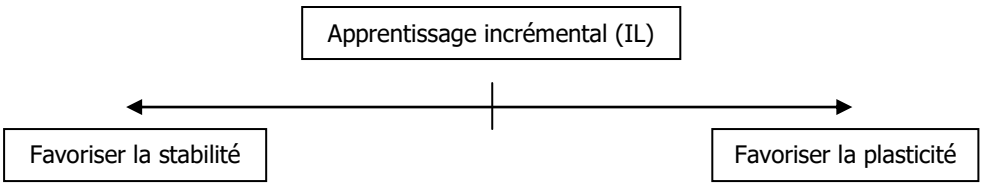

Figure 2.1. Spectre d'apprentissage

Par conséquent, on peut ajouter comme définition de l'apprentissage incrémental:

- L'IL devrait être capable d'adapter la plasticité en acquérant des connaissances à partir de nouvelles données. Ces données peuvent faire référence à la structure déjà connue ou à une nouvelle structure du système;
- IL ne peut utiliser que de nouvelles données et ne doit avoir accès à aucun moment aux données précédemment utilisées pour mettre à jour le système existant;
- IL devrait pouvoir observer la stabilité du système en évitant d'oublier

2.3. Principe de l'apprentissage incrémental

Le principe de l'apprentissage incrémental est basé sur la remarque suivante: les performances d'un tel modèle dépendent fortement de la « qualité » de la base d'apprentissage qu'on lui fournit. Il est difficile de savoir a priori quelles sont les données pertinentes à mettre dans cette base puisque cela dépend à la fois du problème traité et de la méthode d'apprentissage utilisée [LEC 95].

Un algorithme d'apprentissage incrémental peut être exécuté en cinq étapes [MIS 06]:

1. Apprendre les règles à partir des données de la base d'apprentissage;
2. Stocker les nouvelles règles et supprimer les exemples les plus anciens;
3. Utiliser les règles apprises pour prédire et naviguer;
4. Lorsque de nouveaux exemples arrivent, apprendre de nouvelles règles en utilisant d'anciennes règles et de nouvelles instances;
5. Passer à l'étape 2.

2.4. Critères de l'apprentissage incrémental

Un algorithme d'apprentissage incrémental est défini dans [POL 01] comme répondant aux critères suivants:

- Il devrait pouvoir apprendre des informations supplémentaires à partir de nouvelles données;
- il ne devrait pas exiger d'accès aux données originales (c'est-à-dire les données utilisées pour former le classifieur existant);
- Il devrait préserver les connaissances précédemment acquises (il ne devrait pas souffrir d'un oubli catastrophique, ni d'une perte importante des connaissances acquises à l'origine);
- Et il devrait pouvoir prendre en charge de nouvelles classes pouvant être introduites avec de nouvelles données.

Un grand nombre des algorithmes existants d'apprentissage incrémental ne sont pas vraiment incrémentaux parce qu'il manque au moins l'un des critères mentionnés au dessus. Ces critères peuvent être brièvement exprimés par le «dilemme de la stabilité et de la plasticité» [WIL 05], selon lequel un système doit pouvoir apprendre pour s'adapter à un environnement en mutation, mais ce changement constant peut conduire à un système instable capable d'apprendre de nouvelles informations uniquement en oubliant tout ce qu'il a appris jusqu'à présent.

2.5. Principes de conception de l'apprentissage incrémental

Six principes de conception doivent être pris en compte lors de la conception d'un système de classification de l'apprentissage incrémental. Nous les résumons comme suit [WU 07]:

1) Le coût de mise à jour de la méthode doit être faible;
2) La méthode doit accepter les observations en tant qu'entrée décrite par toute combinaison de variables symboliques et numériques, parfois de variables continues;
3) La méthode devrait être capable de gérer plusieurs classes comme sorties bien que deux classes;
4) La méthode doit être suffisamment puissante pour gérer les données bruitées et incohérentes;
5) Capable de traiter des données établies, la méthode devrait prendre en compte la possibilité que les données entre les catégories soient déséquilibrées;
6) Capable de gérer certains problèmes avec des relations fortes entre plusieurs attributs.

Les algorithmes d'apprentissage incrémental offrent de nouveaux défis aux chercheurs [WU 07]:

(1) comment mémoriser les connaissances acquises pour une mise à jour ultérieure sans enregistrer chaque cas appris précédemment; (2) comment oublier (ou conserver) les effets d'ordre dans lesquels les cas ont été appris; (3) comment concevoir un algorithme de mise à jour rapide; (4) comment rendre les résultats d'apprentissage interprétables aux être-humains.

2.6. Caractéristiques d'un système d'apprentissage incrémental

Comme indiqué dans [LAN 95], les trois hypothèses les plus importantes qui caractérisent un système d'apprentissage incrémental sont les suivantes:

a) Il doit être capable d'utiliser les connaissances acquises à n'importe quelle étape de l'apprentissage;

b) L'incorporation de l'expérience en mémoire pendant l'apprentissage devrait être efficace du point de vue du calcul (la révision de la théorie doit être efficace pour nouvelles observations entrantes);

c) Le processus d'apprentissage ne doit pas engendrer des besoins d'espace déraisonnables, de sorte que les besoins en mémoire augmentent en fonction de l'expérience (les besoins en mémoire ne doivent pas dépendre de la taille d'entraînement).

Un algorithme d'IL, à notre avis, devrait remplir les caractéristiques suivantes:

- Capacité d'apprentissage tout au long de la vie et de faire face à la plasticité et à la stabilité;
- Les anciennes données ne sont jamais utilisées aux étapes suivantes;
- Aucune connaissance préalable de la structure (topologique) du système n'est nécessaire;
- Capacité à ajuster progressivement la structure du système;
- Aucune connaissance préalable des propriétés statistiques des données n'est nécessaire;
- Aucune connaissance préalable sur le nombre de classes existantes et le nombre de catégories par classe et aucune initialisation de prototype n'est requise.

2.7. Catégories de l'apprentissage incrémental

Le système d'apprentissage incrémental est séparé en deux catégories : les systèmes adaptatifs et les systèmes évolutifs [CHE 19a]:

- **Systèmes adaptatifs:** sont les systèmes d'apprentissage incrémental qui vont apprendre de manière incrémentale leurs paramètres mais dont la structure est fixe. Ce type d'apprentissage peut être considéré comme un algorithme « d'adaptation ». La structure de ces systèmes est initialisée au départ et reste ensuite inchangée pendant l'apprentissage et les paramètres d'apprentissage sont adaptés pendant l'apprentissage.
- **Systèmes évolutifs:** sont une sous-catégorie des systèmes incrémentaux, capables de modifier leur structure. Ils vont non seulement apprendre leurs paramètres de manière incrémentale, mais également modifier leur structure, comme par exemple lors de l'apparition d'une nouvelle classe et les méthodes d'augmentation et d'élagage de réseau

Un système évolutif est donc bien plus flexible qu'un système adaptatif puisqu'il permet, en cours d'utilisation, d'ajouter des classes supplémentaires lorsque cela s'avère nécessaire.

2.8. Avantages de l'apprentissage incrémental

1. L'apprentissage incrémental présente l'avantage d'éliminer a priori le problème de la représentativité de la base d'apprentissage [HEB 08];
2. En apprentissage incrémental, vous apprenez vite, vous acquérez une énorme quantité de connaissances, vous vous souvenez de presque tout ce que vous avez appris, vous comprenez mieux les choses[4];
3. L'avantage majeur de cet apprentissage est que nous n'avons pas besoin de stocker les données déjà vues; les connaissances ont été stockées dans la série d'hypothèses développées au cours de la vie d'apprentissage [ADE 13];
4. L'apprentissage incrémental permet de gérer efficacement des modèles de classe précis et à jour. Un autre avantage de l'apprentissage incrémental est la faible complexité de calcul requise pour mettre à jour un classifieur [GRA 08];
5. L'apprentissage incrémental peut constituer un outil puissant dans les applications centrées humain [GRA 08];

[4] http://super-memory.com/archive/help16/advantages.htm

6. L'apprentissage incrémental est économique, dans l'espace et dans le temps, car il évite de stocker et de traiter à nouveau des anciennes instances. Il est plus approprié pour les tâches d'apprentissage dans lesquelles des ensembles de données d'apprentissage deviennent disponibles sur une longue période [CHE 07, MUR 08].

2.9. Méthodes et algorithmes d'apprentissage incrémental

Durant les dernières années, plusieurs travaux ont été effectués dans le domaine d'apprentissage incrémental. Dans les sous sections suivantes, nous allons décrire les principales versions incrémentales proposées, en particulier les algorithmes d'apprentissage supervisé et non supervisé vus précédemment dans le chapitre 1:

2.9.1. Les architectures ART et le modèle ARTMAP

Les architectures ART (Adaptive Resonance Theory) inspirés des travaux de Grossberg [GRO 82] sont basées sur une théorie inspirée de la biologie nommée "Théorie de la Résonance Adaptative". Cette théorie est cognitive et neurale de la manière dont le cerveau apprend de manière autonome à catégoriser, reconnaître et prédire des objets et des événements dans un monde en changement, et un ensemble de algorithmes incorporant de manière informatique les principes de l'ART et utilisés dans les applications techniques et d'ingénierie à grande échelle où il est nécessaire d'apprendre rapidement, de manière stable et incrémentale, dans des environnements complexes et en constante évolution.

L'ART clarifie les processus conceptuels à partir desquels des expériences conscientes émergent. Il prédit un lien fonctionnel entre les processus de conscience, d'apprentissage, d'attente, d'attention, de résonance et de synchronisation, y compris la prédiction selon laquelle «tous les états conscients sont des états de résonance». Cette connexion explique comment la dynamique du cerveau permet à un individu de s'adapter de manière autonome en temps réel dans un monde en changement rapide. ART prédit comment fonctionne l'apprentissage rapide et stable des catégories de reconnaissance. En particulier, ART définit le rôle critique que doivent jouer les états «résonants» pour favoriser un apprentissage rapide et stable, d'où le nom résonance adaptative. Ces états de résonance sont liés ensemble, en utilisant un retour d'information de haut en bas sous la forme d'attentes apprises, dans des représentations cohérentes du monde. ART clarifie comment le cerveau effectue un calcul prédictif. Les algorithmes ART ont été utilisés dans des applications à grande échelle telles que la prédiction de bases de données médicales, la télédétection, la conception d'avion et le contrôle de robots adaptatifs autonomes.

Les réseaux ART sont des réseaux à compétition [BOU 09]: s'il y a une approximation suffisante entre l'entrée du réseau et le prototype déjà connu par le réseau alors les valeurs de poids sont adaptées et il existe une résonance. Sinon, l'entrée du système est très éloignée du prototype et une nouvelle classe est créée.

Dans ART, le nombre de classes est préalablement fixé, celles-ci étant créées dynamiquement en fonction de la structure de l'ensemble des données présentées. La sélection de ces classes dépend de leur nombre total. En effet, le degré de nouveauté d'une donnée est mesuré par rapport à un paramètre de seuil fixé par l'opérateur. Ce paramètre, appelé seuil de vigilance (de simimlarité) proposé par Grossberg [GRO 76], qui contrôle la sélectivité des classes.

Le paramètre de vigilance ρ est défini par l'utilisateur du système à l'aide d'une valeur comprise entre 0 et 1 déterminant le nombre de catégories reconnues. La valeur choisie influe grandement sur la qualité ainsi que la taille des classes à obtenir. Plus la valeur est proche de 1, plus les classes sont nombreuses et plus la classification est stricte. Par contre, plus la valeur est proche de zéro, plus les classes obtenues seront volumineuses mais peu nombreuses. En pratique, la valeur de ce paramètre est définie par l'utilisateur par essai et erreur.

Les méthodes ART fonctionnent comme suit [MAM 13] :

- Au début l'ensemble des vecteurs prototypes est déterminé.
- Si un nouveau patron est introduit afin d'être classé, le système le compare à l'ensemble des vecteurs prototypes existants.
- Le système identifie le vecteur prototype le plus proche du patron d'entrée et calcule la distance entre le vecteur prototype sélectionné et le patron d'entrée grâce à une fonction G différente selon le type de réseau ART utilisé.
- Si le patron d'entrée est suffisamment similaire au vecteur prototype (selon le seuil de vigilance ρ), le patron d'entrée est inséré dans la classe décrite par le vecteur prototype sélectionné et le vecteur prototype est ajusté.
- Un prototype existant est modifié seulement si le patron courant est suffisamment similaire à l'individu moyen de ce prototype, sinon, ce patron est ajouté comme vecteur prototype aux prototypes existants.

Ce mécanisme est repris à partir de la troisième étape avec chaque nouveau patron.

2.9.1.1. Le modèle ART1

Le modèle ART1 [CAR 87] est un algorithme d'apprentissage non supervisé qui possède la particularité de prendre en entrée un ensemble de vecteurs binaires et créer à partir desquels des regroupements sur la base de certains critères de similarité.

L'architecture ART1 simple consiste en deux couches de neurones : une couche d'entrée qui comprend les neurones représentant les caractéristiques d'entrée, et une couche de sortie qui comprend les neurones représentant le concept de sortie. Les connexions b_{ij} de chaque entrée i à chaque sortie j et les connexions t_{ji} des sorties en arrière aux entrées sont montrées dans la figure. Chacun des neurones de sortie a une connexion excitatrice forte à lui-même et une connexion inhibitrice forte à chacun des autres neurones de sortie. Les poids des connexions t_{ji} représentent les modèles prototypes appris et les poids des connexions b_{ij} représentent un plan pour les nouvelles entrées devant être accommodées dans le réseau. Les modèles, associés à un neurone de sortie j, sont représentés collectivement par le vecteur poids de ce neurone t_j (vecteur prototype). La réaction du neurone j à un nouveau vecteur d'entrée particulier est définie par un autre vecteur poids b_j.

L'algorithme d'apprentissage ART1 pour les entrées et les sorties binaires est donnée au-dessous. Il consiste en deux phases majeures. La première présente le modèle d'entrée et calcule les valeurs d'activation des neurones de sortie. Le neurone gagnant est défini. La deuxième phase est pour calculer la désadaptation entre le modèle d'entrée et le modèle associé actuellement au neurone gagnant. Si la désadaptation est au-dessous d'un seuil (paramètre de vigilance) ce modèle est mis à jour pour accommoder le nouveau neurone. Mais si la désadaptation est au dessus du seuil, la procédure continue soit à trouver un autre neurone de sortie, soit à créer un nouveau neurone.

Le fonctionnement de modèle ART1 est résumé par le code suivant:

Algorithme 2.1 (ART1)

1. Les coefficients des poids sont initialisés:

$t_{ij}(0): 1, b_{ij}: = 1/(1+n)$, pour chaque $i = 1,2,\dots,n;\ j = 1,2,\dots m$

2. Un coefficient de similarité r, prétendu un facteur de vigilance, est défini, $0<=r<=1$. Plus la valeur de r est grande, les modèles les plus similaires devraient être dans l'ordre d'activer le même neurone de sortie représentant une catégorie, une classe, ou un concept.

3. Tant que (il y a des vecteurs d'entrées) faire

a) Un nouveau vecteur d'entrée $x(t)$ est acheminé au moment $t, x = (x_1, x_2, \dots, x_n)(t)$

b) Les sorties sont calculées:

$O_j = b_{ij}(t).x_i(t)$, for $j=1, 2\dots, m$

c) Une sortie O_{j*} avec la plus haute valeur est définie;

d) La similarité du modèle d'entré associé à $j*$ est définie :

SI (nombre de "1" dans l'interaction du vecteur $x(t)$ et $t_j*(t)$) divisé par le nombre de "1" dans $x(t)$ est plus grand que la vigilance r) Alors aller à (f)

SINON

e) La sortie $j*$ est abandonnée et la procédure revient à (b) afin de calculer une autre sortie qui devant être associée à $x(t)$;

f) Le modèle $x(t)$ est associé au vecteur $tj*(t)$, par conséquent le modèle $tj*(t)$ est changé en utilisant son interaction avec $x(t)$:

$t_{ij}*(t+1) := t_{ij}*(t).xi(t)$, pour $i=1,2,\dots,n$

g) Les poids b_{ij} sont changés :

$b_{ij}*(t+1) := b_{ij}*(t) + t_{ij}*(t).x_i/(0.5 + t_{ij}*(t).x_i(t))$

2.9.1.2. Le modèle Fuzzy ARTMAP

ARTMAP est un algorithme supervisé, rapide, possédant un apprentissage en-ligne et incrémental, introduit par Carpenter et al. [CAR 92]. Son fonctionnement est de classifier les entrées par l'ensemble des caractéristiques qu'elles possèdent ; ces valeurs sont binaires (absence ou présence de chaque caractéristique). Il peut apprendre de nouveaux événements rencontrés grâce à son architecture incrémentale. Carpenter et al. ont développé, après une année, un autre modèle Fuzzy ARTMAP qui permet de classifier les entrées par un ensemble de données floues. Le modèle intègre la fonction d'appartenance dont la valeur se situe entre 0 et 1. L'algorithme Fuzzy ARTMAP est constitué du réseau fuzzy ART [CAR 91] qui utilise un algorithme non supervisé pour la classification des données. Il apprend rapidement sur les poids grâce à son utilisation d'une stratégie de vote.

L'architecture de Fuzzy ARTMAP (voir figure 2.2) inclut une paire de réseau ART: ART_a *et* ART_b. Les deux réseaux sont reliés entre eux par le réseau intermédiaire F^{ab}

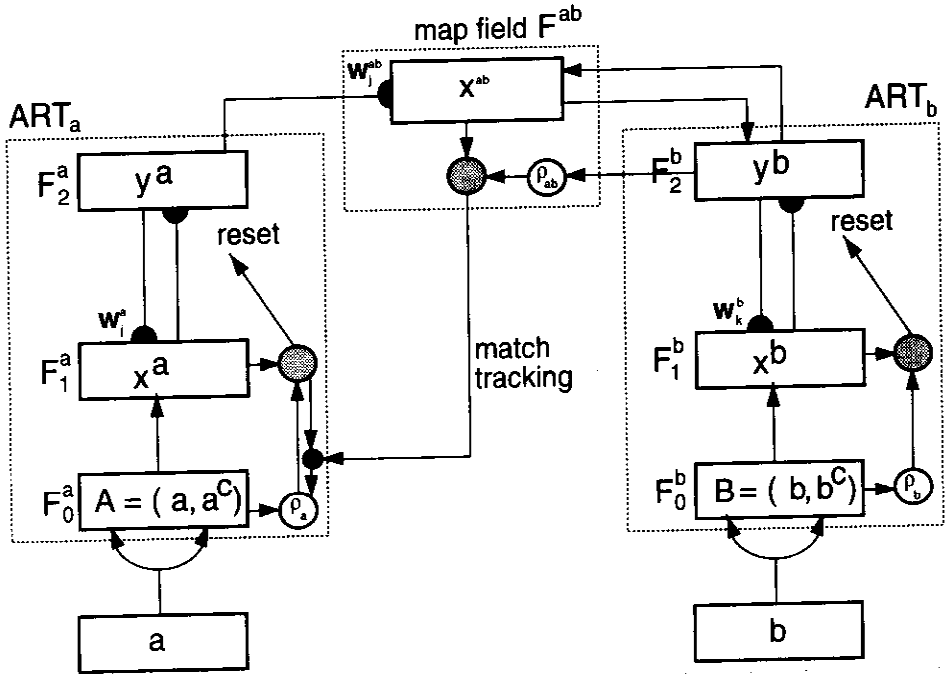

Figure 2.2. L'architecture de Fuzzy ARTMAP [CAR 92]

Généralement, l'inconvénient des méthodes ART est, leur sensibilité aux données, en particulier le bruit des données de la base d'apprentissage ce qui rend la sélection du seuil de similarité une tâche difficile.

2.9.2. PBIL: Population-Based Incremental Learning

L'apprentissage incrémental basé sur la population (PBIL) est une technique d'optimisation stochastique simple qui peut être appliquée rapidement à un problème. L'algorithme a été décrit pour la première fois en 1994 [BAL 94] et a été amélioré plus récemment [GRE 97].

PBIL est une méthode de combinaison d'algorithmes génétiques (AG) et d'apprentissage compétitif pour l'optimisation des fonctions. PBIL est une extension de l'AG, il prend sa simplicité et le dépasse en termes de rapidité et de précision.

PBIL utilise un vecteur de probabilité $p(x)$ qui s'appelle le prototype de la population pour construire un ensemble de m solutions à chaque itération. Il utilise aussi un coefficient d'apprentissage α qui contrôle l'amplitude des changements pour mettre à jour $p(x)$.

L'algorithme PBIL peut être décrit comme suit [KAR 04] :

Algorithme 2.2 (PBIL)
Initialiser le vecteur de probabilité, P;
Répéter
Générer des échantillons :
Générer un vecteur d'échantillon en fonction de probabilités dans P ;
Évaluer le vecteur d'échantillon;
Trouver le vecteur correspondant à l'évaluation maximale

(max ≡ meilleur échantillon)

Mettre à jour le vecteur de probabilité selon la règle suivante :

$P_i = P_i*(1 - LR) + maxi*(LR)$

où LR représente le taux d'apprentissage

Muter le vecteur de probabilité

si ($random\ (0, 1]$ < probabilité de mutation)

$P_i = P_i*(1 - Mut\_Shift) + rand(0, 1]*(Mut\_Shift)$

où Mut_Shift est la quantité de mutation affectant le vecteur de probabilité

2.9.3. ILFN: Incremental Learning Fuzzy Neural Network

ILFN [YEN 01] est un réseau neuronal flou à apprentissage incrémental. Le classifieur ILFN utilise des fonctions à base radiales gaussiennes pour structurer les bordures de décision. Il utilise un système hybride d'apprentissage supervisé et d'apprentissage non supervisé pour générer ses prototypes. Cet algorithme répond aux critères d'apprentissage incrémental décrit dans la section (2.4) et peut être considéré comme une stratégie d'apprentissage rapide. Cependant, il utilise une structure de prototype de base qui ne peut pas faire face à la complexité de certains problèmes de classification.

L'architecture réseau du classifieur ILFN est la suivante [YEN 99]:

Il se distingue par deux modes différents: un mode d'apprentissage (illustré à la figure 2.3) et un mode de fonctionnement (illustré à la figure 2.4). Les deux modes ne diffèrent que par le module de contrôleur et le module d'étiquetage cible. Le mode d'apprentissage utilise le schéma d'apprentissage supervisé nécessitant des paires d'entrées et de formes cibles pour construire des prototypes du système. D'autre part, le mode de fonctionnement utilise l'algorithme d'apprentissage non supervisé pour déterminer la classe cible pour un modèle d'entrée donné. Lorsque le système détecte de nouvelles catégories, il utilise le module d'étiquetage de cible pour affecter les cibles correspondantes aux modèles d'entrée à venir. Les cibles assignées aux nouveaux prototypes sont très différentes des cibles existantes dans le module cible.

Le système ILFN comporte quatre couches: une couche d'entrée, une couche cachée, une couche de sortie et une couche de décision, comme illustré aux figures (2.3) et (2.4). Généralement, le système est composé de deux sous-systèmes: un sous-système d'entrée et un sous-système cible. Chaque sous-système comporte trois couches: une couche d'entrée, une couche cachée et une couche de sortie. Les couches cachée du sous-système d'entrée et du sous-système cible sont liées entre elles via un module de contrôle qui permet de contrôler les neurones en croissance dans la couche cachée. Chaque couche de sortie des deux sous-systèmes est composée de deux modules. La couche de sortie du sous-système d'entrée comprend un module d'élagage et un module d'appartenance, tandis que la couche de sortie du sous-système cible comprend un module d'élagage et un module cible. Le module d'appartenance du sous-système d'entrée et le module cible du sous-système cible sont mis à jour simultanément avec leur nombre de neurones contrôlés par les modules d'élagage. La sortie du classifieur est liée via une couche de décision.

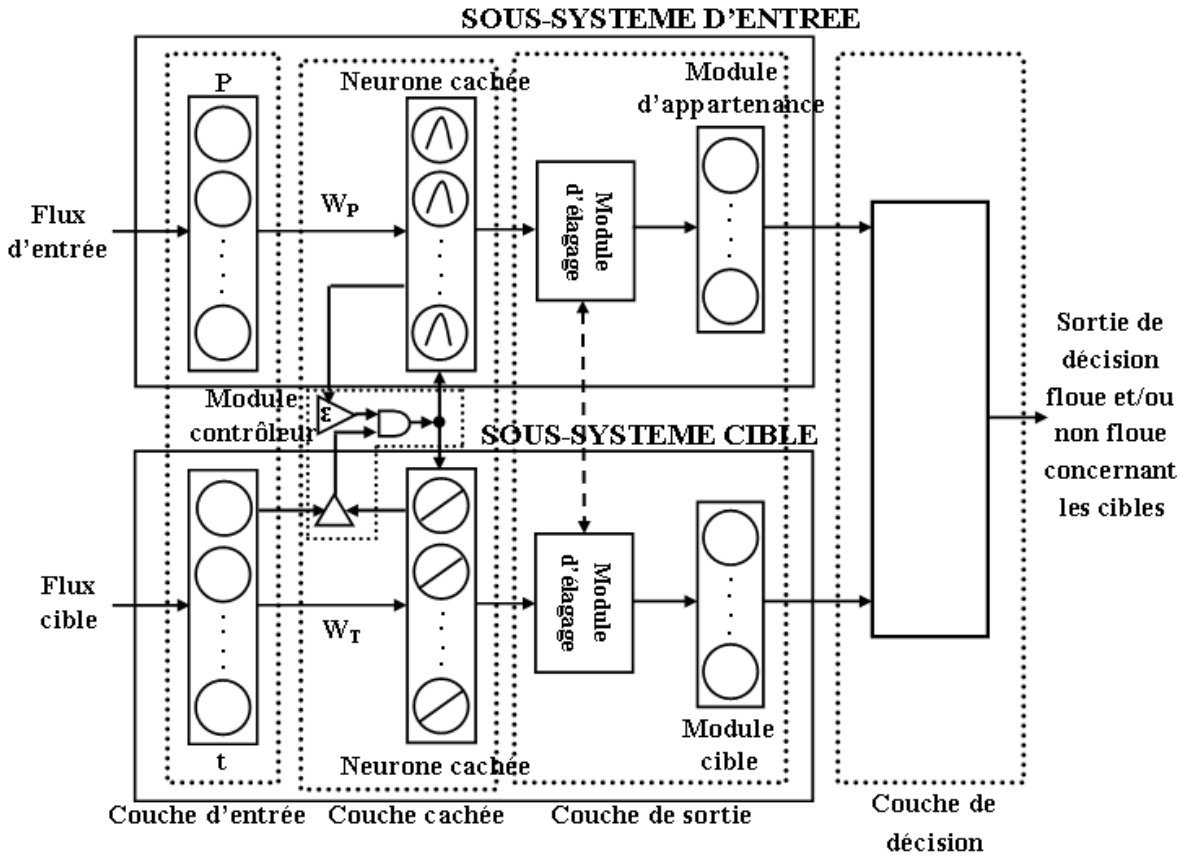

Figure 2.3. Architecture réseau du classifieur ILFN dans le mode d'apprentissage [YEN 99]

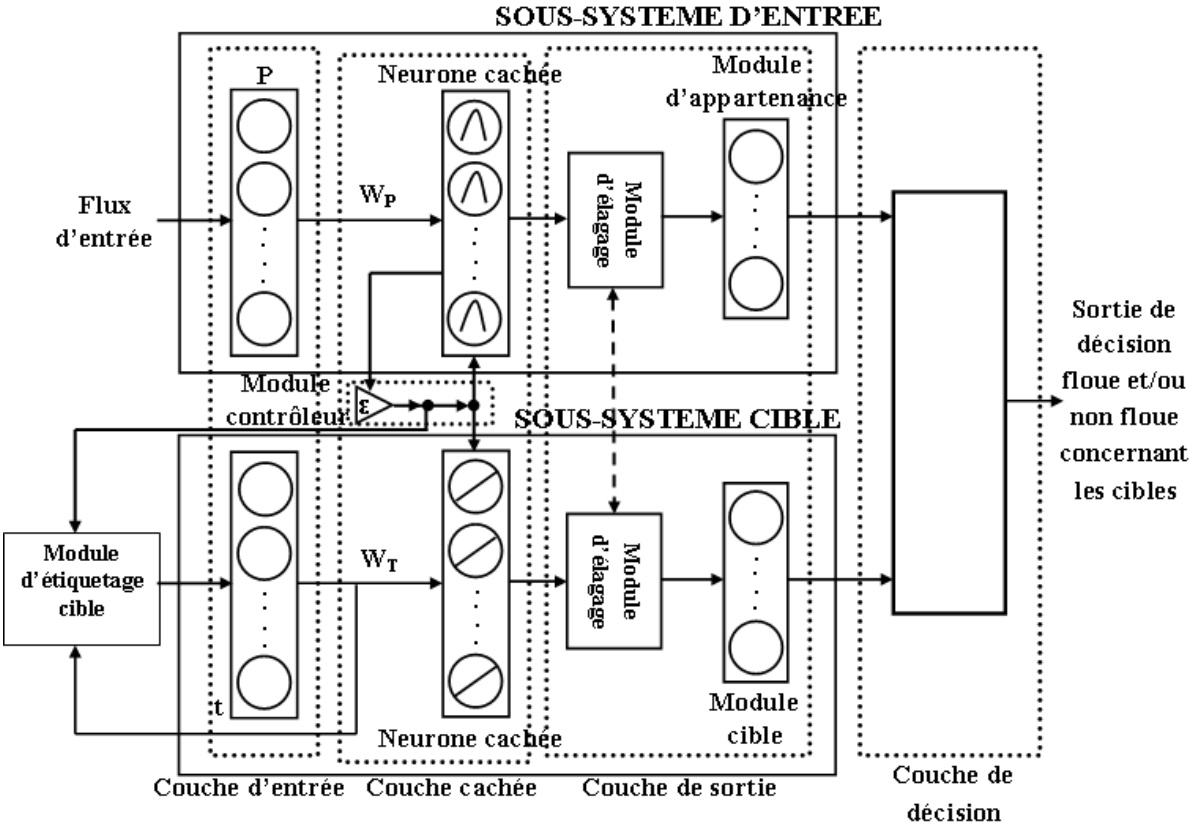

Figure 2.4. Architecture réseau du classifieur ILFN dans le mode de fonctionnement [YEN 99]

Algorithme 2.3 (IFLN)

Étape 1: Mettre le paramètre seuil défini par l'utilisateur (ε), la déviation standard initiale $\sigma_0$et le nombre maximal de modèles permis à inclure dans chaque cluster qui est utilisé dans le mode d'exploitation.

Étape 2: Lire dans le premier modèle d'entrée

– Utiliser le premier modèle d'entrée pour installer le premier prototype (ou moyenne) à Wp.

– Mettre le nombre de modèles pour le premier nœud à être 1.

– Mettre la déviation standard égale à la déviation standard initialeσ_0.

– Mettre un nouveau neurone à W_T en utilisant la première cible t pour être la cible correspondante du prototype dans W_p.

Étape 3: Lire dans le prochain exemple d'apprentissage avec une entrée et un modèle cible.
Étape 4: Mesurer la distance Euclidienne entre l'entrée p et le prototype W_p.

Étape 5: Calculer les valeurs d'appartenance pour chaque nœud en utilisant la fonction à base radiale de type Gaussien.

Étape 6: Assigner des valeurs d'appartenance à chaque nœud. Le modèle d'entrée courant a des degrés différents pour chaque nœud ou sous-classe. Pour chaque classe, sélectionner la valeur d'appartenance maximale à partir de chaque sous-classe pour représenter le degré de similarité en ce qui concerne cette classe.

Étape 7: Identifier la plus grande appartenance en utilisant l'opérateur floue **OU**.

Étape 8: Pour le nœud gagnant, S'il y une a cible correspondante (c.-à-d., il est dans le mode d'apprentissage),

1. si le gagnant est plus grand que ε et la cible t est la même valeur que W_T au nœud gagnant alors la mise à jour de poids W_p, la déviation standard et le nombre de modèles appartenant à ce nœud.

2. si 1) n'est pas satisfait, alors:

- Mettre un nouveau centre du nœud pour W_p en utilisant le modèle d'entrée p.
- Mettre le nombre de modèles pour le nouveau nœud à être 1.
- Mettre la déviation standard initiale au nouveau nœud.
- Ajouter un nouveau neurone à W_T, en utilisant la nouvelle cible t comme la cible correspondante d'un nouveau prototype dans W_p.

S'il n'y a aucune cible correspondante (c.-à-d., il est dans le mode d'exploitation),

1. si le gagnant est plus grand que ε et le nombre de modèles est moins que le nombre maximal de modèles permis, alors mettre à jour le poids W_p, la déviation standard et le nombre de modèles qui appartenant à ce nœud. Identifier la classe de sortie qui est stockée dans WT au même indice du nœud gagnant de W_p.

2. si le gagnant est plus petit que ε alors

- Mettre un nouveau centre du nœud W_P en utilisant le modèle d'entrée p.
- Mettre le nombre de modèles pour le nouveau nœud à être 1.
- Mettre la déviation standard initiale au nouveau nœud.

– Ajouter un nouveau neurone à W_T et affecter une nouvelle cible comme la cible correspondante d'un nouveau prototype dans W_P.

Étape 9: S'il n'y a plus de modèles d'entrée, alors arrêter. Autrement, aller à étape 3.

2.9.4. Algorithme LEARN++

Learn++ [POL 01] est un algorithme d'apprentissage incrémental d'un réseau de neurones inspiré de l'algorithme AdaBoost. Il se base sur le principe d'une combinaison de classifieurs faibles pour prendre une décision. Le système va entraîner plusieurs classifieurs sur plusieurs sous-ensembles de l'ensemble d'apprentissage. Les difficultés de cet algorithme résident dans la création des sous-ensembles d'apprentissage et la combinaison de ces classifieurs.

L'idée de Learn++ est de modifier la distribution des éléments dans le sous ensemble d'apprentissage afin de renforcer la présence des éléments les plus difficiles à classifier en fonction des erreurs du classifieur faible généré. Cette procédure est ensuite répétée avec un ensemble différent de données de la même base d'apprentissage et de nouveaux classifieurs sont générés. La sortie sera la combinaison des sorties des classifieurs en utilisant le vote majoritaire. Les classifieurs faibles sont des classifieurs qui fournissent une estimation grossière d'une règle de décision car ils doivent être très rapides à générer.

L'algorithme Learn++ reçoit en entrée une distribution d'exemples d'apprentissageDt, à partir de laquelle les sous-ensembles d'apprentissage sont choisis.

Un algorithme d'apprentissage faible *WeakLearn* qui sera utilisé comme classifieur de base, génère des hypothèses multiples en utilisant des sous-ensembles différents des données d'apprentissage S_k et chaque hypothèse apprend seulement une portion de l'espace d'entrée.

Pour la première itération, les poids $w_1\,(i)$ sont initialisés à $1/m$

A chaque itération $t=1, 2, \ldots, T_k$, Learn++ divise S_k en :

-un sous-ensemble d'apprentissage TR_t et

-un sous-ensemble de test TE_t d'après la distribution courante D_t.

Faire appel à *WeakLearn* pour générer l'hypothèse h_t en utilisant le sous-ensemble d'apprentissage TR_t

Ensuite on calcule l'erreur ε_t de ce classifieur

Si $\varepsilon_t > 1/2$, h_t est ignoré et des nouveaux TR_t et TE_t sont sélectionnés.

Sinon la sortie est l'erreur normalisée β_t

- Toutes les hypothèses générées dans les itérations antérieures seront ensuite combinées en utilisant le vote majoritaire pondéré. Une décision de classification est alors prise en se basant sur les sorties combinées d'hypothèses individuelles qui constituent l'hypothèse générée H_t.

- Notant que H_t choisit la classe qui reçoit le vote total le plus haut de toutes les hypothèses Calculer l'erreur E_t.

- Si $E_t > 1/2$, le h_t courant est abandonnée, un nouveau sous-ensemble d'apprentissage est sélectionné et une nouvelle h_t est générée

-Sinon normaliser l'erreur β_t

Les poids $w_t\ (i)$ sont ensuite mis à jour, pour calculer la prochaine distribution D_{t+1} qui sera utilisé pour sélectionner les prochains sous-ensembles d'apprentissage TR_{t+1} et de test $TE_{t+1.}$

La règle de mise à jour de la distribution $w_{t+1}\ (i)$ constitue le cœur de l'algorithme, parce qu'elle permet à Learn++ d'apprendre de façon incrémentale.

D'après cette règle, si l'exemple x_i est classé correctement par l'hypothèse générée H_t, son poids est diminué.

-Si x_i est mal classé, son poids de distribution est gardé inchangé.

Cette règle réduit la probabilité des exemples classés correctement d'être choisies dans TR_{t+1}, bien qu'elle augmente la probabilité des exemples mal classés pour être sélectionnés dans TR_{t+1}. Après que T_k hypothèses soient générées pour chaque base de données D_k. L'hypothèse finale H_{finale} générée par cet algorithme est obtenue par le vote majoritaire pondéré de toutes les hypothèses générées [POL 01].

Algorithme 2.4 (Learn++)

Entrée : Pour chaque base de données tirée de D_k, $k=1,2,\ldots\ldots\ldots,k$

-Séquence de m exemples d'apprentissage $S=[(x_1,y_1),(x_2,y_2),\ldots\ldots(x_m,y_m)]$

-Algorithme d'apprentissage faible *WeakLearn*.

- Entier T_k, spécifiant le nombre d'itérations.

pour $k=1,2,\ldots\ldots\ldots,k$ faire :

Initialiser $w1= D_1(i) = 1/m$, $\forall i$ a moins qu'il y ait connaissance antérieure pour sélectionner autrement.

Pour $t=1,2,\ldots T_k$ faire :

1. Déterminer $D_t = w_t / \sum_{i=1}^{m} w_t(i)$ pour que D_t est une distribution
2. Choisir aléatoirement les sous-ensembles d'apprentissage TR_t et de test TE_t selon D_t
3. Faire appel à *WeakLearn* en utilisant TR_t
4. Récupérer une hypothèse $h_t : X \rightarrow Y$,
5. -Calculer l'erreur de $h_t : \varepsilon_t = \sum_{i\,:ht(xi)\neq yi} D_t(i)$ sur $S_t = TR_t + TE_t$

-Si $\varepsilon_t > ½$ alors $t = t - 1$ et supprimer h_t et aller à l'étape 2

-Sinon normaliser l'erreur comme $\beta_t = \varepsilon_t / (1 - \varepsilon_t)$

6. Obtenir l'Hypothèse composée : $H_t = arg\,max_{y\in Y} \sum_{t\,:ht(x)=y} log(1/\beta t)$

Erreur composée $E_t = \sum_{i\,:ht(xi)\neq yi} D_t(i) = \sum_{i=1}^{m} D_t(i)\ [/H_t(x_i)\neq y_i/]$

-Si $E_t > 1/2$ alors $t=t-1$ supprimer H_t et aller à l'étape 2

$$w_{t+1}(i) = w_t(i) \times \{B_t, si\ H_t(x_i) = y_i\}$$

$$\{1, \quad autrement$$

$$= w_t(i) \times B_t^{1-[|H_t(x_i)\neq y_i|]}$$

7. Déterminer l'erreur composite normalisée : $\beta_t = E_t / (1-E_t)$

-Mettre à jour des poids des instances :

-Faire appel au vote majoritaire pondéré sur les hypothèses combinées H_t et produire l'hypothèse finale:

$$H_{final} = \arg\max_{y \in Y} \sum_{k=1}^{K} \sum_{t:H_t(x)=y} \log |(1/\beta_t)$$

2.9.5. LEARN++.MT

Dans les approches d'ensemble qui utilisent un mécanisme de vote pour combiner les sorties du classifieur, chaque classifieur vote sur la classe qu'il prédit. Le classement final est alors déterminé comme la classe qui reçoit le total de votes le plus élevé de tous les classifieurs. Learn ++ utilise le vote majoritaire pondéré, où chaque classifieur reçoit une pondération de vote basée sur ses performances d'apprentissage. Cela fonctionne bien dans la pratique pour la plupart des applications. Cependant, pour les problèmes d'apprentissage incrémental impliquant l'introduction de nouvelles classes, le schéma de vote s'avère injuste envers la classe nouvellement introduite: puisqu'un des classifieurs précédemment générés ne peut choisir la nouvelle classe, un nombre relativement important de nouveaux classifieurs qui reconnaissent le nouveau sont nécessaires pour que leur poids total puisse dépasser le vote du premier groupe de classifieurs sur les occurrences de la nouvelle classe. En retour, cela donne à l'ensemble un nombre inutilement grand de classifieurs.

Learn ++. MT [MUH 04] est spécialement conçue pour résoudre le problème de la propagation des classifieurs. La nouveauté de Learn ++. MT est le moyen par lequel les poids de vote sont déterminés. Learn ++. MT utilise également un ensemble de pondérations de vote basées sur les performances du classifieur. Toutefois, ces pondérations sont ensuite ajustées en fonction de la classification de l'instance spécifique au moment du test, par le biais du vote de pondération dynamique (Dynamic Weight Voting DWV). Pour toute instance de test donnée, Learn ++. MT compare les prédictions de classe de chaque classifieur et les compare aux classes sur lesquelles ils ont été formés. Si un ensemble suivant choisit massivement une classe qu'il a déjà vue, le poids de vote des classifieurs non formés à cette classe est proportionnellement réduit. Par exemple, supposons qu'un ensemble ait vu les classes 1 et 2 et un second ensemble, les classes 1, 2 et 3. Pour un exemple donné, si le second ensemble (formé sur la classe 3) choisit la classe 3, les classifieurs du premier ensemble (qui n'a pas vu la classe 3) réduit leur poids de vote proportionnellement à la confiance du second ensemble. En d'autres termes, lorsque l'algorithme détecte que les nouveaux classifieurs choisissent massivement une nouvelle classe sur laquelle ils ont été formés, les poids des autres classifieurs qui n'ont pas vu cette nouvelle classe sont réduits.

Muhlbaier et al. définit Learn ++. MT comme une modification de l'approche Learn ++ où les pondérations sont mises à jour de manière dynamique. La mise à jour dynamique du poids réduit l'effet de la sur-vote des nouvelles classes, comme le montre Learn ++, où les nouvelles classes ont une précision de classification très faible. Il utilise la technique selon laquelle si un ensemble choisit massivement une classe qu'il a vue auparavant, les poids des ensembles qui n'ont pas vu la classe sont réduits [ERD 05]. Cette modification montre de meilleures précisions que celles de la norme Learn ++.

Algorithme 2.5 (Learn++.MT)

Entrée : Pour chaque base de données tirée de D_k, $k = 1,2, \ldots\ldots, k$

-Séquence de $i=1\ldots..m_k$ instances x_i avec des libellés $y_i \in Y_k = \{1\ldots c\}$

-Algorithme d'apprentissage faible *BaseClassifier*.

Entier T_k, spécifiant le nombre d'itérations.

pour $k=1,2,\ldots\ldots k$ faire :

Si $k=1$, Initialiser $w_1 = D_1(i) = 1/m$, $eT_1 = 0\ \forall i$,

Sinon Aller à l'étape 5 pour évaluer l'ensemble actuel sur les nouvelles données D_k, mettre à jour les poids, et rappeler le nombre actuel des classifieurs→ $eT_k = \sum_{j=1}^{k-1} T_j$

Pour $t = eT_k + 1, eT_k + 2, \ldots\ldots eT_k + T_k$ faire :

Déterminer $D_t = w_t / \sum_{i=1}^{m} w_t(i)$ pour que D_t est une distribution
Faire appel à *BaseClassifier* avec un sous ensemble de D_k choisi aléatoirement en utilisant D_t ;
Récupérer une hypothèse $h_t : X \rightarrow Y$, et
Calculer l'erreur de $h_t : \varepsilon_t = \sum_{i\,:ht(xi)\neq yi} D_t(i)$
Si $\varepsilon_t > ½$ alors supprimer h_t et aller à l'étape 2
Sinon normaliser l'erreur comme $\beta_t = \varepsilon_t / (1 - \varepsilon_t)$

4. $CTr_t = Y_k$ pour sauvegarder les libellés des classes utilisées pour l'apprentissage h_t.
5. Faire appel à *DWV* pour obtenir l'hypothèse composée H_t.
6. Calculer l'erreur de l'hypothèse composée $E_t = \sum_{i\,:ht(xi)\neq yi} D_t(i)$
7. Déterminer $B_t = E_t/(1 - E_t)$, et mettre à jour les poids des instances:

$$w_{t+1}(i) = \begin{cases} w_t(i) \times \{B_t, si\ H_t(x_i) = y_i\} \\ \{1, \quad \text{autrement} \end{cases}$$

Faire appel à *DWV* pour obtenir l'hypothèse finale H_t

Algorithme 2.6 (DWV: Dynamic Weight Voting)

Entrée : Pour, $i = 1,2, \ldots\ldots, n$ les instances d'apprentissage, ou l'instance de test x_i

-Les classifieurs h_t

-Les valeurs d'erreur correspondante β_t

-Les classes CTr_t utilisées dans l'apprentissage de h_t

Entier T_k, spécifiant le nombre d'itérations.

pour $t=1,2,\ldots\ldots T$ où T est le nombre total des classifieurs faire :

1. Initialiser les poids de classifieur $W_t = \log\left(\frac{1}{\beta_t}\right)$
2. Créer le facteur de normalisation Z, pour chaque classe

   $Z_c = \sum_{t:c\in CTr_t} W_t$, for $c=1,\ldots\ldots,C$ classes
3. Définir la décision préliminaire $P_c = \sum_{t:ht,(xi)=c} W_t / Z_c$, for $c=1,\ldots\ldots,C$

4. Mettre à jour les poids de vote $W_{:c\notin CTr_t} = W_{:c\notin CTr_t}(1 - P_c)$, for $c=1,\ldots\ldots,C$
5. Calculer l'hypothèse finale/ composée

$$H_{final}(x_i) = \arg\max_{c} \sum_{t:ht,(xi)=c} W_t$$

2.9.6. Machine à vecteurs support (SVM) incrémentale

De nombreuses versions SVM ont été développées, nous mentionnons, dans cette section, certains de ces travaux:

Le premier travail a été réalisé par Syed et al. [SYE 99] qui ont proposé d'apprendre de nouvelles données en conservant les vecteurs de support identifiés au cours des premières phases d'apprentissage sans conserver les données elles-mêmes puisque ces derniers résument parfaitement les données déjà apprises.

Ruping [RUP 01] propose un algorithme SVM incrémental qui vise à gérer tous les vecteurs supports de manière incrémentale. Lorsqu'une nouvelle donnée d'apprentissage est mal classée ou à l'intérieur de la marge, le séparateur est recalculé en fonction du vecteur de support et de ce nouveau point.

Ralaivola et dAlché Buc [RAL 01] proposent une version incrémentale de SVM. Ils considèrent SVM comme une combinaison d'experts, chaque expert étant un vecteur de support. Le noyau utilisé est un noyau gaussien ou un autre noyau basé sur la notion de voisinage, la zone d'influence d'un vecteur de support n'est significative que dans un voisinage proche de lui. Lors de l'arrivée d'une nouvelle donnée, il n'est pas nécessaire d'interroger l'ensemble de tous les vecteurs de support, mais uniquement ceux dont l'influence est la plus forte dans un autre exemple.

En 2001, Cauwenberghs et Poggio [CAU 01] ont conçu un algorithme en ligne exact pour l'apprentissage incrémental de SVM, qui met à jour les paramètres de la fonction de décision lors de l'ajout ou de la suppression d'un vecteur à la fois (nous allons le détailler dans le chapitre suivant).

Un SVM proximal (PSVM) proposé par Fung et Mangasarian [FUN 02]. L'idée de base est d'utiliser des exemples de bordure du PSVM hyper-plan autour de cet hyperplan. À l'arrivée d'un nouvel exemple, les auteurs calculent la distance entre l'hyperplan et la nouvelle frontière. Si elle est proche, elle sera ajoutée (ajout de nouveaux exemples) et supprimera les anciens exemples.

En 2003, Diehl et Cauwenberghs [DIE 03] améliorent le travail précédent de Cauwenberghs et al. [CAU 01] et présentent un cadre pour l'apprentissage, l'adaptation et l'optimisation exacts des SVM, afin de simplifier la sélection du modèle en perturbant la solution SVM lors de la modification des paramètres du noyau et lors de la régularisation.

Loosli [LOO 06] montre que SVM a la capacité de gérer les grandes bases de données en ligne. Cette étude est basée sur l'enrichissement des bases initiales d'apprentissage en introduisant des modèles de variabilité appliqués à ces bases. Le SVM apprenne de nouveaux exemples en suivant les mêmes distributions que les exemples déjà appris, car les modifications introduites par la nouvelle instance sont locales.

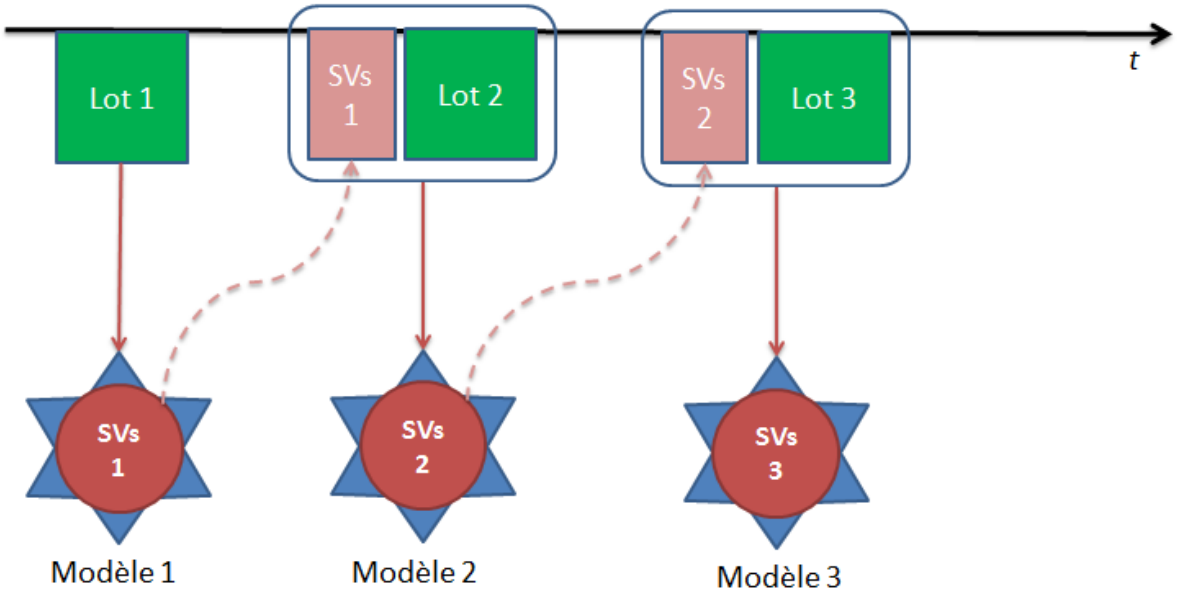

Figure 2.5. Apprentissage incrémental avec la méthode de Syed [NGO 15]

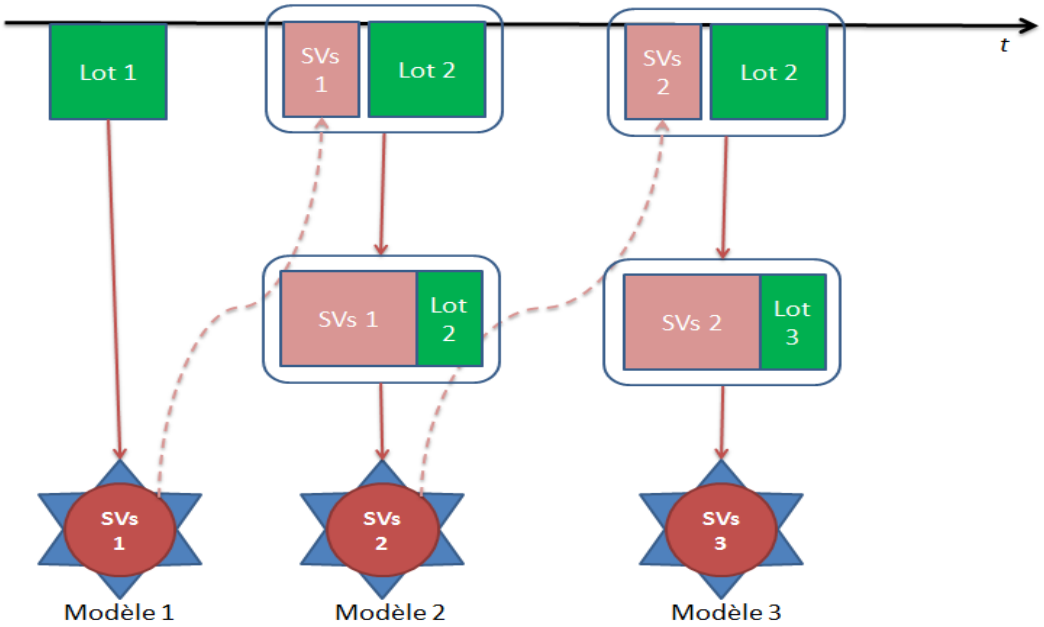

Figure 2.6. Illustration de la méthode de Rüping [NGO 15]

2.9.7. Arbre de décision incrémental

Les arbres de décision sont semi-liés à l'apprentissage incrémental. Ils sont construits progressivement en utilisant une procédure récursive (ID3, C4.5 ...). Une arborescence est construite de manière incrémentale à partir d'un nœud racine, mais lors de l'insertion ou de la suppression d'un seul exemple d'apprentissage, l'arbre devrait être reconstruire à zéro, car la procédure (par exemple, ID3) prend en compte tous les exemples d'apprentissage à chaque itération pour calculer la distance et l'entropie de chaque attribut.

Les algorithmes d'arbres de décision incrémentaux les plus connus sont les suivants:

- ID4 est le premier algorithme proposé par Schlimmer et Fisher [SCH 86]. Il incorpore les données progressivement. Cependant, certains concepts étaient incompréhensibles car ID4 ignorait les sous-arbres lorsqu'un nouveau test était choisi pour un nœud;
- ID5 développé par Utgoff [UTG 89], il ne supprime pas les sous-arbres, mais ne garantit pas non plus qu'il produira le même arbre qu'ID3;
- ID5R est un algorithme qui effectue l'apprentissage par induction à partir d'exemples de manière incrémentale. Cette méthode génère le même arbre de décision trouvé par ID3 pour le même ensemble d'apprentissage [SAL 10];
- ITI (Incremental Tree Induction) est une méthode efficace pour induire progressivement des arbres de décision. Le même arbre est produit pour le jeu de données d'apprentissage. La méthode consiste à stocker les statistiques dans les feuilles, ce qui permet une restructuration de l'arbre lors de l'arrivée de nouveaux exemples [SAL 10];
- Le VFDT (Very Fast Decision Tree) arbre de décision très rapide basé sur le calcul du gain d'information pour la sélection des attributs. L'apprenant VFDT réduit le temps d'apprentissage pour les grands ensembles de données incrémentales en sous-échantillonnant le flux de données entrant [SAL 10].

Nous avons choisi de détailler ID5R, l'un des arbres de décision incrémentaux les plus utilisés. La raison du choix de l'ID5R est sa flexibilité intégrée en ce qui concerne le niveau de granularité et sa robustesse.

La structure de l'arbre induite par ID5R contient [UTG 89]:

- Un nœud feuille (ou nœud de réponse) qui contient: un nom de classe et l'ensemble des descriptions d'instance sur le nœud appartenant à la classe;
- Un nœud (ou nœud de décision) non-feuille qui contient: un test d'attribut, avec une branche vers un autre arbre de décision pour chaque valeur possible de l'attribut, les comptes positifs et négatifs pour chaque valeur possible de l'attribut et l'ensemble des éléments suivants: attributs non-test sur le nœud, chacun avec des comptes positifs et négatifs pour chaque valeur possible de l'attribut.

La différence essentielle entre ID5R et ID3 réside dans le fait que dans ID5R, les exemples sont conservés dans l'arborescence. Ceci permet d'obtenir une capacité de généralisation sans perdre la spécificité des exemples.

L'algorithme d'apprentissage utilisé par ID5R est le suivant:

Algorithme 2.7 (ID5R)

- Si l'arbre est vide, le définir comme une forme non expansée, en définissant le nom de la classe sur la classe de l'instance et l'ensemble des instances sur le singleton défini contenant l'instance;
- Sinon, si l'arborescence est sous forme non développée et que l'instance appartient à la même classe, ajouter l'instance à l'ensemble d'instances conservées dans le nœud;
- autrement;
- Si l'arborescence est sous forme non développée, développer-la à un niveau, en choisissant l'attribut de test pour la racine de manière arbitraire;
- Pour l'attribut test et tous les attributs non test au nœud actuel, mettre à jour le nombre d'instances positives ou négatives pour la valeur de cet attribut dans l'instance d'apprentissage;
- Si le nœud actuel contient un test d'attribut qui n'a pas le score E le plus bas, alors
- Restructurer l'arbre par un attribut avec le plus bas score

- E-score sera à la racine;
- Répéter de manière récursive l'attribut de test dans chaque sous-arbre (excepté celui qui sera mis à jour à l'étape 3 et mis à jour récursivement);
- Mettre à jour de manière récursive l'arbre de décision situé sous le nœud de décision actuel le long de la branche pour la valeur de l'attribut test apparaissant dans la description de l'instance. Exploiter la branche si nécessaire.

2.9.8. Méthode AttributeNets

AttributeNets [WU 07] est une méthode de classification basée sur une structure de données appelée Graphe de Concepts (GC) qui a été développée par Enembreck [ENE 05] pour un apprentissage incrémental. GC est composé de plusieurs couches d'attributs représentant chacune un attribut et d'une couche de classification représentant les catégories. La couche d'attribut est composée de plusieurs nœuds d'attribut mappant les valeurs de cet attribut et la couche de classe comprend certains nœuds de classification, chacun mappant à une catégorie. Pendant la phase d'apprentissage, GC enregistre tous les cas en associant le numéro de séquence du cas au nœud correspondant lorsque la valeur de l'attribut est égale à la valeur du nœud pour chaque couche d'attributs et au nœud de classification auquel le cas appartient. Ensuite, ils ont utilisé une méthode basée sur l'entropie appelée ELA pour utiliser les informations stockées dans GC pour la classification. ELA marque une étiquette sur le cas non étiqueté de la même façon que les cas les plus similaires.

AttributeNet a les avantages suivants:

1. C'est en lui même un classifieur multi-catégories en raison de la structure multi-réseaux;
2. Il est remarquable en termes d'utilisation de la mémoire et de rapidité d'adaptation, ce qui est d'une importance vitale pour l'apprentissage incrémental, en particulier l'apprentissage en ligne;
3. Les résultats de la classification sont faciles à comprendre;
4. Il est robuste avec les données bruitées et la rareté des cas d'apprentissage.

2.9.8.1. La structure AttributeNets

La structure AttributeNets fait référence à la combinaison de structures isomorphiques individuelles appelées AttributeNet. Pour chaque catégorie, une AttributeNets est construite, et est composé de plusieurs couches d'attributs comprenant des nœuds d'attributs. De même, chaque couche correspond à un attribut spécifique d'observations et un nœud de la couche correspond à une valeur spécifique de cet attribut. Cependant, il existe deux différences significatives entre GC et AttributeNet: premièrement, AttributeNet ne possède pas de couche de classification, chaque AttributNet ne faisant simplement référence qu'à une seule catégorie; deuxièmement, au lieu d'attacher le numéro de séquence de cas à chaque nœud, les informations statistiques sont enregistrées dans AttributeNet. Chaque nœud conserve un compteur (degré de nœud) pour enregistrer le nombre de cas appartenant à ce nœud; pour n'importe lequel des deux nœuds, un autre compteur (degré de liaison) est conservé, enregistrant le nombre de cas appartenant aux deux nœuds.

La méthode AttributeNets est basée sur deux algorithmes différents: un algorithme d'apprentissage et un algorithme de classification.

2.9.8.2. L'algorithme d'apprentissage d'AttributeNets

Le processus d'apprentissage d'AttributeNets est simple et efficace en termes de complexité temporelle, ce qui rend cette méthode adaptée à l'apprentissage en ligne. AttributeNets mémorise les informations statistiques relatives aux valeurs d'attributs et aux relations entre deux valeurs d'attributs différents, en prenant en compte les cas de la seule catégorie du réseau.

Algorithme 2.8 (Apprentissage d'AttributeNets)

Entrée: AttributeNets ($Attri, 1 \leq i \leq ||Catégories||$) à être mis à jour, nouveau cas d'apprentissage (Cas)

Sortie : AttributeNets mis à jour

Etape1 : $i = catégorieDe\ (Cas)$

Etape2 : Pour $1 \leq j \leq ||Couches||$ $degré\_nœud\ [j][k]++$ ($degré\_nœud\ [j][k]$ est le degré du nœud j_k qui est un nœud de la couche j de $Attri$ et est activé par Cas)

Etape3 : Pour $1 \leq j \leq ||Couches||$ Pour $1 \leq u \leq ||Couches||$ $degré\_nœud\ [j][k][u][v]++$ ($degré\_nœud\ [j][k][u][v]$ est le degré du lien du nœud entre les nœuds activés i.e. nœud j_k de la couche j et nœud uv de la couche u de $Attri$)

Etape4 : Fin

Lorsqu'un cas d'apprentissage de la catégorie i arrive, AttributeNets est activé alors que d'autres réseaux autres que la catégorie i sont simplement ignorés par ce cas. Avec AttributeNets, pour chaque attribut du cas, c'est-à-dire chaque couche d'AttributeNets, le degré de nœud augmente si cet attribut a une valeur identique à la valeur du nœud. Pour deux nœuds quelconques de couches différentes, le degré de liaison entre ces deux nœuds augmente de 1 si les deux nœuds sont activés par la casse.

2.9.8.3. L'algorithme de classification d'AttributeNets

Le processus d'apprentissage et le processus de classification pourraient être liés dans la méthode AttributeNets. Cette capacité est favorable dans le scénario d'apprentissage en ligne. Dans cette section, un algorithme de classification basé sur AttributeNets est donné.

Algorithme 2.9 (Classification d'AttributeNets)

Entrée : AttributeNets ($Attri, 1 \leq i \leq ||Catégories||$) appris, nouveau cas (Cas) avec sa catégorie inconnue

Sortie : Catégorie c du Cas

Etape1 : Pour $1 \leq i \leq ||Catégories||$ $ri = 1$

Etape2 : Pour $1 \leq i \leq ||Catégories||$

Pour $1 \leq j \leq ||Couches||$

$ri = ri \times degré\_noeud\ [i][j] + \Delta$

($degré\_nœud\ [i][j]$ est la valeur du nœud activé par Cas dans la couche j de Attri, Δ est un petit nombre empêchant r_i d'être 0)

Etape3 : Pour $1 \leq i \leq ||Catégories||$

Pour $1 \leq j \leq ||Couches||$

Pour $1 \leq k \leq ||Couches||$

$r_i = r_i \times (degrélien\ i\ j\ k + \Delta)$

(degré_lien [i][j][k] est la valeur du nœud lien entre les nœuds activés de la couche j et la couche k de $Attri$, Δ est un petit ajustement empêchant r_i d'être 0)

Etape4 : Retourner i qui Maximize (r_i)

2.9.9. Perceptron Auto-Organisé PAO (SOP : Self-Organizing Perceptron)

Le Perceptron -Auto-Organisé (PAO) [HEB 99] est un nouveau réseau de neurones qui partage des caractéristiques communes avec le PMC et le RBF. C'est un réseau multicouche où l'information se propage de la couche d'entrée vers la couche de sortie. Il possède une architecture hybride qui permet de partitionner les neurones d'un PMC en groupes de neurones grâce à la liaison de l'ensemble $C = \{c_1, c_2, ...c_q\}$ d'une carte auto-organisée préalablement entraînée aux neurones cachés d'un PMC. Ce partitionnement est réalisé en reliant chaque neurone de la carte auto-organisatrice à un neurone de la dernière couche cachée du PMC, tel qu'illustré à figure 2.7. Ce lien entre les deux réseaux nécessite que le nombre de neurones au sein de la carte auto-organisée corresponde exactement au nombre de neurones situés sur la couche cachée du PMC. Le PMC et la carte auto-organisatrice sont alimentés par le même vecteur d'entrée. Le nombre de neurones de sortie du PMC doit être connu dès le début de l'apprentissage c.à.d. le nombre maximal de classes doit être fixé.

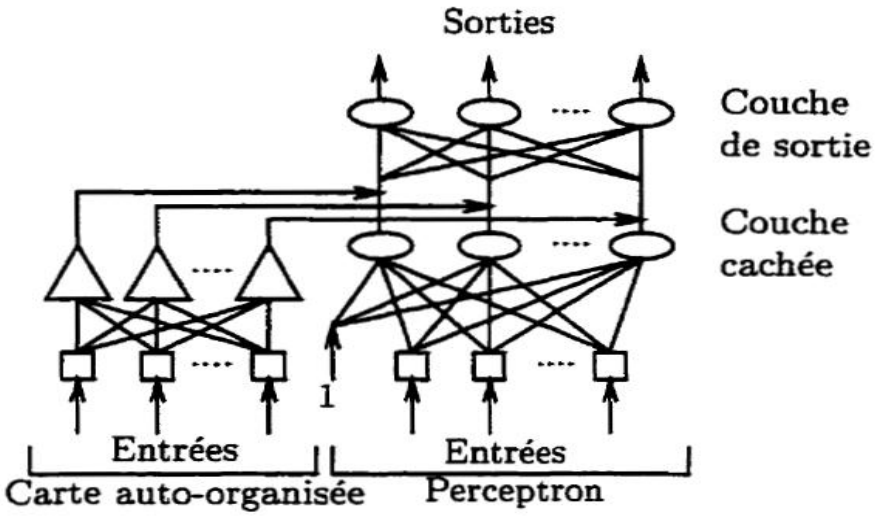

Figure 2.7. Architecture hybride de PAO [HEB 00]

La tâche générale pour laquelle le réseau doit être entraîné se divise en sous tâches. Chacune d'elles étant individuellement plus simple à résoudre.

Le PAO peut fonctionner selon deux modes d'apprentissage :

- **L'apprentissage passif:** il contient deux phases successives. Dans la première phase, la carte auto-organisée est entraînée, puis l'apprentissage du PMC s'effectue selon l'algorithme de rétro-propagation des erreurs dans la deuxième phase;
- **L'apprentissage incrémental:** le partitionnement dynamique de tâches permet au réseau d'apprendre de façon incrémentale.

Il est possible d'affiner le taux de reconnaissance en faisant réapprendre les classes et les données qui ont été mal apprises grâce à ce réseau incrémental. La stratégie d'apprentissage incrémental du PAO repose sur la définition de plusieurs ensembles de données de vigilance locales qui visent à limiter l'influence que pourrait avoir une nouvelle donnée sur le PAO et permettent de mettre à jour l'état des connaissances de ce dernier en fonction de l'évolution temporelle du problème posé. Grâce à la spécialisation des neurones cachés du réseau PAO, il est possible d'assigner chaque donnée à une zone (ou une branche) bien précise du réseau PAO, d'où l'appellation de données de vigilance locales. L'apprentissage du perceptron auto-organisé par une nouvelle donnée x consiste à déterminer quel est le neurone gagnant n_1 de la carte auto-organisée. Ensuite, il faut déterminer N' l'ensemble des neurones voisins à n_1, ceux fortement activés par la donnée x, qui sont déterminés selon :

$$N'=\{nj\in N/||x-wj||\leq\delta\times\sigma j\}.....(2.1)$$

δ est un paramètre utilisé pour pondérer la sphère d'influence σ_j de chaque neurone n_j. Tous les ensembles de vigilance associés au neurone gagnant doivent être récupéré ainsi qu'aux neurones appartenant à l'ensemble de neurones voisins. Une nouvelle phase d'apprentissage du PAO est ensuite réalisée sur un ensemble constitué de toutes ces données et de la donnée x. On vérifie que celle-ci soit correctement apprise:

Si c'est le cas, on ajoute x à l'ensemble de vigilance de n_1 et l'apprentissage de cette donnée est terminé;

Sinon, s'il existe un nouveau neurone de la carte auto-organisée qui est activé par la donnée x alors on déplace ce neurone vers la donnée x;

Si un tel neurone n'existe pas et que le neurone gagnant est activé par la donnée x, un nouveau neurone est créé avec comme position dans l'espace de représentation celle de la donnée x. Ce nouveau neurone est ensuite connecté au neurone n_1. Si le neurone gagnant n'est pas activé par la donnée x, alors deux nouveaux neurones sont créés et sont connectés entre eux. Un de ces neurones a la position de la donnée x et l'autre a pour position :

$$wnew2=w1+(x-w1) \quad .....(2.2)$$

Les données de vigilance sont remises à jour et le réseau est entraîné à nouveau, jusqu'à ce que la donnée x soit bien apprise.

2.9.10. Apprentissage incrémental à l'aide d'un algorithme génétique (ILUGA)

ILUGA [HUL 07] (Incremental Learning Using Genetic Algorithm) utilise des classifieurs SVM puissants et optimisés en utilisant les algorithmes génétiques pour trouver l'hyperplan de séparation optimal. Ceci est fait en trouvant le meilleur noyau et la meilleure marge. Les poids de vote sont ensuite générés par l'AG en utilisant les classifieurs forts. ILUGA applique le mécanisme de vote utilisé dans de nombreuses approches d'ensembles où les classifieurs votent pour la classe qu'ils prédisent [POL 01], à l'exception du fait que le vote est pondéré. Par conséquent, un système de vote majoritaire pondéré sera utilisé. Les pondérations envoyées au

système de vote majoritaire pondéré sont des pondérations individuelles dépendant de la décision du classifieur. ILUGA permet de former autant de classifieurs que nécessaire et les données d'apprentissage de chaque nouveau classifieur formé sont randomisées de sorte que les sections utilisées respectivement pour l'apprentissage et la validation soient toujours différentes, ce qui donne aux classifieurs une nouvelle hypothèse sur les données.

2.9.10.1. Première phase d'apprentissage

La première étape de l'apprentissage pour ILUGA est la construction d'un ensemble de classifieurs forts. Cet ensemble est constitué de classifieurs SVM binaires permettant de classer le jeu de données multi-classes. Les données d'apprentissage sont séparées de manière aléatoire en trois sections, à savoir l'apprentissage des classifieurs (Train) et des deux ensembles de validation (Val1, Val2). Les classifieurs binaires sont formés à l'aide de l'ensemble Train et sont ensuite développés en utilisant l'AG pour trouver les paramètres optimaux pour le SVM. Les variables à optimiser sont les fonctions du noyau (quadratique, fonction à base radiale, tangente polynomiale et hyperbolique) et la marge souple. L'évolution des classifieurs est effectuée à l'aide des résultats de validation de l'ensemble Val1 afin de déterminer l'aptitude de chacun des chromosomes. C'est la première étape d'apprentissage.

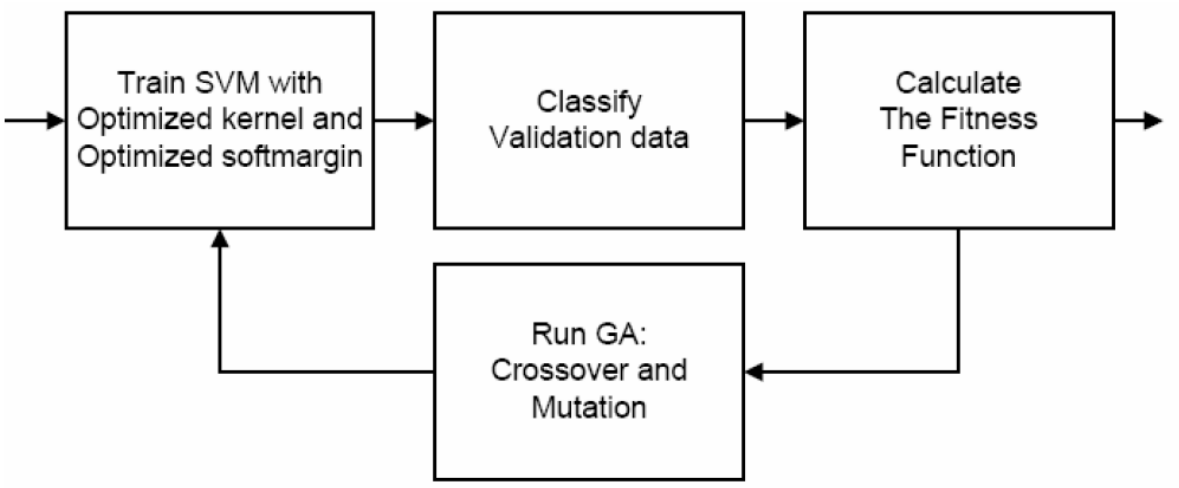

Figure 2.8. L'optimisation du SVM en utilisant un algorithme génétique [HUL 07]

2.9.10.2. Deuxième phase d'apprentissage

La deuxième étape de l'apprentissage pour ILUGA utilise un ensemble de classifieurs binaires forts créés lors de la première étape. Chaque décision des classifieurs binaires se voit attribuer une pondération telle que chaque classifieur binaire a deux pondérations: une pour chaque décision. Les poids sont ensuite développés en utilisant les AG, qui sont évalués en utilisant la fonction fitness. La fonction de mise en forme évalue la forme des chromosomes (poids) par le pourcentage correctement classé en utilisant le vote majoritaire pondéré sur l'ensemble Val2. L'optimisation des pondérations individuelles rend la classification très robuste, éliminant ainsi les décisions mal classées et donnant une pondération importante aux décisions correctement classées. Cela permet également de classer correctement les nouvelles classes telles qu'elles sont vues.

2.9.10.3. Le vote

Le vote est effectué pour déterminer la classe en cours de classement. Ceci est fait en utilisant l'ensemble des classifieurs et leurs poids correspondants. Tout d'abord, le nombre de votes

potentiels pour chaque classe est calculé, puis les poids de toutes les classes sont divisés par leur nombre de votes potentiels. Cela donne aux nouvelles classes qui ont été introduites le même pouvoir de vote que les classes vues auparavant. Les pondérations entrent ensuite dans un vote majoritaire pondéré, comme le montre la figure 2.9, où chaque classifieur binaire va au vote majoritaire pondéré et le vote le plus élevé correspond à la classe prédite.

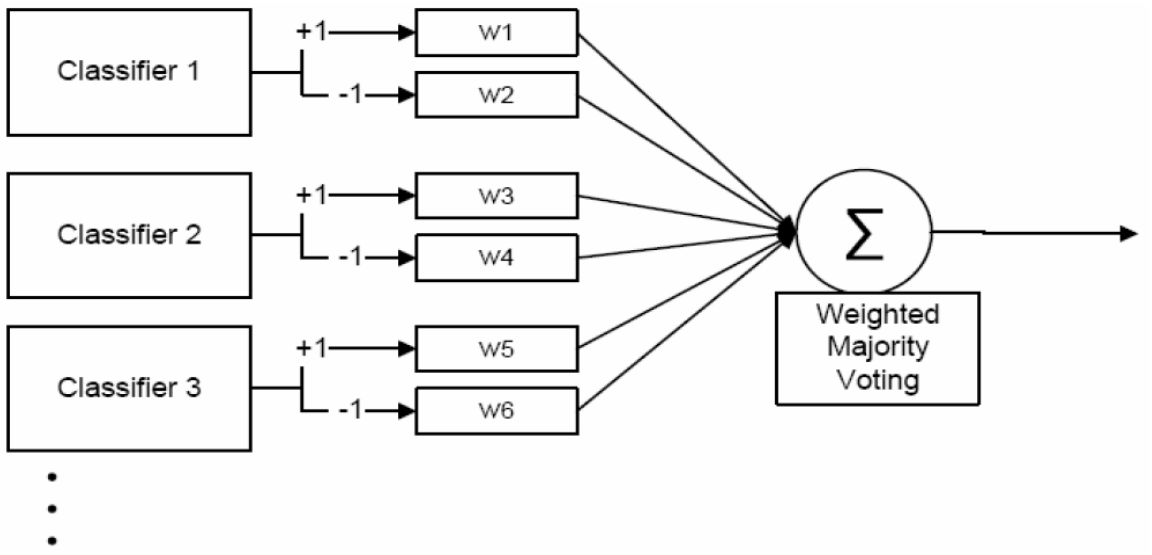

Figure 2.9. Le vote majoritaire [HUL 07]

2.9.10.4. Ajout des nouvelles données incrémentales

Lorsque ILUGA doit être formé de manière incrémentale, il a besoin des informations des classifieurs précédents, des poids et de ce que chacun des classifieurs classifie à partir de la phase d'apprentissage précédente. Le système passe ensuite aux étapes un et deux de l'apprentissage pour former les nouveaux classifieurs et poids, qui sont ensuite ajoutés à l'ensemble. Tous les classifieurs entrent ensuite dans le vote.

Algorithme 2.10 (ILUGA)

Entrées:

- Classifieurs binaires précédents
- Poids précédents
- Classes classées par chaque classifieur binaire
- Nombre de classifieurs à former (m)

Faites pour $k = 1$ à m

1. Entraîner le classifieur fort en utilisant le jeu *Train*, puis optimiser les paramètres en utilisant AG avec l'ensemble *Val1*

2. Optimiser les poids de vote pour chaque décision binaire du classifieur en utilisant AG sur l'ensemble *Val2*

3. Ajouter les classifieurs et les poids à l'ensemble

4. Pour la classification, le nombre de votes potentiels pour chaque classe est calculé, puis les poids de toutes les classes sont divisés par leur nombre de votes potentiels. Le vote se fait à la majorité pondérée pour arriver à la classe prédite.

2.9.11. Algorithme d'apprentissage incrémental pour le classifieur de réseau hybride RBF-BP (ILRBF-BP)

ILRBF-BP [WEN 16] est une méthode d'apprentissage incrémental pour la construction de neurones cachés RBF. Elle peut générer progressivement des neurones cachés et estimer efficacement le centre et le nombre de neurones cachés en déterminant la densité de différentes régions de l'espace échantillon d'apprentissage. Une architecture de réseau hybride RBF-BP est conçue pour former les poids de sortie. La sortie de la couche cachée RBF d'origine est traitée et connectée à un réseau à plusieurs couches de perceptron (PMC); ensuite, un algorithme de rétropropagation est utilisé pour mettre à jour les poids PMC. Les neurones cachés RBF sont utilisés pour le mappage non linéaire du noyau et le réseau BP est ensuite utilisé pour la classification non linéaire, ce qui améliore encore les performances de la classification.

Dans un réseau RBF typique, les poids de sortie sont généralement estimés par un algorithme linéaire des moindres carrés, tel que l'algorithme LMS (Least Mean Squares) ou RLS (Recursive Least Squares). L'algorithme des moindres carrés linéaires doit être transformé en un algorithme non linéaire qui peut fournir une meilleure surface de classification pour adapter l'espace d'échantillonnage. À cette fin, une architecture de réseau hybride RBF-BP est conçue. La sortie des neurones cachés RBF est traitée et connectée à un réseau PMC, puis l'algorithme non linéaire BP est utilisé pour mettre à jour les pondérations des PMC. L'architecture du réseau hybride RBF-BP est illustrée par la figure 2.10, qui comprend quatre composants:

1. La couche d'entrée, constituée de nœuds sources, où k est la dimension du vecteur d'entrée;

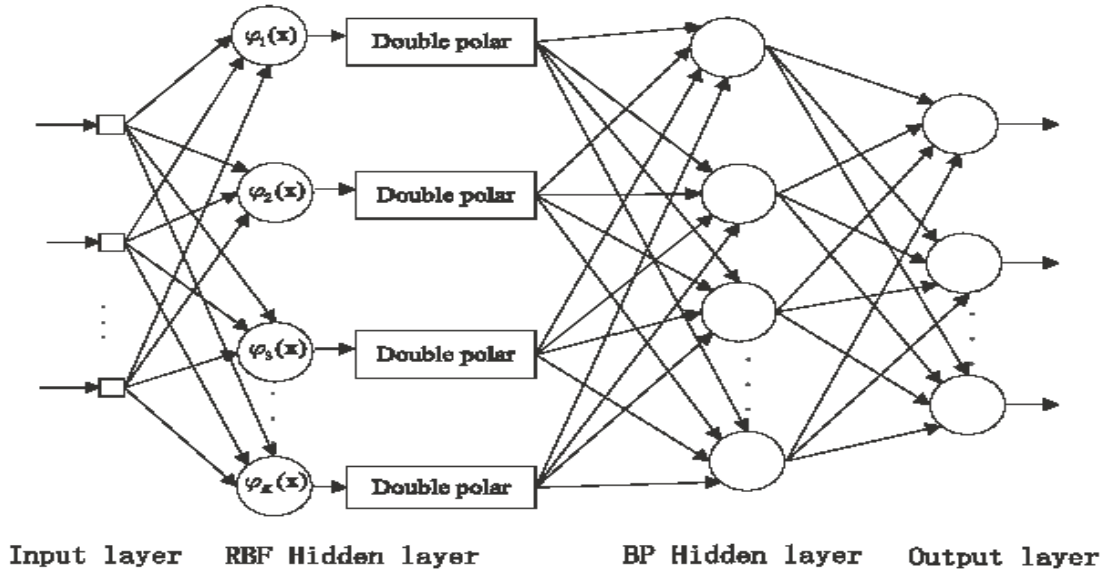

Figure.2.10. Architecture hybride RBF-BP [WEN 16]

2. La couche cachée RBF, qui consiste en un groupe de fonctions des noyaux gaussiennes:

$$\varphi_k(x) = \exp\left(-\frac{1}{2\sigma_k^2}\|x-\mu_k\|2\right) k = 1,2,\ldots\ldots,K \quad \text{.....(2.3)}$$

Où μ_k et σ_k sont le centre et la largeur du neurone caché, respectivement, et K le nombre de neurones cachés.

3. La couche masquée BP, constituée des neurones situés entre la couche masquée RBF et une couche de sortie. Le champ local induit $v_j^{(l)}$pour le neurone j dans la couche l du réseau BP est:

$$v_j^{(l)} = \sum_i w_{jl}^{(l)} y_i^{(l-1)} \text{ .....(2.4)}$$

Où $y_i^{(l-1)}$est le signal de sortie du neurone i dans la couche précédente l-1 du réseau BP et $w_{jl}^{(l)}$ est le poids synaptique du neurone j dans la couche l qui est alimenté par le neurone i de la couche 1-1. En supposant l'utilisation d'une fonction sigmoïde, le signal de sortie du neurone j dans la couche $l$est:

$$y_j^{(0)} = \varphi_j(v_j) = a\tanh(bv_j) \text{.....(2.5)}$$

Où a et b sont des constantes;

Si le neurone j est dans la première couche cachée du réseau BP, c.-à-d., $L = 1$, on définit:

$$y_j^{(0)} = g_j(x) \text{.....(2.6)}$$

Où $g_j(x)$ est la sortie double polaire de $\varphi_j(x)$et peut être notée:

$$g_j(x) = 2.\varphi_j(x) - 1 \text{.....(2.7)}$$

4. La couche de sortie. Définir L comme étant la profondeur du réseau BP. Noter que la profondeur du réseau est égale à la somme de la couche en entrée du réseau, de la couche cachée et de la couche en sortie, c'est-à-dire que si $l = 1$, alors $L = 3$ et la sortie peut être donnée comme:

$$o_j = y_j^{(l)} \text{.....(2.8)}$$

Dans la figure 2.10, le traitement à double polaire peut assurer la validité de l'entrée du réseau BP. L'architecture réseau hybride RBF-BP est conçue pour que le réseau RBF présente une bonne stabilité, où la réponse d'activation dans les neurones cachés RBF présente des caractéristiques locales et mappe la valeur de sortie entre 0 et 1. Ainsi, les échantillons originaux, y compris les valeurs éloignées, être limité à un espace fini. Lorsque les résultats de la cartographie des neurones cachés RBF sont traités et utilisés pour l'entrée du réseau BP, le taux de convergence de l'algorithme BP peut être augmenté et les minima locaux évités. Pour un réseau BP, la réponse d'activation dans les neurones cachés a des caractéristiques globales, en particulier dans les régions qui ne sont pas entièrement affichées dans l'ensemble d'apprentissage. Par conséquent, l'architecture de réseau hybride RBF-BP constitue un modèle raisonnable, et elle fournit une nouvelle stratégie qui combine les caractéristiques locales du réseau RBF avec les caractéristiques globales du réseau BP. En outre, le réseau hybride simplifie le nombre de neurones dans la couche cachée de la couche BP tout en réduisant davantage la dépendance vis-à-vis de mappage de l'espace dans la couche cachée RBF. Un réseau de neurones PMC à couche masquée avec un mappage entrée-sortie peut fournir une réalisation approximative de toute cartographie continue. Combinée à la discussion ci-dessus, dans le réseau hybride, le nombre de couches cachées du réseau BP est défini comme étant $l = 1$.

2.9.12. IGNG: Incremental Growing Neural Gas

IGNG [PRU 06] est une extension de l'algorithme GNG (Incremental Growing Neural Gas) proposé dans [FRI 95]. Les deux algorithmes sont constructifs et permettent l'extraction de la topologie des données au sens non supervisé. Ils effectuent une construction incrémentale d'un graphe avec une structure et une dimension variable dans l'espace de représentation. L'ajout de nœuds dans GNG s'effectue à chaque fois qu'un nombre fixe de données sont présentées, par contre dans IGNG, les nœuds sont créés uniquement en cas de besoin, et non à étapes régulières.

Le modèle de données dans IGNG est lié à chaque nœud qu'il représente et la continuité de données entre deux nœuds est représentée par un arc. Chaque modèle est associé par une valeur qui représente la probabilité d'existence et une position dans l'espace de représentation. Le modèle doit représenter au mieux les données qui s'y projettent, ce modèle prend plusieurs formes comme par exemple une gaussienne dans le cas de données numériques.

Figure 2.11. Données d'apprentissage

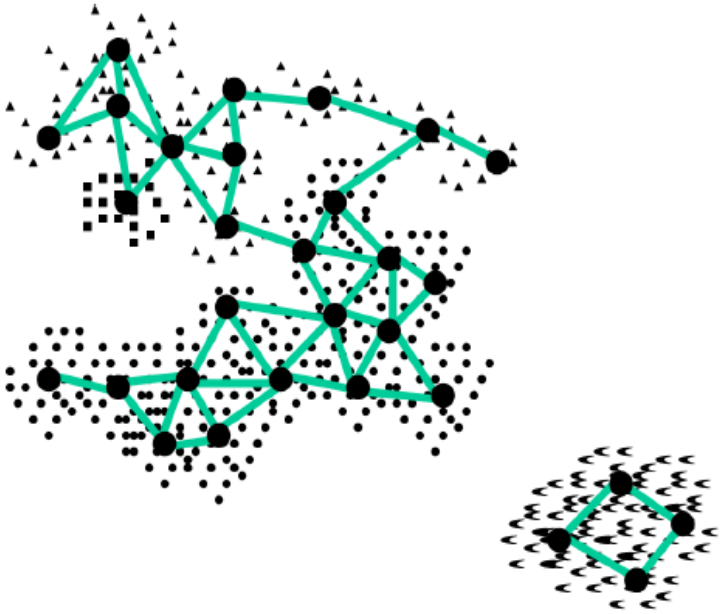

Figure 2.12. Graphe obtenu après l'apprentissage

La figure 2.12 illustre le graphe obtenu (au sens non supervisé) sur un petit ensemble de données d'apprentissage.

Les capacités de modélisation de la topologie des données dans l'espace de représentation sont reflétées sous forme d'un graphe d'une part et le caractère incrémental de l'algorithme d'autre part. Quand de nouvelles données sont ajoutées, de nouveaux nœuds sont créés dans le graphe. S'il y a suffisamment de données qui activent les nœuds créés alors ils seront confirmés. Sinon, ces nœuds sont retenus dans un état "provisoire" dans l'attente de données supplémentaires pour confirmer ou infirmer leur existence. La continuité entre deux nœuds est

traduite par un arc, ce dernier peut être détruit lors de l'apparition des nouvelles données et adapter le graphe à la position des nœuds. Le but de l'algorithme IGNC est la construction de graphe.

Quelques notations:

- $n_i \in N$ un nœud,
- $ani -> nj \in A$ un arc entre les nœuds n_i et n_j,
- $P(ni)$ est la probabilité d'existence du nœud ni,
- $P(a n_i -> n_j)$ est la probabilité d'existence de l'arc $a n_i -> n_j$,
- wn_i est la position dans l'espace de représentation du nœud n_i,
- Mn_i est le modèle associé à n_i,
- $PMn_i(x)$ est la probabilité que la donnée représentée par x appartienne au modèle Mn_i.

L'algorithme IGNG fonctionne selon le principe le suivant:

Au départ le graphe est vide, lors de la présentation de première donnée, le premier nœud est créé et il est initialisé à sa position dans l'espace de représentation. A chaque fois qu'un batch de données est disponible, il est présenté plusieurs fois pour effectuer l'apprentissage afin d'affiner la position de chaque nœud du graphe, le cycle indique chaque présentation de tout le batch de données en cours d'apprentissage. A chaque étape, la sélection de donnée est effectuée aléatoirement parmi les données disponibles et le graphe est adapté, jusqu'à ce qu'il n'y ait plus de donnée à apprendre. Ceci est recommencé selon le nombre de cycles qui a été prédéfini.

Lors de l'arrivée d'une nouvelle donnée x au réseau, on cherche s'il existe des modèles Mni associés à des nœuds ni tel que la probabilité que la donnée présentée appartienne à ces modèles soit supérieure à un seuil fixé au préalable θ c.à.d. $PMni\, x > \theta$, on distingue trois cas :

- Si un seul nœud $s1$ vérifiant cette condition existe: on adapte sa position en le déplaçant au centre des données qui s'y projettent, et tous les arcs émanant de ce nœud voient leur probabilité d'exister diminuer.

- S'il existe plusieurs nœuds vérifiant cette condition: on crée un arc entre les deux nœuds les plus probables $s1$ et $s2$ en mettant sa probabilité d'existence au maximum et si cet arc existe déjà on remet sa probabilité d'existence au maximum. La probabilité d'existence associée à chaque arc est donnée par : $P\, ani \rightarrow nj = e - \gamma ni \rightarrow nj \beta$ avec $\gamma ni \rightarrow nj$ est l'âge de l'arc $ni \rightarrow nj$ et β est un paramètre à fixer a priori.

- Si aucun nœud ne vérifie cette condition: alors un nouveau nœud est créé et initialisé à la position de la donnée. La probabilité d'existence de ce nœud est initialisée au minimum pour permettre une résistance aux données bruitées et elle va augmenter au fur et à mesure que des données viennent se projeter dans ce nœud. La probabilité d'existence d'un nœud ni est donnée par $Pni = 1 - e - \tau ni \alpha$ avec τni le nombre de fois où ni a été considéré comme le nœud le plus probable et α est un paramètre à fixer a priori.

Enfin, tout arc dont la probabilité d'existence devient inférieure ou égale à un seuil $\emptyset$ donné sera supprimé. Les nœuds dont la probabilité d'existence devient inférieure ou égale à un seuil ψ ne seront pas supprimés mais ne seront considérés qu'en phase d'apprentissage et non en phase d'utilisation. Ces nœuds correspondent en réalité à des hypothèses de modèles de données en attente de confirmation.

2.9.13. AI2P: modèle d'Apprentissage Incrémental en 2 Phases

AI2P [ALM 08] est un modèle d'apprentissage incrémental à base de prototypes qui se déroule en deux phases pour apprendre progressivement un système de reconnaissance. La première phase consiste à créer un nouveau prototype pour chaque nouvel exemple. La seconde consiste à modifier les prototypes existants à chaque nouvel exemple. Deux techniques (méthodes) complémentaires ont été proposées dans le modèle AI2P, pour arriver à un apprentissage incrémental rapide et fiable.

Le taux de reconnaissance sera amélioré et la complexité (en termes de temps et mémoire) va diminuer lors de la création de prototypes. Pour cela, l'utilisation de l'adaptation devient indispensable afin d'avoir un système à la fois dynamique et léger. La création de nouveaux prototypes est « occasionnellement » par la méthode ADAPT.

Les processus d'apprentissage et d'adaptation sont supervisés: chaque exemple est étiqueté correctement. Cet étiquetage est possible en demandant au scripteur de vérifier le résultat de la reconnaissance ou à l'aide d'une technique auto-supervisée.

2.9.13.1. Phase1: Création de nouveaux prototypes

Dans cette phase, l'apprentissage est rapide et le processus d'apprentissage incrémental se compose de deux algorithmes principaux: un prototype hypersphérique est créé autour de chaque nouvel exemple et les conclusions sont calculées. Si la conclusion corresponde à la classe du caractère alors ses conclusions sont initialisées à 1, sinon 0. La méthode ADAPT est appliquée pour adapter tous les prototypes existants et leurs conclusions et pour ajuster globalement le système au nouvel exemple. Après avoir eu pour une classe N exemples (N prototypes) de cette classe, le système passe de la phase 1 à la phase 2.

2.9.13.2. Phase2: Modification de prototypes

La méthode ADAPT est utilisée dans cette phase pour modifier les prototypes en les déplaçant et déformant pour prendre en compte les connaissances acquises par les nouveaux exemples. Si le nombre d'erreurs de reconnaissance $(nbErr)$ pour une classe donnée dépasse un seuil défini (S) alors un nouveau prototype est crée pour cette classe.

Algorithme 2.11 (AI2P)

Pour chaque nouvel exemple e de la classe C faire
  Si classe C est dans phase 1 alors
    Appeler l'algorithme de création de prototypes avec e;
    Appeler l'algorithme d'adaptation;
      Si nombre d'exemples de classe $C \geq N$ alors
      Basculer vers phase 2
      Fin
  Fin
  Si classe C est dans phase 2 alors
   Appeler l'algorithme d'adaptation;

```
        Si e est mal-classé alors
          nbErr[C] + +;
            Si nbErr[C] ≥ S alors
            Appeler l'algorithme de création de prototypes;
             nbErr[C] = 0;
           Fin
       Fin
  Fin
Fin
```

2.9.14. K-moyennes incrémental

L'algorithme K-means est très coûteux en calcul, car dans chaque itération on calcule les distances entre les points de données et toutes les centroïdes. C'est pourquoi différentes approches de clustering de K-means incrémental sont proposées.

Gupta et Ujjwal [GUP 13] ont proposé un algorithme utilisant le seuil T qui dénote la dissimilarité entre les données et la valeur donnée de T^{th} ensuite un objet est choisi de manière aléatoire dans les jeux de données et devient le centre d'un cluster, puis un autre nouvel objet dans les jeux de données est sélectionné et la distance entre cet objet et le centre de cluster existant est calculée. Si cette distance est supérieure à T^{th}, créer un nouveau cluster et l'objet sélectionné sera le centre du cluster. Dans le cas contraire, grouper l'objet dans le cluster existant et mettre à jour son centroïde. Alors, il n'est pas nécessaire de spécifier la valeur de K et produit des clusters en un temps de calcul minimal.

Sowjanya et Shashi [SOW 10] ont proposé une approche CFIFA (Cluster Feature-based Incremental Clustering Approach) de clustering incrémental basée sur les fonctionnalités de cluster pour traiter efficacement les données incrémentales. La mise en clusters initiale et la gestion des données incrémentales sont deux étapes importantes des approches de mise en clusters incrémentale. CFICA a été appliquée pour la mise en clusters initiale, à l'aide de l'algorithme K-means. En utilisant la valeur moyenne, les deux groupes de clusters les plus proches ont été fusionnés après le traitement de l'ensemble de points de données. Une fois la base de données mise à jour, trois possibilités s'offrent pour regrouper les points mis à jour: (1). ajout avec le cluster existant; (2). apprentissage d'un nouveau cluster (3).possibilité de fusionner les clusters existants lorsque des points mis à jour se trouvent entre les deux clusters existants. Cette approche de regroupement incrémental est très efficace en termes de précision et de regroupement.

Chakraborty et al. [CHA 11] ont proposé un algorithme qui définit et évalue que le point de changement particulier dans la base de données ("seuil" ou "% delta") jusqu'à laquelle la mise en cluster incrémentale en K-means donne meilleurs résultats que la mise en cluster existante en K-means. De nos jours, les bases de données sont dynamiques, l'algorithme incrémental est utilisé pour gérer ces données. L'algorithme proposé identifie la valeur du pourcentage de la taille de la base de données (BDD) d'origine x, qui pouvant être ajoutée au BDD d'origine. Il pourrait y avoir deux cas: (1) si la base de données d'origine change jusqu'à $x\%$, alors utiliser le

résultat précédent; (2) si la base de données d'origine est modifiée jusqu'à plus de $x\%$, ré exécuter l'algorithme.

$Valeur\ seuil\ =\ (Nouvelles\ données\ -\ anciennes\ données)\ /\ anciennes\ données\ *\ 100$....(2.9)

En conséquence, l'algorithme de regroupement de K-means incrémental est appliqué aux données incrémentales après la collecte des informations nécessaires à partir de la base de données résultante, de sorte que les données à venir soient directement insérées dans la base de données existante sans exécuter à nouveau l'algorithme K-means. Cet algorithme permet une exécution plus rapide car le nombre d'analyses de la base de données sera diminué. Donc, fondamentalement, il offre de meilleures performances que l'algorithme de clustering K-means.

L'algorithme de [CHA 11] se résume comme suit:

Algorithme 2.12 (K-means incrémental)

Entrée: X: la base d'apprentissage contient n objets $\{X_1, X_2,....X_n\}$; et n: Nombre de données.

Sortie: K: ensemble de clusters.

Algorithme:

C_i $(i=1,2,3,....)$ est la nouvelle donnée.

1. Utiliser l'algorithme classique K- Means pour classifier le nouvel exemple Ci. Répéter jusqu'à ce que tous les exemples soient regroupés.

k- Means actuel----> $T1$ (temps d'exécution).

Pseudo-code de K- Means incrémental:

Début:

2. a) k représente les clusters existants.

b) Calculer la $moyenne\ (M)$ des clusters existants. et classifier directement la nouvelle donnée C_i.

Pour $i=1$ à n faire

Trouver la $moyenne\ (M)$ d'un cluster K_p parmi les moyennes calculées des clusters K, sachant que la distance $dis(C_i,M)$est minimale;

Si $dis(C_i,M)=min$ alors $K_p=K_p \cup C_i$

Recalculer la $moyenne(M)$ et le compare

Sinon si $dis(C_i!=min)--->min$ alors C_i va considérer comme un aberrant.

Mettre à jour les clusters existants.

c) Répéter l'étape b jusqu'à ce que tous les exemples sont classifiés.

k-Means incrémental---> $T2$ (Temps d'exécution).

Fin;

3. *Comparer (T1, T2), résultat<---(T2>T1) supérieur d'un certain seuil .*

2.9.15. DBSCAN incrémental

Plusieurs versions de DBSCAN incrémental ont été proposées, nous avons choisi de présenter l'algorithme le plus ancien d'Ester et al. [EST 96]. Cet algorithme est une extension de l'algorithme DBSCAN, il est capable d'ajouter des individus à un ensemble existant de clusters. Les nouveaux exemples sont classifiés un par un, en utilisant l'algorithme DBSCAN statique et par la suite les nouveaux clusters seront fusionnés avec les clusters existants. Comme les exemples sont insérés de manière incrémentale, les clusters sont découverts aussi de manière incrémentale.

Afin de comprendre la fonctionnalité de cet algorithme, quelques définitions ont été proposées :

Définition1 **(exemples affectés):** Soit X un ensemble de données et p le nouvel exemple ($p \in X$ ou $p \notin X$). Les exemples affectés sont potentiellement ceux qui peuvent changer leurs appartenance à un cluster après l'insertion du nouvel exemple p et ils sont définis par les exemples voisins de p $(N_\epsilon(p))$ ainsi que tous les autres exemples qui sont accessibles par densité depuis les exemples voisins**.** On définit l'ensemble des exemples dans X affectées par l'insertion de nouvel exemple p comme suit:

$$Affect_X(p) = N_\epsilon(p) \cup \{q | \exists o \in N_\epsilon(p) \wedge q >_{X \cup \{p\}} o\} \text{.....(2.10)}$$

L'exemple p influe sur l'ensemble de ses voisines. Après l'affectation, certains exemples peuvent changer leurs états par exemple: un objet bordure devient noyau.

Définition 2 (les exemples noyaux après la mise à jour / seed objects for the update) :

On prend en compte uniquement les points qui ont changé leurs états et sont devenus des noyaux. Soit X un ensemble de données et un exemple $p \notin X$:

$$UpSeed = \{ q | q \text{ est un noyau dans } X \cup \{p\},$$

$$\exists d : d \text{ est un noyau dans } X \cup \{p\} \text{ Mais pas dans } X \; et \; q \in N_\epsilon(d) \} \text{.....(2.11)}$$

La fonction d'insertion: lors de l'insertion d'un nouvel individu $\boldsymbol{p}$, des nouvelles connexions par densité peuvent être établies dans ce cas, il suffit de limiter l'application de la procédure de clustering à l'ensemble $\boldsymbol{Seed}$ **.**

Pour insérer un exemple p à l'ensemble de données X, quatre cas possibles:

1. **L'exemple $\boldsymbol{p}$ est un aberrant**: si $UpSeed$ est vide, l'exemple p sera classifié comme un élément aberrant et l'ensemble de clusters reste inchangé.
2. **Création du nouveau cluster** : si $UpSeed$ contient des exemples noyaux qui ne font pas partie d'un cluster, l'exemple p sera formé un nouveau cluster avec ces derniers.
3. **Absorption de p**: si $UpSeed$ contient des exemples déjà formés un cluster, l'exemple p sera absorbé dans ce cluster.
4. **Fusion des clusters**: si $UpSeed$ contient des exemples qui appartiennent aux clusters différents, l'exemple p sera formé un seul cluster avec ces clusters.

La figure 2.13 illustre les quatre cas de l'insertion d'un exemple p dans un ensemble des exemples X, en utilisant ces paramètres: le rayon $\varepsilon = 3$ et le nombre de voisinage $MinPts = 3$.

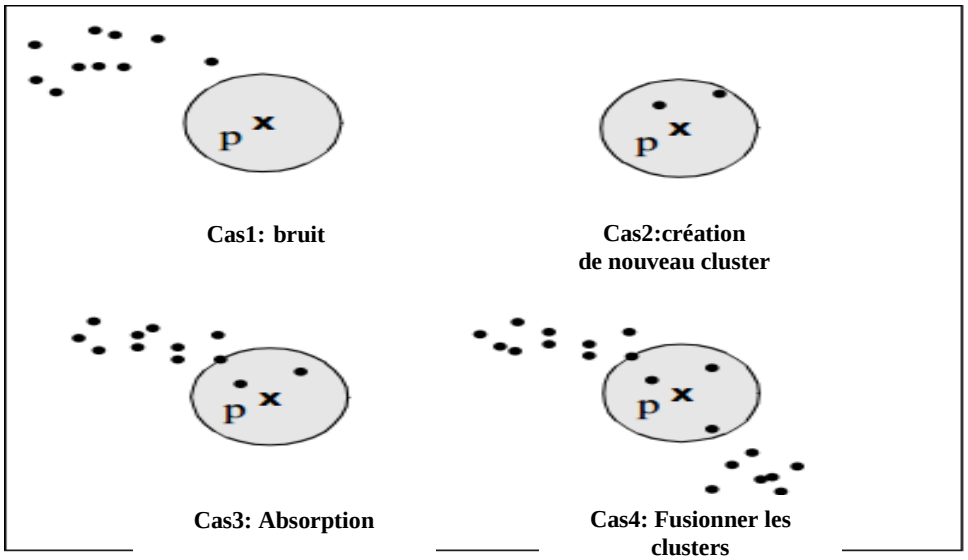

Figure 2.13. Les différents cas d'insertion d'un exemple p

L'algorithme DBSCAN incrémental proposé dans [EST 96] est résumé comme suit :

Algorithme 2.13 (DBSCAN Incrémental)

Entrées : *X:* ensemble des exemples, ε: le rayon, *MinPts:* le nombre minimum de voisins.
Sortie : C ensemble de clusters.

Début

1. Lorsqu'un nouvel exemple x_i arrive:
   A - Calculer l'ensemble des exemples noyaux "UpSeed " par la formule (2.11);
   B- Appliquer la fonction de l'insertion (quatre cas cités précédemment) sur l'ensemble des exemples noyaux "UpSeed " de l'étape précédente A;
2. Mettre à jour les clusters obtenus par l'étape 1.

Fin.

2.10. Conclusion

Nous avons abordé, au début de ce chapitre, les concepts de base de l'apprentissage incrémental tels que: les définitions, la relation dilemmatique entre la stabilité et la plasticité qui nécessite d'apprendre des nouvelles informations en retenant autant que possible les connaissances précédentes. Nous avons présenté ensuite le principe, les caractéristiques ainsi que les catégories de l'apprentissage incrémental. Nous avons consacré la suite du chapitre à présenter les principales méthodes et algorithmes d'apprentissage incrémental dont l'utilisation a une grande influence sur la performance des systèmes lorsque les ensembles de données d'apprentissage deviennent disponibles progressivement dans le temps.

CHAPITRE 3

CONTRIBUTIONS DANS L'APPRENTISSAGE SUPERVISE INCREMENTAL EN RECONNAISSANCE DE FORMES

Ce chapitre présente une partie de nos contributions dans le cadre de l'apprentissage automatique incrémental. L'objectif final de cette partie de notre travail, est de proposer un algorithme incrémental supervisé basé sur une hybridation de deux algorithmes incrémentaux.

Ce chapitre est scindé en deux parties: la première contient une synthèse d'un ensemble de travaux de l'apprentissage supervisé incrémental dans le domaine de la reconnaissance des formes [CHE 19a]. Dans la deuxième partie, nous proposons un système basé sur deux algorithmes incrémentaux ISVM-LEARN++ [CHE 19b] qui combine (fusionne) le SVM incrémental et le RN incrémental Learn++ pour la classification d'un grand nombre d'exemples d'apprentissage. Notre choix est motivé par notre intérêt pour les deux notions de combinaison et d'incrémental dans la classification. Nous avons choisi d'utiliser le type de combinaison parallèle et la méthode de combinaison de la somme pondérée afin de fusionner les deux classifieurs incrémentaux. La section 3.1 de ce chapitre est dédiée à la synthèse et la discussion d'un ensemble de travaux entrepris dans le domaine de l'apprentissage incrémental en reconnaissance de formes.

Les sections suivantes de ce chapitre sont consacrées à la présentation, la description et l'évaluation expérimentale de notre proposition de combinaison ISVM-Learn++ de classifieurs supervisés incrémentaux.

3.1. Synthèse de travaux relatifs à l'apprentissage supervisé incrémental dans la reconnaissance de formes

Au cours des dernières années, nous avons remarqué qu'il y a de nombreuses applications dans le domaine de la reconnaissance de formes utilisant l'apprentissage incrémental.

3.1.1. Description résumée des travaux

Nous résumons dans cette section certaines des approches de reconnaissance incrémentale supervisée, que nous avons choisi de présenter de manière chronologique:

- Déniz et al. [DÉN 02] proposent un algorithme appelé: IRDB (Incremental Refinement of Decision Boundaries). La principale caractéristique de cet algorithme est l'utilisation d'une combinaison de résultats de classification. IRDB utilise deux classifieurs: le classifieur NN (Nearest Neighbor) et le classifieur SVM (Support Vector Machines), avec une fonction noyau à base radiale (RBF). IRDB proposé a été testé sur un système de détection et de suivi des personnes.
- Mandziuk et al. [MAN 02] proposent une nouvelle approche ICL (Incremental Class Learning) qui est une procédure d'apprentissage supervisé pour les réseaux de

neurones. Au lieu d'apprendre un problème complexe à la fois, ICL se concentre sur l'apprentissage des sous-problèmes de manière incrémentale (les sous problèmes s'apprennent progressivement), en utilisant les résultats des apprentissages antérieurs pour un apprentissage ultérieur. Les solutions des sous problèmes sont combinées de manière appropriée pour résoudre le problème complet. ICL a été appliqué sur le problème de reconnaissance manuscrite de chiffres sans contrainte, reposant sur une représentation spatio-temporelle de modèles.

- Toh et Ozawa [TOH 03] proposent un système de reconnaissance de visage intelligent qui comporte deux caractéristiques: l'apprentissage incrémental en une passe et la génération automatique de données d'apprentissage. Ils utilisent un réseau de neurones appelé réseau d'allocation de ressources avec la mémoire à long terme (RANLTM), pour réaliser un apprentissage incrémental efficace sans souffrir d'un oubli important. Dans le système de détection de visage, la localisation du visage est effectuée sur la base des informations relatives à la couleur de la peau et aux bords dans la première étape. Ensuite, les traits du visage sont recherchés au sein de régions localisées utilisant un RAN et les caractéristiques sélectionnées sont utilisées pour la construction de visages candidats. Après la détection du visage, les visages candidats sont classés à l'aide de RANLTM. L'évaluation des performances de reconnaissance montre que la précision s'améliore sans augmenter le taux de faux positifs, même si l'apprentissage incrémental se poursuit. Ce fait suggère que l'apprentissage incrémental est une approche utile et efficace pour faire face aux tâches de reconnaissance.
- Ozawa et al. [OZA 05] proposent une nouvelle approche pour la construction de systèmes de reconnaissance de visage adaptatifs dans lesquels un espace de fonction des variables d'entrée et un classifieur sont appris de manière incrémentale. Une version étendue de l'analyse IPCA (Incremental Principal Component Analysis) et du réseau d'allocation de ressources avec mémoire à long terme (RAN-LTM) est efficacement combinée pour mettre en œuvre cette idée. Dans la mesure où IPCA met à jour de manière incrémentale un espace de fonctions en faisant pivoter les axes propres et en augmentant les dimensions, les entrées d'un classifieur neuronal doivent également changer dans leurs valeurs et le nombre de variables d'entrée.
- Erdem et al. [ERD 05a] proposent une nouvelle méthode incrémentale (SVMLearn ++) à l'aide d'un ensemble de SVM's générés avec la règle d'apprentissage Learn ++. SVMLearn ++ fonctionne avec deux fonctions différentes du noyau (noyau polynomial et noyau RBF) et il a été testé sur un jeu de données du monde réel et un jeu de données de référence.
- Zhao et al. [ZHA 06] proposent l'algorithme SVDU-IPCA, une nouvelle méthode IPCA (Incremental PCA) basée sur l'idée d'un algorithme de mise à jour de décomposition en valeurs uniques (SVD: Singular Value Decomposition), à savoir un algorithme IPCA basé sur la mise à jour de SVD. Le but de ce travail est de réaliser une analyse d'erreur de l'algorithme IPCA proposé, en utilisant une dérivation mathématique. L'algorithme SVDU-IPCA proposé fournit l'erreur maximale lors de l'évaluation de PCA en mode batch. Il contient un espace facilement étendu à un noyau et un algorithme IKPCA est également présenté.
- Prudent [PRU 06] propose deux contributions: la première est PAO (Perceptron Auto-Organisé), un réseau de neurones non supervisé utilisant une approche hybride. Ce système a été appliqué à la reconnaissance des lettres manuscrites, il donne de bonnes performances mais les capacités d'apprentissage de la PAO n'ont pas été réellement exploitées. La deuxième contribution concerne l'algorithme Learn ++. La contribution

réelle de Prudent était considérée comme intermédiaire entre les deux: il a proposé un nouvel algorithme d'apprentissage pour la construction d'une carte topologique permettant d'extraire progressivement la topologie des données. Cet algorithme, appelé Incremental Growing Neural Gas (IGNG), introduit la notion de probabilité d'existence de neurones et de connexions. Ce concept permet de gérer la pertinence d'un neurone dans l'arrivée d'un nouvel exemple. Il a proposé un autre algorithme appelé: SApIn utilisant la topologie de données fournie par l'IGNG pour échantillonner un jeu de données d'apprentissage et créer plusieurs sous-ensembles d'apprentissage utilisés pour former plusieurs classifieurs. SApIn est plus efficace et gère mieux l'apprentissage incrémental que Learn ++.

- Ghassabeh et Moghaddam [GHA 07] propose un nouveau système de reconnaissance de visage incrémental (IFR). L'IFR utilise une séquence de données pour la phase d'apprentissage. Par conséquent, la nécessité de conserver une grande quantité d'échantillons de données pour la phase d'apprentissage est réduite. La réduction de la taille de la mémoire et de la complexité fournie par les nouveaux réseaux d'extraction de caractéristiques adaptatifs (LDA) les rend appropriés pour différentes applications de reconnaissance de formes en temps réel. Ce système peut être résumé en deux étapes: (1) le prétraitement d'image, qui comprend la normalisation, l'égalisation de l'histogramme, le centrage moyen et l'omission de l'arrière-plan et (2) l'évaluation avec la fonction adaptative LDA. Le nouveau système IFR a été considéré comme le résultat d'une combinaison cascade d'un nouveau réseau adaptatif et le réseau adaptatif PCA (APCA). Toutes les images d'entrée sont recadrées et préparées pour le système IFR. Les résultats de la simulation montrent l'efficacité du système proposé pour l'estimation adaptative de l'espace des fonctions pour la reconnaissance faciale en ligne.
- Huang et al. [HUA 07] utilisent un algorithme d'apprentissage incrémental pour ajuster un classifieur puissant renforcé aux échantillons en ligne. L'entrée de cet algorithme est l'espace d'observation (instance) et le résultat est la prédiction d'espace. L'algorithme commence par la modification de la probabilité de chaque catégorie. L'algorithme d'apprentissage incrémental proposé prend Real AdaBoost avec des hypothèses faibles de partitionnement de domaines. Il atteint le même niveau de précision de classification dans les ensembles de test hors ligne et en ligne sans un échantillon hors ligne et est plus robuste contre les mauvaises conditions causées par le choix incorrect du rapport de renforcement en ligne. Il possède deux propriétés: la méthode d'estimation hors ligne basée sur la compétence et la méthode de calcul stable basée sur une factorisation de type Naïve-Bayes et l'optimisation globale de la fonction de perte objective hybride centrale par la mise à jour parallèle de toutes les hypothèses faibles. Cette approche incrémentale fonctionne tout en maintenant une bonne capacité de généralisation pour les environnements courants (tels que le jeu de tests de face). Il est non seulement extrêmement intéressant pour les applications pratiques dans divers environnements de la vie réelle mais il est aussi théoriquement significatif dans le domaine de l'apprentissage adaptatif.
- Hulley et Marwala [HUL 07] proposent une méthode d'apprentissage incrémental à l'aide d'un algorithme génétique (ILUGA) utilisant des classifieurs binaires SVM formés pour être des classifieurs puissants utilisant un algorithme génétique. Chacun des classifieurs est optimisé en utilisant GA pour trouver l'hyperplan de séparation optimal. Cela se fait en recherchant le meilleur noyau et la meilleure marge. Les poids de vote sont ensuite générés par GA à l'aide des classifieurs puissants.

- Luo et al. [LUO 07] proposent une version de SVM incrémentale, qui permet à la mémoire d'accroître le contrôle alors que le système continue à obtenir de nouvelles données. Cette méthode consiste à (a) oublier les vecteurs de support les plus anciens pour prendre en compte les données les plus récentes lors de la mise à jour de la fonction de décision; (b) reconnaître si une nouvelle base de données contient ou non de nouvelles informations. Cette méthode fournit des performances de reconnaissance statistiquement équivalentes à celles de l'algorithme batch tout en obtenant une réduction de la mémoire. Quelle que soit la complexité de l'algorithme, la mise à jour de la représentation interne à chaque étape incrémentale est coûteuse en temps de calcul.
- Ozawa et al. [OZA 08] présentent Chunk IPCA, un schéma d'apprentissage incrémental dans lequel l'algorithme IPCA étendu est utilisé pour l'apprentissage en espace propre et la combinaison de la méthode ECM (modèle évolutionniste connexionniste) et du voisin le plus proche (K-NN) est adoptée en tant qu'apprentissage par classifieur. Ce schéma d'apprentissage donne un nouveau concept aux systèmes de reconnaissance de formes: la sélection des caractéristiques et l'apprentissage du classifieur se font simultanément en ligne. Le Chunk IPCA apprend les vecteurs propres principaux sans erreur d'approximation grave et un rapport d'accumulation désigné est maintenu en augmentant automatiquement l'évolution des nouveaux axes propres.
- Reddy et al. [RED 09], proposent un système de reconnaissance d'action incrémental sur la base de l'arborescence des fonctionnalités, qui augmente lorsque des instances d'apprentissage supplémentaires deviennent accessibles. Cela nécessite de stocker toutes les instances d'apprentissage vues sous la forme d'un arbre de fonctionnalités. Les auteurs proposent d'utiliser dans un premier temps un SR (sphère / rectangle) pour générer l'arborescence en utilisant les caractéristiques (entités, spatiotemporelles) des exemples d'apprentissage étiquetés. Au lieu de diviser l'espace de caractéristiques de haute dimension par des hyperplans, comme dans les techniques d'arborescence précédentes, les points de caractéristiques sont organisés en régions dans l'espace de caractéristiques, qui sont spécifiées par l'intersection d'une sphère et d'un rectangle. L'arborescence SR-tree peut conserver un volume de région plus petit pour fournir des régions disjointes et à un plus grand diamètre afin de prendre en charge une recherche rapide et une recherche NN dans un espace de grande dimension. Lors de la phase de reconnaissance, ils ont extrait les fonctionnalités d'une vidéo d'action inconnue et ont classé chaque fonctionnalité dans une catégorie d'action à l'aide de SR-tree. La reconnaissance de l'action doit s'adapter par une méthode de vote simple en comptant les libellés des caractéristiques.
- Zou et al. [ZOU 09] proposent un algorithme d'apprentissage incrémental de SVM basé sur la partition fixe filtrante de l'ensemble de données. Les auteurs présentent l'algorithme «Two-class problems» et le généralisent à l'algorithme «Multiclass problems». Les résultats expérimentaux montrent que la technique d'apprentissage incrémental proposée peut considérablement améliorer l'efficacité de l'apprentissage SVM et que l'apprentissage incrémental SVM permet non seulement d'améliorer le taux d'identification correcte mais également d'accélérer le processus d'apprentissage.
- Almaksour [ALM 11] propose Evolve ++, une solution incrémentale pour l'apprentissage des classifieurs basée sur les systèmes d'inférence floue de premier ordre Takagi – Sugeno (TS). Cette approche comprend, d'une part, l'adaptation des conséquences linéaires des règles floues à l'aide de la méthode des moindres carrés récursifs et, d'autre part, l'apprentissage incrémental de l'antécédent de ces règles afin de modifier la composition en fonction de l'évolution de la densité de données dans l'espace

d'entrée. Avec cette approche, le système de reconnaissance est capable d'apprendre de nouvelles formes à partir de données très limitées, il peut encore s'adapter et s'améliorer pour chaque nouvel exemple disponible. La méthode proposée, Evolve ++, résout les problèmes d'apprentissage incrémental grâce à un paradigme d'apprentissage global dans lequel les prémisses et les conclusions des règles sont apprises dans une synergie de système SIF et non de manière indépendante. Dans leur deuxième contribution, les auteurs proposent la génération automatique de données artificielles pour accélérer l'apprentissage de nouveaux symboles. Cette technique de génération est basée sur la théorie sigma-lognormale, qui propose un nouvel espace de représentation des formes manuscrites basé sur la modélisation neuromusculaire du mécanisme d'écriture.

- Mohemmed et al. [MOH 12] évaluent la faisabilité de l'apprentissage SPAN, un algorithme d'apprentissage incrémental qui est basé sur l'encodage des informations d'entrée en tant que temps précis de pics, et le codage temporel pour Spiking Neural Network (SNN) sur la base de données MNIST (classification d'images de l'ensemble de données numériques manuscrites). SNN a été choisi car des pics sont utilisés pour communiquer. Les données sont codées dans le temps des pics, ce qui les rend appropriées pour le traitement de données spatio-temporelles et les données de la méthode SPAN sont codées dans le temps précis des pics. La première étape du processus d'apprentissage consiste à convertir les images en formes de pics à l'aide du simulateur logiciel Virtual Retina (VR) qui transforme l'entrée image / vidéo en formes de pics. Les modèles générés sont utilisés pour former une seule couche de SPAN pour la classification. Le MNIST est un ensemble de données non linéaire qui garantit que les différentes classes de formes de pointes ont des caractéristiques inter/ intra-classes plus complexes, c'est-à-dire que les formes appartenant à la même classe sont variées et qu'il existe un chevauchement entre les différentes classes, ce qui donne une indication claire sur la faisabilité d'apprentissage SPAN pour des applications pratiques.
- Kawewong et al. [KAW 13] proposent un cadre incrémental d'apprentissage sur les catégories de scènes d'intérieur. Ce cadre prend en charge les interactions incrémentales avec les humains, qui fournissent une rétroaction au système pour un apprentissage étendu à tout moment. Il est basé sur le réseau de neurones auto-organisant et incrémental à n valeurs proposé (n-SOINN), qui a été dérivé en modifiant le SOINN d'origine pour qu'il puisse être utilisé dans la reconnaissance de scènes. Le n-SOINN-SVM proposé effectue un apprentissage incrémental rapide, ce qui le rend compatible avec le cadre. Il offre une grande précision, comparable à celle de la méthode SOA (Self Organizing Actuators), tout en étant capable d'apprendre des retours supplémentaires d'experts humains.
- Molina et al. [MOL 14] proposent un algorithme d'apprentissage incrémental utilisant des machines à vecteurs support (SVM) et des images multimodales et une stratégie d'information à base de texture. Le système proposé intègre des caractéristiques anatomiques, de texture et fonctionnelles. L'ensemble de données a été prétraité par interpolation B-Spline, correction de champ de polarisation et standardisation d'intensité. Des approches statistiques indépendantes angulaires du premier et du second ordre ainsi qu'une quantification de phase locale invariante par rotation (RI-LPQ) ont été utilisées pour quantifier les informations de texture. Un ensemble d'apprentissage incrémental SVM a été mis en place pour répondre aux conditions de travail des applications médicales et pour améliorer l'efficacité et la robustesse du système.

- Lu et al [LU 14] présentent un système de reconnaissance permettant de détecter des actions humaines anormales à partir d'une surveillance de scène en temps réel basée sur une caméra, en utilisant un classifieur SVM incrémental. Dans ce système, l'extraction et la sélection des caractéristiques sont mises en œuvre en fonction des caractéristiques de couleur et de texture (apparence de la personne). Le corps de chaque personne est segmenté en trois parties (tête, haut et bas), et les caractéristiques de chaque silhouette, couleur et texture sont extraites dans les séquences vidéo, puis un ensemble de 93 caractéristiques est formé. Une méthode pour réduire l'espace des fonctionnalités est ensuite appliquée afin d'obtenir le meilleur ensemble. Les auteurs introduisent la technique de SVM incrémental pour reconnaître une personne car les classes (personnes) ne sont pas complètement connues au début et le SVM classique ne peut pas traiter ce cas.
- Bai et al. [BAI 15] décrivent comment construire un modèle de segment structuré incrémental pour la reconnaissance d'objet. La méthode proposée analyse les informations structurelles globales et de multiples attributs locaux d'objets pour la caractérisation d'un modèle d'objet. Ils utilisent des modèles de segment pour représenter les nœuds de structure, qui couvrent les informations locales d'un objet. Les segments sont obtenus par la méthode de segmentation et de classification et sont utilisés pour construire les modèles de segment en termes de fusion multi-facteurs et de SVM multi-classes. Le modèle de segment structuré est ensuite assemblé par interaction de différents segments via une configuration déformable. De plus, ils introduisent une stratégie d'apprentissage incrémental, qui apprend un modèle de segment en utilisant seulement un petit nombre d'échantillons d'apprentissage. Les images expliquées avec des entropies élevées sont utilisées pour mettre à jour le modèle formé. L'avantage de cette méthode est qu'elle permet d'appréhender les connexions inhérentes aux parties sémantiques des objets et caractérise les relations structurelles entre elles.
- Liu et al. [LIU 15] proposent un nouveau cadre d'apprentissage incrémental basé sur des réseaux de neurones profonds (DNN) afin de résoudre le problème de la classification des images. L'idée de base est d'utiliser les paramètres du réseau déterminés avec des images à basse résolution pour améliorer les valeurs initiales des paramètres du réseau pour les images de haute résolution. Il y a deux solutions pour mettre en place cette idée: l'une consiste à utiliser les réseaux avec des filtres mis à l'échelle lorsque la taille des filtres dans les réseaux profonds est étendue en augmentant les paramètres du réseau précédemment formé avec des images de résolution inférieure. L'autre consiste à ajouter des filtres de convolution au réseau. Les avantages de cette méthode sont les suivants: un modèle préformé peut être utilisé comme initialisation d'autres réseaux pour optimiser le temps d'apprentissage fastidieux de jeux de données à grande échelle, par exemple. Le jeu de données ImageNet est un problème inquiétant. La méthode d'apprentissage incrémental peut atteindre une efficacité supérieure pour l'apprentissage sans sacrifier la précision et la performance, mais peut légèrement améliorer l'efficacité et la performance grâce aux méthodes d'apprentissage incrémentales.
- Zribi et Boujelbene [ZRI 16] utilisent un système neuronal incrémental: les réseaux comme outil de détection des erreurs humaines et techniques de diagnostic du cancer du sein. Cette méthode incrémentale aide à améliorer le diagnostic correct dans ce domaine, elle permet de classer une masse dans le sein (bénigne et maligne) en

utilisant une sélection des facteurs de risque les plus pertinents et une prise de décision pour le diagnostic de cancer du sein.

- Han et al. [HAN 16] proposent un nouveau classifieur boosté incrémental CNN (IB-CNN) qui intègre le renforcement dans le réseau de neurones convolutif CNN (Convolutional Neural Networks) via une couche de renforcement incrémental. Les neurones discriminants de la couche inférieure sont sélectionnés et la mise à jour est effectuée de manière incrémentale sur des mini-lots successifs. En outre, une nouvelle fonction de perte, tenant compte des erreurs du classifieur incrémental renforcé, a été proposée pour affiner le réseau IB-CNN.
- Hacene et al. [HAC 17] introduisent un nouvel algorithme incrémental basé sur des CNN pré-formés (réseaux de neurones convolutifs) et des mémoires associatives pour classifier les images. Les premiers utilisent les poids de connexion pour traiter les images et les secondes utilisent l'existence de connexions pour les stocker efficacement. Cette combinaison de méthodes permet d'apprendre et de traiter des données en diminuant le nombre d'exemples, la taille de la mémoire et la puissance de calcul.
- Lawal et Abdulkarim [LAW 17] présentent un SVM adaptatif pour la classification des flux de données. Ils proposent une méthode de sélection de modèle d'apprentissage incrémental (MS-IL) pour le SVM, dans laquelle ils ont introduit l'idée de la validation croisée de k-échantillons afin de permettre le réglage incrémental des hyper-paramètres de SVM pour garantir son efficacité et exploiter les flux de données d'apprentissage nouvellement acquis. L'IL-MS a été testé sur le problème du filtrage des spams en ligne et de la classification humaine dans les flux vidéo.
- Sarwar et al. [SAR 17] développent un réseau de neurones à convolution profonde (DCNN). Pour apprendre un nouvel ensemble de classes, ils forment un nouveau réseau en réutilisant les couches convolutionnelles précédemment apprises (partagées à partir de la partie initiale du réseau de base), suivies de nouvelles couches convolutives qui peuvent être entraînées (ajoutées) vers les couches ultérieures du réseau. Les couches de convolution partagées fonctionnent comme des extracteurs de fonctions fixes (les paramètres d'apprentissage sont gelés) et minimisent la surcharge d'apprentissage pour un nouvel ensemble de classes. La propriété de résilience aux erreurs du réseau neuronal garantit que même si certains paramètres d'apprentissage sont fixés, le réseau sera en mesure de s'y adapter et d'apprendre.
- Chen et al. [CHE 19d] proposent une nouvelle méthode d'apprentissage hybride HIL (Hybrid Incremental Learning) qui utilise à la fois l'apprentissage incrémental de données et l'apprentissage incrémental de classe pour la reconnaissance d'objets portables. Cette nouvelle méthode d'apprentissage incrémental apprend automatiquement de nouveaux concepts en ajoutant de nouveaux plans de classification au modèle et améliore la qualité de reconnaissance des concepts appris en ajustant les plans de classification existants du modèle dans le cadre de SVM.

3.1.2. Discussion et tableau de synthèse des travaux

D'après notre étude des techniques supervisées incrémentales dans le domaine de la reconnaissance des formes, nous pouvons formuler les remarques suivantes:

- La plupart des techniques supervisées incrémentales ont été testées dans le domaine de la reconnaissance de visages;

- les performances de ces techniques sont excellentes en termes de précision de reconnaissance, comme l'étude de Zribi et Boujelbene [ZRI 16], où la précision de reconnaissance obtenue atteint 99,95%;
- la plupart des études améliorent les versions de SVM et de réseau de neurones artificiels par rapport à tout autre algorithme d'apprentissage supervisé;
- nous remarquons que la tendance actuelle de la recherche dans ce domaine est la combinaison d'un apprentissage incrémental et d'un apprentissage en profondeur (Deep Learning), particulièrement pour la reconnaissance d'images.

Le tableau 3.1 synthétise les travaux étudiés et résumés dans la section 3.1.1. Les travaux sont triés par ordre chronologique, en mettant en évidence, pour chaque travail, la méthode utilisée, le domaine d'application, les bases de données et les résultats obtenus.

Auteurs et réference	Méthode proposée	Domaine d'application	Bases de données	Les résultats comparés avec	Résultats obtenus
(Déniz et al) [DÉN 02]	– IRDB (Incremental Refinement of Decision Boundaries) utilise un classifieur NN (Nearest Neighbor) et un classifieur SVM (Support Vector Machines), avec la fonction noyau à base radiale RBF.	Reconnaissance de visages	2 classes ont été étudiées: 10 séquences, une pour chaque individu, chacune avec 167 images. Les images sont fournies par le système DESEO. Toutes les images utilisées dans les expériences ont une taille de 39x43 pixels. Pour chaque séquence, 3 images ont été utilisées pour former le classifieur, 50 en tant qu'informations supervisées pour l'apprentissage et les autres pour le test.	/	--
(Mańdziuk and Shastri) [MAN 02]	– Approche ICL (Incremental Class Learning) pour les réseaux de neurones. – Apprentissage des sous-problèmes de manière incrémentale. – Combinaison des solutions des sous problèmes pour résoudre le problème complet.	Reconnaissance de chiffres manuscrits	La base de données USPS CEDAR est composée d'environ 2 400 codes postaux manuscrits recueillis à partir de pièces de courrier.	Le classifieur K-plus proche voisin (K-NN)	Précision=93.13%
(Toh et Ozawa) [TOH 02]	– Un algorithme d'apprentissage incrémental proposé fournit deux fonctionnalités: l'apprentissage incrémental en une passe et la génération automatique de données d'apprentissage. – Utilisation d'un réseau d'allocation de ressources avec mémoire à long terme (RAN-LTM) pour classifier les visages candidats.	Reconnaissance de visages	Des clips vidéo de sept personnes: quatre personnes (2 hommes et 2 femmes) sont choisies comme personnes enregistrées et les trois autres personnes sont des personnes non enregistrées. L'ensemble de données en ligne comprend 654 images détectées à partir des clips vidéo de la première semaine. Par contre, le jeu de données de test est constitué de 499 images détectées à partir des clips de la deuxième semaine.	Les résultats de trois réseaux de neurones	--
(Ozawa et al) [OZA 05]	– Combinaison d'une version étendue de l'analyse incrémentale en composantes principales (IPCA) et du réseau d'allocation de ressources avec mémoire à long terme (RAN-LTM).	Reconnaissance de visages	224 clips vidéo auto-compilés, provenant de 22 personnes (19 hommes et 3 femmes), sont transmis à la partie détection de visage, puis les 7582 images de visage détectées sont utilisées pour l'évaluation des performances.	/	Précision=97%
(Erdem et al) [ERD 05]	– Nouvelle méthode incrémentale utilisant un ensemble de SVM formés avec la règle d'apprentissage Learn ++ (SVMLearn ++) est utilisée. SVMLearn ++ avec deux fonctions différentes du noyau: polynôme et gaussien	Reconnaissance optique de caractères	Les données de reconnaissance optique de caractères (OCR) ont 10 classes avec des chiffres de 0 à 9 et 64 attributs. Le jeu de données composés organiques volatils (COV) se compose de 5 classes avec 6 attributs.	/	La performance de la généralisation sur la base de données OCR améliorée à partir de 78-80%; La performance de la généralisation sur la base de données VOC

					amélioré à partir de 83-85%
(Zhao et al) [ZHA 06]	– Algorithme SVDU-IPCA pour effectuer une analyse d'erreur de l'algorithme IPCA proposé, en utilisant une dérivation mathématique contient un algorithme IKPCA facilement étendu à l'espace noyau	Reconnaissance de visages	Base de données FERET, base de données ARface.	Reconnaissance des visages propres et visuels. Algorithme IPCA	Précision=93%
(Prudent) [PRU 06]	– AO, un réseau de neurones non supervisé utilisant une approche hybride. – Incremental Growing Neural Gas (IGNG) introduit la notion de probabilité de l'existence de neurones et de connexions. – SApIn utilisant la topologie de données fournie par l'IGNG.	Reconnaissance des lettres manuscrites	NIST et optdigits de l'UCI pour valider les performances de la base de données IGNC, OCR pour tester SApIn.	Learn++	
(Ghassabeh et Moghaddam) [GHA 07]	– Algorithme d'apprentissage adaptatif pour l'estimation de la caractéristique LDA pendant le processus d'apprentissage incrémental. – Combinaison d'un nouveau réseau $\Sigma - 1/2$ adaptatif en cascade avec le réseau APCA.	Reconnaissance de visages	Base de données de visages YALE: contient des images en niveaux de gris de 15 sujets au format GIF. Ils ont choisi 5 sujets individuels et pris en compte 64 images pour chaque sujet (320 images au total) contenant des éclairages différents et des poses différentes.	/	Les erreurs normalisées = 0.121, 0.232 and 0.305
(Huang et al) [HUA 07]	– Algorithme d'apprentissage incrémental prend Real AdaBoost avec des hypothèses de partitionnement de domaine faibles comme pour la base, basé sur Discrete AdaBoost.	Reconnaissance de visages	20000 pièces frontales de taille 24x24 sont collectées dans la base de données FERET. 1425 images de visage frontal de la base de données CMU PIE.	L'approche combinée d'apprentissage	Taux de détection=0.97
(Hulley et Marwala) [HUL 07]	– Approche d'apprentissage incrémental à l'aide d'un algorithme génétique (ILUGA). – Classifieurs binaires SVM optimisés avec un algorithme génétique.	Reconnaissance optique de caractères	Le jeu de données OCR comporte 10 classes d'attributs de chiffres de 0 à 9 et de 64 attributs. Base de données de reconnaissance de wine.	Learn++, Learn++.MT, SVMLearn++, SVMLearn++.MT	Une précision globale de 93% et 94%
(Luo et al) [LUO 07]	– Approche d'apprentissage incrémental discriminatoire pour la reconnaissance de position. – Version de SVM incrémental, qui permet de contrôler la croissance de la mémoire alors que le système continue d'acquérir de nouvelles données.	Reconnaissance des lieux dans des environnements dynamiques	La base de données IDOL2 (base de données d'images pour rObot Localisation 2) contient 24 séquences d'images acquises par une caméra en perspective, montées sur deux plates-formes de robots mobiles, PeopleBot Minnie et PowerBot Dumbo.	Techniques à partition fixe, pilotées par les erreurs et contrôlées par la mémoire	Taux de classification=97%
(Ozawa et al) [OZA 08]	– Chunk IPCA, l'algorithme étendu IPCA pour l'apprentissage en espace propre. – Combinaison de la méthode ECM (modèle évolutionniste connexionniste) et du voisin le plus proche (K-NN).	Reconnaissance de formes	Six jeux de données standard dans le référentiel UCI Machine Learning: Adult, Letter recognition, Spambase, MUSK, Isolet Spoken Letter recognition, Internet Advertisement, Microarray.	Eigenspace fixe, IPCA et PCA d'origine	Précision=88%

(Reddy et al) [RED 09]	– Reconnaissance d'action incrémentale à l'aide d'une arborescence de fonctionnalités. Il ne nécessite pas d'apprentissage intensif (construction de vocabulaire et modélisation de catégories). Il est implémenté à l'aide d'une structure de données d'arborescence SR basée sur disque.	Reconnaissance des actions	Le jeu de données KTH et la vue multiple IXMAS ensemble de données pour chaque vidéo, ils extraient 200 points d'intérêt et les cuboïdes correspondants. Descripteur d'entité basé sur le dégradé.	/	Précision=90.3%
(Zou et al) [ZOU 09]	– Algorithme d'apprentissage incrémental de SVM basé sur la partition fixe de filtrage de l'ensemble de données. – Algorithme "Two-class problems" est présenté, puis généralisé à l'algorithme "problèmes "multiclass problems"".	Reconnaissance d'images	Trois types de données: Images de télédétection à texture naturelle; bImages infrarouges de l'avion; les données de test couramment utilisées (données de test XOR bi-classe)	SVM statique et SVM incrémental	Taux correct de classification =0.92
(Almaksour) [ALM 11]	– Evolve ++, approche incrémental pour l'apprentissage des classifieurs basés sur des systèmes d'inférence floue SIF Takagi-Sugeno d'ordre 1. – La génération automatique de données artificielles pour accélérer l'apprentissage de nouveaux symboles, basée sur la théorie Sigma-lognormal.	Reconnaissance de gestes manuscrits	SIGN "composé de 25 gestes différents dessinés par 11 auteurs différents sur des Tablet PC. Chaque auteur a tiré 100 échantillons de chaque geste. Bases de données UCI: CoverType, PenDigits, Segment, Lettres.	Modèle AI2P; Méthodes de classification en deux lots (KNN et MLP)	
(Mohemmed et al) [MOH 12]	– Algorithme d'apprentissage incrémental SPAN pour les réseaux de neurones à pointes qui sont basés sur le codage d'informations d'entrée en tant que temps précis de pointes (codage temporel).	Reconnaissance de chiffres manuscrits	Les chiffres manuscrits MNIST où chaque image est de 28 × 28 pixels. Ils utilisent 200 échantillons d'images par chiffre (soit un total de 2 000 images) pour l'apprentissage; Le modèle de pointes produit est constitué de 784 trains de pointes.	/	La précision moyenne de 92% dans l'ensemble d'apprentissage et de 86,6% dans l'ensemble de test
(Kawewong et al) [KAW 13]	– Nouveau mode incrémental pour ajouter de nouvelles classes et de mettre à jour les classes existantes avec de nouvelles informations à tout moment. – Basée sur les réseaux neuronaux auto-organisés et incrémentaux à n valeurs proposés (n-SOINN) qui ont été modifiés par rapport au SOINN d'origine.	Reconnaissance de scènes d'intérieur	Les jeux de données de scènes d'intérieur du MIT 67 catégories	RBF-SVM-k-means RBF-SVM-Aléatoire Original-SOINN-SVM	--
(Molina et al) [MOL 14]	– SVM utilisant des images IRM multimodales et une stratégie d'information à base de texture intégrant des caractéristiques anatomiques, fonctionnelles, utilisant l'interpolation B-Spline, la correction de champ biais et la	Classification multimodale de l'adénocarcinome de la prostate	La base de données IRMmage comprenait des coupes transversales IRM avec trois modalités: flux plasmatique amélioré à contraste dynamique (DCE-PF) pondéré en T2 et temps de transit moyen du DCE (DCE-MTT).	/	La sensibilité moyenne (TPR) de 0,8446 ± 0,068; la spécificité moyenne (SPC) de 0,7806 ± 0,038

	normalisation d'intensité.				
(Lu et al) [LU 14]	– L'extraction et la sélection des caractéristiques sont mises en œuvre en fonction des caractéristiques de couleur et de texture (apparence de la personne). – Classifieur SVM incrémental pour reconnaître une personne.	Reconnaissance humaine en ligne de la vidéosurveillance	La base de données CASIA-Gait contient un ensemble de 93 fonctionnalités.	SVM classique	Taux correct de classification est de 98.46%
(Bai et al) [BAI 15]	– Un modèle générique basé sur une pièce structurée incrémentale générique, qui permet non seulement une caractérisation d'objet hiérarchique étendue, mais également la mise à jour du modèle par rapport à de nouveaux échantillons d'apprentissage, est développé.	Reconnaissance des objets	Le jeu de données Caltech-256 contient 30 607 images dans 256 catégories, chaque classe contenant au moins 80 images. Le jeu de données Pascal VOC 2007 comprend 9 963 images provenant de 20 classes.	PASCAL 07 Best, LLC et Su	Le taux de ratés = 0,1 faux positifs par image. Précision moyenne (PA) = 40,9% pour la première catégorie, 13,1% pour la seconde et 32,6% pour la troisième.
(Liu) [LIU 15]	– Nouvel algorithme d'apprentissage incrémental basé sur des réseaux de neurones profonds (DNN) pour résoudre le problème de la classification des images. Il utilise les paramètres réseau formés avec des images à basse résolution pour améliorer les valeurs initiales des paramètres réseau pour les images à haute résolution.	Classification d'images	Le jeu de données ImageNet et le jeu de données Places205	Méthode traditionnelle	Précision=88%
(Zribi et Boujelbene) [ZRI 16]	– Méthode incrémentale de réseau de neurones artificiel proposée pour classer une masse dans le sein (bénigne et maligne) en utilisant une sélection des facteurs de risque les plus pertinents et une prise de décision pour le diagnostic de cancer du sein.	Erreurs humaines et techniques de diagnostic du cancer du sein	Base de données du Wisconsin sur le cancer du sein (WBCD) contient un échantillon de 410 patients pour l'apprentissage et 273 pour l'échantillon test; Le réseau comprend un terminal d'entrée et 9 neurones de couche d'entrée, 4 neurones cachés associés à une fonction d'activation sigmoïdale dans la couche de sortie (malin et bénigne)	Algorithme de Perceptron Multilayer de Métaplasticité Artificielle (AMMLP)	Taux de classification = 99.95%
(Han et al) [HAN 16]	– Système IB-CNN pour intégrer la classification de renforcement dans un CNN. – Un algorithme de renforcement incrémental a été développé pour accumuler les informations de plusieurs lots au fil du temps. – Une nouvelle fonction de perte qui prend en compte les erreurs du classifieur fort	Reconnaissance de l'unité d'action faciale	Quatre bases de données de référence codées par l'UA: La base de données CK qui contient 486 séquences d'images de 97 sujets. La base de données FERA2015 SEMAINE qui contient 6 UA et 31 sujets avec 93 000 images. La base de données FERA2015 BP4D contient 11 AU et 41 sujets avec 146 847	CNN traditionnel	Taux de classification moyen: Pour la base de données CK = 0,951; pour la base de données SEMAINE = 0,416; pour la base de données BP4D = 0,578; pour la base de données

	incrémental et des classifieurs faibles individuels est proposée pour affiner le réseau IB-CNN.		images. La base de données DISFA contient 12 AU étiquetés et 27 sujets avec 130 814 images.		DISFA = 0,858
(Hacene et al) [HAC 17]	– Combinaison d'un CNN pré-entraîné (réseaux de neurones convolutionnels) et des mémoires associatives pour classifier les images.	Reconnaissance d'images	Base de données ImageNet (contenant 10 classes chacune distincte des 1000 classes utilisées pour former le CNN) et base de données Cifar10	Le 1-NN (voisin le plus proche) et 5-NN	Précision=86.07%
(Lawal et Abdulkarim) [LAW 17]	– Une méthode de sélection de modèle d'apprentissage incrémental (IL-MS) pour la SVM pour le réglage incrémental des hyper-paramètres de SVM afin de garantir son efficacité tout en exploitant les nouvelles fonctions acquises sur les flux de données d'apprentissage.	Filtrage en ligne des spams et classification des actions humaines à partir de vidéos	L'ensemble de données de courrier électronique de courrier indésirable (9324 courriers) comprend deux classes: les courriers électroniques légitimes et le courrier indésirable. Jeu de données vidéo d'action KTH: 600 clips vidéo pour toutes les combinaisons de 25 personnes, 6 actions et 4 scénarios.	Le NO-MS SVM et C-MS SVM	La précision de classification moyenne de 79,1% à 91,6%
(Sarwar et al) [SAR 17]	– Un réseau de neurones à convolution profonde (DCNN) pour accueillir un nouvel ensemble de classes par partage de réseau, sans oublier les anciennes classes.	Classification d'images	Les bases de données ImageNet, CIFAR100 et CIFAR10 pour la reconnaissance d'objets et TiCH pour la reconnaissance de caractères.	/	Taux de classification =84.13%
(Chen et al) [CHE 19d]	– Une nouvelle méthode d'apprentissage hybride (HIL) utilisant l'apprentissage incrémental de données et l'apprentissage incrémental de classe pour apprendre de nouveaux concepts – Utilisation des SVM dans le processus de l'apprentissage.	Reconnaissance des objets portables	HOD-20 contient 16,000 images et 20 catégories d'objets communs; ImageNet-30 contient 1 300 images par classe et 39 000 images au total.	MULTIpLE; SV-L-Inc; HIL-Offline;	Précision= 94.2%

Tableau 3.1. Synthèse des applications de l'apprentissage incrémental dans le domaine de la reconnaissance de formes.

Dans le tableau 3.1, nous avons constaté que les algorithmes incrémentaux combinés donnent des bons résultats par rapport à d'autres méthodes. Nous citons, à titre d'exemple:

- L'algorithme incrémental d'Ozawa et al. [OZA 05] qui combine l'analyse incrémentale en composantes principales (IPCA) et le réseau d'allocation de ressources avec mémoire à long terme (RAN-LTM) et donne une précision de reconnaissance de 97%;
- L'algorithme SVDU-IPCA de Zhao et al. [ZHA 06] utilise l'algorithme IPCA et un algorithme IKPCA avec une précision de reconnaissance de 93%;
- L'algorithme d'apprentissage incrémental ILUGA proposé par Hulley et Marwala [HUL 07] utilisant les algorithmes génétiques pour optimiser des classifieurs binaires SVM avec une précision globale aux alentours de 94%.
- Le tableau 3.1 nous permet d'observer que les réseaux de neurones incrémentaux donnent également de bons taux de reconnaissance par rapport à d'autres algorithmes d'apprentissage supervisé. Nous identifions:
- L'approche ICL (Incremental Class Learning) pour les réseaux de neurones proposée par Mandziuk et al. [MAN 02] avec une précision de 93, 13%;
- L'algorithme d'apprentissage incrémental SPAN proposé par Mohemmed et al. [MOH 12] et utilisé pour les réseaux de neurones avec une précision moyenne de 92%;
- Le réseau incrémental de neurones artificiels proposé de Zribi et Boujelbene [ZRI 16] avec une précision de 99,95%.

3.2. Proposition de combinaison de classifieurs incrémentaux: ISVM-Learn++

Un classifieur est défini comme étant tout système de traitement de données qui reçoit en entrée une forme x et donne en sortie des informations ou connaissances à propos de la classe correspondante à cette forme. Quel que soit le domaine d'application pour lequel il est utilisé, la mise en œuvre de tout classifieur nécessite de choisir d'abord une représentation pour décrire les données (caractéristiques), un algorithme de décision et une base d'apprentissage permettant de fixer les paramètres du classifieur.

3.2.1. Pourquoi combiner des classifieurs?

Plusieurs chercheurs ont fait les constats suivants [CHE 12] :

- Il n'existe pas de « meilleur » classifieur capable d'apprendre n'importe quelle base d'apprentissage;
- Aucun classifieur ne peut séparer suffisamment et efficacement n'importe quel ensemble de classes;
- Lorsqu'on utilise plusieurs classifieurs séparément pour le même ensemble de formes, les ensembles des formes mal classées ne sont pas forcément les mêmes.

De ces constats et de bien d'autres aussi, a émergé l'idée de faire coopérer les classifieurs.

L'idée principale derrière la combinaison de classifieurs est l'augmentation de performances [Hul 94]. Cette augmentation de performance peut avoir plus de fiabilité dans les réponses, ou moins de rejet, ou bien les deux à la fois. Donc, deux raisons majeures sont derrières la combinaison :

- **La Précision:** une décision plus fiable peut être obtenue en combinant l'avis (les sorties) de plusieurs experts (classifieurs);

- **L'efficacité:** un problème complexe peut être décomposé en plusieurs sous-problèmes qui sont plus faciles à comprendre et à résoudre (diviser pour mieux régner...).

Cette précision et cette efficacité proviennent essentiellement des faits suivants :

- Distribuer les caractéristiques sur des classifieurs différents;
- Exploiter la complémentarité entre les classifieurs;
- Compensation mutuelle des limites des méthodes utilisées;
- Plusieurs avis valent mieux qu'un;
- Prendre en compte les performances de chacun des classifieurs membres de la combinaison.

La combinaison des classifieurs a été utilisée avec succès dans plusieurs domaines tels que : la reconnaissance de formes (caractères, écriture,….). Elle se présente comme une tendance dans les recherches développées sur les systèmes intelligents, les systèmes experts.... et elle dépend de la fonction de fusion et de schéma de ces derniers.

De nombreux chercheurs ont appliqué les techniques de combinaison de classifieurs à l'apprentissage incrémental et de nombreux algorithmes basés sur la combinaison de classifieurs ont été proposés (voir chapitre 2 et section 3.1).

3.2.2. Combinaison proposée ISVM-Learn++

Dans le cadre de l'apprentissage supervisé incrémental, nous proposons la combinaison ISVM-Learn ++ qui intègre deux classifieurs incrémentaux, le SVM incrémental de Cauwenberghs et Poggio [CAU 01] (voir section 2.9.6) et le réseau neuronal incrémental Learn++ de Polikar et al [POL 01] pour la classification d'un grand nombre d'exemples d'apprentissage, afin de bénéficier des apports des deux concepts de combinaison et incrémental dans la classification. Nous avons choisi d'utiliser le type de combinaison parallèle et la méthode de combinaison de somme pondérée pour fusionner les deux classifieurs incrémentaux.

Nous nous basons sur les travaux de Polikar et al [POL 01] (comme mentionné dans la section 2.9.4) qui proposent un algorithme d'apprentissage incrémental Learn++ basé sur l'algorithme AdaBoost, qui utilise plusieurs classifieurs pour permettre au système d'apprendre de manière incrémentale. Cet algorithme fonctionne de nombreux classifieurs qui sont des apprenants faibles pour donner une bonne précision de classification. Un faible classifieur classe les données avec une précision de 50%. Les apprenants faibles sont formés sur un sous-ensemble distinct des données d'apprentissage, puis les classifieurs sont fusionnés en utilisant une technique de combinaison de vote majoritaire pondéré. Les pondérations pour le vote majoritaire pondéré sont choisies en fonction de la performance des classifieurs sur l'ensemble du jeu de données d'apprentissage.

Certaines versions modifiées, telles que: Learn ++. MT [MUH 04] utilisant le vote pondéré dynamique (DWV) et Learn ++. NC [MUH 09] utilisant la consultation et le vote pondérés dynamiquement (DW-CAV).

Dans notre proposition, nous nous sommes aussi inspirés des travaux de Wen et Lu [WEN 07] qui proposent un nouvel algorithme d'apprentissage incrémental, ILbyCC, qui utilise la règle de la moyenne de Bayes pour combiner les classifieurs. ILbyCC peut non seulement préserver les connaissances acquises précédemment, mais également acquérir de nouvelles connaissances à partir de données récemment ajoutées et de nouvelles informations à partir de classes

nouvellement introduites. L'algorithme proposé entraîne une machine à vecteurs de support capable d'émettre des informations de probabilité ultérieure une fois que des données d'apprentissage par lots incrémentales sont acquises. Les sorties de toutes les machines à vecteurs de support résultantes sont simplement combinées par la méthode de moyenne.

3.3. Description détaillée de ISVM-Learn++

Dans cette section, nous détaillons notre proposition, en commençant par une description détaillée de l'architecture proposée pour ISVM-Learn++, ainsi que l'apprentissage et le test des classifieurs classiques et les classifieurs incrémentaux individuels, enfin la combinaison des deux classifieurs incrémentaux.

3.3.1. Architecture de la combinaison proposée

Notre combinaison comprend comme la montre la figure 4.1:

- Le premier classifieur SVM incrémental proposé par Cauwenberghs et Poggio [CAU 01] ;
- Le deuxième classifieur Learn++ proposé par Polikar et al. [POL 01];
- Le module de combinaison ISVM-Learn++.

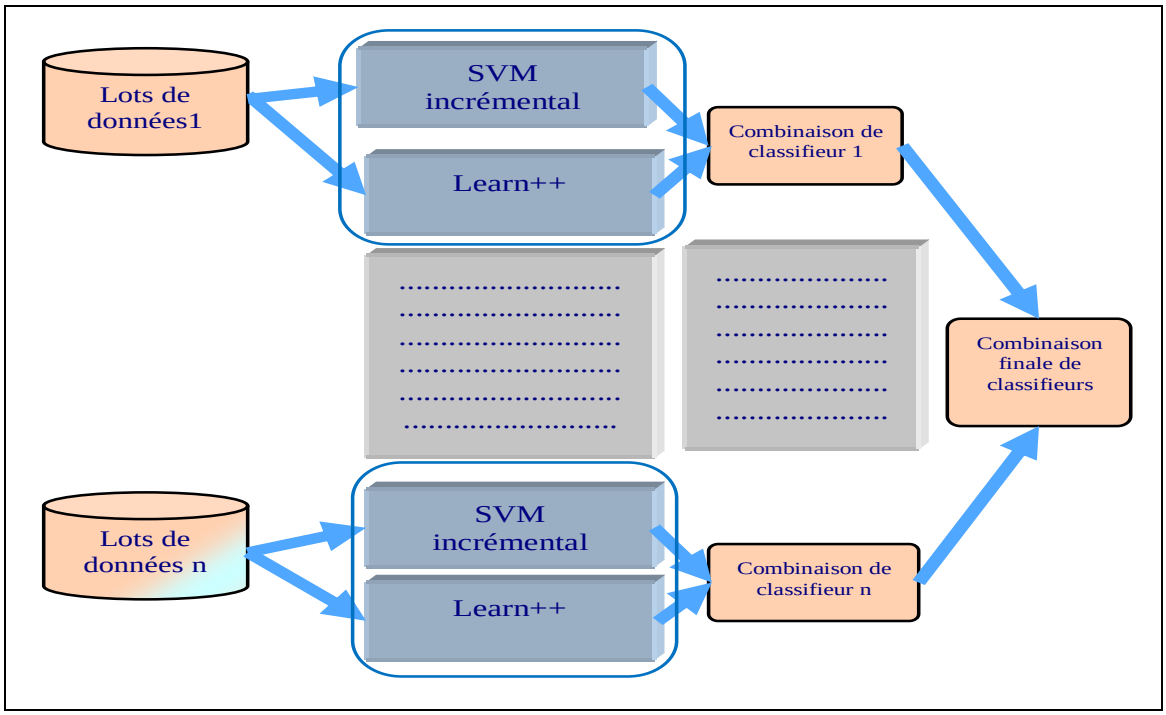

Figure 3.1. Architecture de la combinaison proposée ISVM-Learn++

L'approche proposée opère selon deux modes : le mode d'apprentissage et le mode de test. En premier lieu, le système doit s'entraîner sur une base d'apprentissage composée des exemples à reconnaître, dans la phase de test, les autres exemples constituent la base de test.

Notre système proposé a les caractéristiques suivantes: aucune interaction n'existe entre classifieurs; les classifieurs sont fixes et ne changent pas; les classifieurs utilisent les mêmes données en entrée et la base de données est découpée en paquets ou lots.

Il existe trois schémas de combinaison de classifieurs tels que: la combinaison séquentielle, parallèle et hybride.

Dans notre étude, nous avons choisi la combinaison parallèle pour les raisons suivantes:

- Facilité de la mise en œuvre;
- Ne nécessite pas une re-paramétrisation des autres classifieurs en cas de modification de l'ensemble d'apprentissage.

Dans ce type, les classifieurs opèrent indépendamment les uns des autres, chaque classifieur produit individuellement une sortie simultanée, leurs réponses seront combinées respectives pour produire une décision unique. L'ordre d'exécution des classifieurs est déterminé par le concepteur. Cette combinaison est très utilisée, parce que chaque classifieur qui satisfait le problème de la classification peut être utilisé normalement dans un système multi-classifieur.

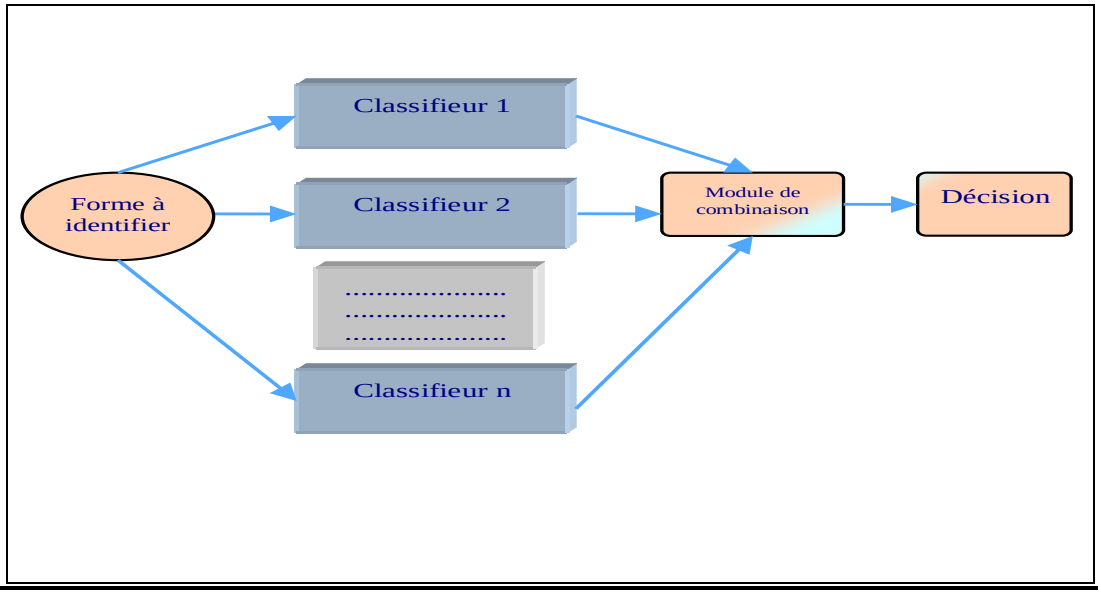

Figure 3.2. Schéma de combinaison parallèle de classifieurs

3.3.2. Choix et description des classifieurs

3.3.2.1. Choix de classifieurs

Le choix de classifieurs est une tâche difficile, ce qui a poussé les chercheurs à développer des méthodes pour aider les concepteurs. La méthode la plus simple et la plus utilisée appartient à la sélection statique, « ***Surproduire et choisir*** » (over produce and choose paradigm) qui consiste à générer des classifieurs différents en se basant sur les méthodes de création d'ensembles et à choisir ensuite le groupe de classifieurs dont les sorties peuvent être ensuite combinées et la combinaison produit le meilleur résultat. D'une manière plus simple, l'idée principale de cette méthode est de produire un ensemble initial large de classifieurs candidats, puis sélectionner un sous-ensemble qui est jugé le plus valable pour aboutir à des performances optimales.

Pour le faire, le procédé suit deux cycles :

- Construire l'ensemble de classifieurs de départ (over production);
- Choisir le sous ensemble le plus intéressant.

Nous avons construit un ensemble de classifieurs incrémentaux: Learn++, SVM de Cauwenberghs et Poggio, SVM de Diehl et Cauwenberghs [DIE 03] et l'arbre de décision incrémental ITI (Incremental Tree Inducer) [UTG 89]. Nous avons testé cet ensemble de classifieurs avant de faire la combinaison sur une base de données choisie (Iris) mais nous

avons trouvé que Learn++ de Polikar et al et SVM de Cauwenberghs et Poggio donnent des performances meilleures par rapport aux autres.

Le choix d'interprétabilité des algorithmes d'apprentissage dépend fortement de la tâche à résoudre et du désir du modèle. Ceci dit, nous avons préféré opter pour ces deux algorithmes pour la suite de notre travail.

3.3.2.2. Les séparateurs à Vastes Marges (SVM) incrémentaux de Cauwenberghs et Poggio [CAU 01]

Plusieurs versions incrémentales des SVM ont été proposées, les premières sont celles de Syed et al. [SYE 99] et de Ruping [RUP 01]. Le principe des SVM's incrémentaux est : de gérer les vecteurs supports de manière incrémentale lorsqu'une nouvelle donnée d'apprentissage est mal classée ou à l'intérieur de la marge. Nous avons choisir de détailler l'algorithme SVM de Cauwenberghs et Poggio parce qu'il permet de mettre à jour une SVM de deux manière incrémentale et décrémentale ceci signifie qu'il est bien adapté au dilemme (stabilité/plasticité).

Principe:

Cauwenberghs et Poggio [CAU 01] proposent un algorithme en ligne, récursif pour l'apprentissage d'un SVM. Au départ, un nouvel exemple est ajouté à la base d'apprentissage. Si cet exemple est bien classé par le SVM alors aucun changement ne sera effectué. Sinon une mise à jour est nécessaire en faisant des modifications sur les multiplicateurs de Lagrange tout en respectant les conditions de Karush Kuhn Tucker (KKT).

La base d'apprentissage est partitionnée en trois groupes :

- Le groupe des exemples bien classés D (points intérieurs) situés à l'intérieur de la frontière de décision.
- Le groupe des vecteurs supports S situés sur la frontière de décision.
- Le groupe des vecteurs erreurs U contient les points extérieurs situés à l'extérieur de la frontière de décision.

Après avoir initialisé l'hyperplan optimal du modèle au début de la classification, l'objectif est d'adapter, de manière séquentielle, cet hyperplan en fonction de l'évolution du modèle au fil de la séquence. Lors de la mise à jour de la frontière de décision, des exemples parmi ces trois groupes (D, S, U) peuvent changer d'état (lors de l'ajout des nouvelles données). Une procédure d'apprentissage décrémentale LOO (Leave One Out) est exécutée pour supprimer les anciennes données. L'idée de base est d'adapter la frontière de décision en ajoutant le nouvel exemple à la solution puis de retirer ceux des données trop anciennes tout en conservant les conditions de KKT satisfaites.

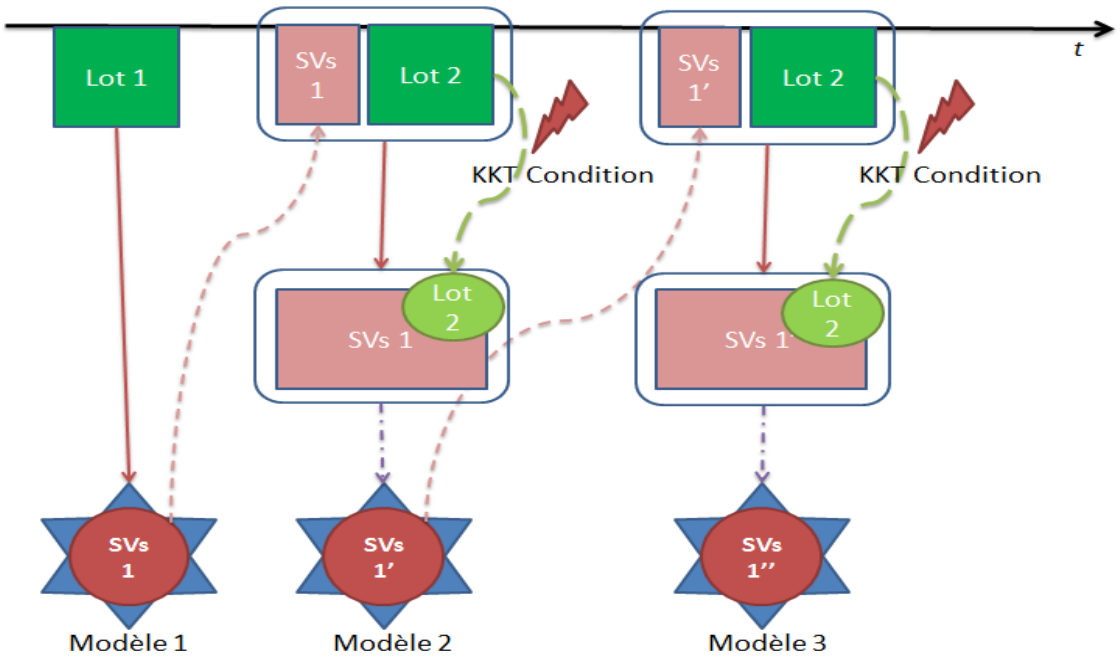

Figure 3.3 Méthode de Cauwenberghs et Poggio.[NGO 15]

Algorithme 3.1 (ISVM)

Posons F l'ensemble d'apprentissage et $H(x)$ la fonction séparatrice se réduit a une combinaison linéaire de la fonction noyau sur les données d'apprentissage. Les multiplicateurs de Lagrange sont obtenus en minimisant la fonction objective quadratique convexe sous les contraintes:

Minimiser $W=\frac{1}{2} \sum_{i=1}^{n} \sum j=1n\, \alpha_i \alpha_j\, k(x_i,x_j)-\sum_{i=1}^{n} \alpha_l + b\sum_{i=1}^{n} y_i \alpha_i$

Les conditions du 1er ordre sur le gradient de W mènent aux condition de KKT suivantes $g_i = \frac{\partial w}{\partial\, \alpha i} = \sum_{j=1}^{s} \alpha_j\, k(x_i,x_j)+b=H(x_i)$

$H(x_i)>0,\ \alpha_i=0;$

$H(x_i)=0,\ 0<\alpha_i<C;$ $\qquad \frac{\partial w}{\partial b} = \sum_{j=1}^{s} \alpha_{j\ -1}=0$

$H(xi)<0,\ \alpha_i=C;$ {C est le paramètre de régularisation}

- $0<\alpha_i<C$: on parle des vecteurs supports (ensemble S);
- $\alpha_i=C$: on parle des vecteurs erreurs (ensemble U);
- $\alpha_i=0$: on parle des vecteurs bien classés (ensemble D);
- g_i: quantité des dérivées partielles.

Procédure d'apprentissage incrémental :

Lorsque le nouvel exemple xi est ajouté à l'ensemble des points supports S, la fonction de décision est adaptée et mise à jour itérativement, c'est –à-dire que ses paramètres α_i, b sont mis à jour et recalculés de manière itérative et incrémentale. A chaque itération, $g_c=H_c(x_c)$ est recalculée jusqu'à ce que $g_c=H_c(x_c)=0$ tout en gardant les conditions KKT satisfaites. On doit alors définir les pas d'incrémentation Δg_i (avec $i=1,\ldots,d$), $\Delta\alpha_i$(avec $i=1,\ldots,s$), et Δb.

Pour utiliser le Lagrangien dans la résolution des équations d'optimisation sous contraintes, la matrice Jacobienne Q est utilisée.

Résumé de la procédure d'apprentissage incrémental d'un nouvel exemple x_c :

Algorithme 3.2 (procédure incrémentale)
Initialisation α_c à zéro
Si $g_c>0$, ajouter x_c à D .mettre à jour $G(G^{s+1} \leftarrow G_c^s)$, Fin
Si $g_c=0$, ajouter x_c à S, mettre à jour des paramètres α_j, b, R et G, Fin.
Si $g_c<0$, ajouter x_c à U,
Tant que $g_c<0$ faire $\alpha_c= \alpha_c+\Delta b$
Calculer β
Calculer Δb, puis $b=b+\beta b$
Pour chaque $x_j \in S$, calculer $\Delta\alpha_{je}$ puis $\alpha_j= \alpha_j+\Delta\alpha j$
Pour chaque $x_j \in D$, calculer Δg_i puis $g_i= g_i+\Delta g$
Vérifier c'est un exemple support se retrouve à l'intérieur de l'hyperplan ($\alpha_j \leq 0$).
Si oui, l'enlever de S et l'ajouter à D, et mettre à jour tous les paramètres.
Répéter jusqu'à ce que $g_c =0$

Procédure d'apprentissage décrémental:

L'apprentissage incrémental est le fait d'apprendre petit à petit, au fur et à mesure que les données arrivent. Par opposition, l'apprentissage décrémental est le fait de désapprendre, d'oublier petit à petit la connaissance issue des données d'apprentissage les plus anciennes [Bou 12]. La procédure de suppression des données anciennes est complémentaire avec la procédure d'ajout de nouvelles données.

Quand un exemple x_r est retiré de S, g_r sera retiré de G et z (ensemble des paramètres $\{b, \alpha_r\}$) sera mis à jour de façon décrémentale et la fonction de décision H_k sera ajustée jusqu'à ce que x_r soit à l'extérieur ($\alpha_r \leq 0$). La matrice R est mise à jour en éliminant de la matrice Q la colonne $r+1$ et la ligne $r+1$ (correspondant x_r à qui a été retiré). Quand un exemple x_r est retiré de D, seul G est mis à jour en lui retirant g_r.

Résumé de procédure d'apprentissage décrémental d'un exemple x_r :

Algorithme 3.3 (procédure décrémentale)
Si $g_r =0$, retirer x_r de F, et retirer g_r de G ($G \leftarrow G - g_r$), Fin.
Si $g_r =0$, retirer x_r de S, et donc de F.
Tant que $\alpha_r>0$ faire $\alpha_r= \alpha_r+\Delta\alpha$
Calculer Δb, puis, $b=b+\Delta b$
Pour chaque $x_j \in S$,

Calculer $\Delta\alpha_j$, puis $\alpha_j = \alpha_j - \Delta\alpha_j$

Pour chaque $x_i \in F$, calculer Δg_i alors $g_i = g_i - \Delta g_i$

Vérifier si un exemple intérieur $x_i \in D$ se trouve à l'extérieur de la frontière de décision ($g_i \leq 0$). Si oui, interrompe la procédure de suppression, et appliquer la procédure d'ajout sur x_i puis reprendre la procédure d'oubli jusqu'à ce que $\alpha_r = 0$.

3.3.2.3. Réseau de neurones incrémental Lean++ de Polikar et al. [POL 01]

Rappel de Learn++:

Learn++ [POL 01] est un algorithme d'apprentissage incrémental (voir section 2.9.4) d'un réseau de neurones inspiré de l'algorithme AdaBoost. Il se base sur le principe d'une combinaison de classifieurs faibles pour prendre une décision. Le système va entraîner plusieurs classifieurs sur plusieurs sous-ensembles de l'ensemble d'apprentissage. Les difficultés de cet algorithme résident dans la création des sous-ensembles d'apprentissage et la combinaison de ces classifieurs.

L'idée de Learn++ est de modifier la distribution des éléments dans le sous ensemble d'apprentissage afin de renforcer la présence des éléments les plus difficiles à classifier en fonction des erreurs du classifieur faible généré. Cette procédure est ensuite répétée avec un ensemble différent de données de la même base d'apprentissage et de nouveaux classifieurs sont générés. La sortie sera la combinaison des sorties des classifieurs en utilisant le vote majoritaire. Les classifieurs faibles sont des classifieurs qui fournissent une estimation grossière d'une règle de décision car ils doivent être très rapides à générer.

Algorithme 3.4 (Rappel de Learn++)

Entrée : Pour chaque base de données tirée de D_k, $k=1,2,\ldots\ldots,k$

-Séquence de m exemples d'apprentissage $S=[(x_1,y_1),(x_2,y_2),\ldots\ldots(x_m,y_m)]$

-Algorithme d'apprentissage faible *WeakLearn*.

- Entier T_k, spécifiant le nombre d'itérations.

pour $k=1,2,\ldots\ldots,k$ faire :

Initialiser $w1 = D_1(i) = 1/m$, $\forall i$ a moins qu'il y ait connaissance antérieure pour sélectionner autrement.

Pour $t=1,2,\ldots T_k$ faire :

1. Déterminer $D_t = w_t / \sum_{i=1}^{m} w_t(i)$ pour que D_t est une distribution
2. Choisir aléatoirement les sous-ensembles d'apprentissage TR_t et de test TE_t selon D_t
3. Faire appel à *WeakLearn* en utilisant TR_t
4. Récupérer une hypothèse $h_t : X \rightarrow Y$,
5. -Calculer l'erreur de $h_t : \varepsilon_t = \sum_{i:ht(xi)\neq yi} D_t(i)$ sur $S_t = TR_t + TE_t$

-Si $\varepsilon_t > ½$ alors $t = t - 1$ et supprimer h_t et aller à l'étape 2

-Sinon normaliser l'erreur comme $\beta_t = \varepsilon_t / (1 - \varepsilon_t)$

6. Obtenir l'Hypothèse composée : $H_t = arg\,max_{y\in Y} \sum_{t:ht(x)=y} log(1/\beta t)$

Erreur composée $E_t = \sum_{i:ht(xi)\neq yi} D_t(i) = \sum_{i=1}^{m} D_t(i)$ $[/H_t(x_i)\neq y_i/]$

-Si $E_t > 1/2$ alors $t=t-1$ supprimer H_t et aller à l'étape 2

$$w_{t+1}(i) = w_t(i) \times \{B_t, si\ H_t(x_i) = y_i\}$$

$$\{1, \quad autrement$$

$$= w_t(i) \times B_t^{1-[|H_t(x_i) \neq y_i|]}$$

7. Déterminer l'erreur composite normalisée : $\beta_t = E_t / (1-E_t)$

-Mettre à jour des poids des instances :

-Faire appel au vote majoritaire pondéré sur les hypothèses combinées H_t et produire l'hypothèse finale:

$$H_{final} = \arg\max_{y \in Y} \sum_{k=1}^{K} \sum_{t:H_t(x)=y} \log\ |(1/\beta_t)$$

3.3.3. Module de combinaison

Le choix de module de combinaison, ou bien la fonction de décision joue un rôle très important dans la conception d'un système multiclassifieurs (MCS). La sortie du MCS reflète la décision de tout l'ensemble en utilisant par exemple la méthode bayésienne, la somme pondérée, etc. Le mécanisme de combinaison sera plus efficace lorsque les classifieurs manifestent des comportements différents.

La méthode de combinaison que nous avons adoptée pour notre système est la méthode de la somme pondérée qui est définie comme une généralisation de la méthode Borda Count dans laquelle les rangs attribués par un classifieur sont pondérés par un coefficient indiquant l'importance accordée à celui-ci. Elle consiste à pondérer la somme des rangs en fonction de la confiance accordée au classifieur.

Dans notre système, chacun des deux classifieurs présentés précédemment donne en résultat une valeur x_{ki} correspondante à la classe de sortie $(k = 1..2)$ correspond au numéro du classifieur, i correspond au numéro de la classe pour chaque base de données.

Pour appliquer la somme pondérée, nous avons utilisé une pondération afin de représenter la notion de confiance accordée au classifieur [GAS 05]. Ce degré de confiance est connu a priori en phase d'apprentissage et en phase de test, pour cela on considère que :

- Soient $x1i, x2i$ les sorties associées à la classe i pour chaque classifieur;
- Soient $dc1, dc2$, les degrés de confiance de chacun des classifieurs déterminés par le taux de reconnaissance pendant la phase de test.

La méthode de somme pondérée est ensuite appliquée :

$$X_i = \frac{X_{1i} * dc1 + X_{2i} * dc2}{dc1 + dc2} \text{.....(3.1)}$$

Cette formule est appliquée pour toutes les sorties obtenues par chaque classifieur et pour tous les paquets de base de données.

3.4. Evaluation expérimentale d'ISVM-LEARN++

3.4.1. Description des bases de données

Dans le but de mieux évaluer l'algorithme ISVM-LEARN++ proposé dans ce chapitre, nous avons utilisé trois bases de données publiques disponibles dans l'UCI Machine Learning Repository. Le Tableau 3.2 présente une brève description de ces bases:

Base de données	Nombres d'instances	Nombre d'attributs	Type d'attributs	Valeurs manquantes	Nombre de classes
Haberman's Survival	306	3	Entier	Non	2
Blood Transfusion Service Center	748	5	Réel	Non	2
Ionosphere	351	34	Entier et réel	Non	2

Tableau 3.2. Description des bases de données

3.4.2. Évaluation des performances de classification

Afin de valider correctement le processus de classification, nous utilisons des mesures de performances sur les résultats de la classification. L'efficacité peut se définir selon plusieurs critères (voir section 1.4.4.2):

Le **taux de reconnaissance** correspond au terme justesse ou précision. C'est l'un des critères permettant d'évaluer les modèles de classification. De façon non formelle, le taux de reconnaissance désigne la proportion des prédictions correctes effectuées par le modèle. Formellement, le taux de reconnaissance est défini ainsi:

$$Taux\ de\ reconnaissance = \frac{Nombre\ de\ prédictions\ correctes}{Nombre\ total\ de\ prédictions} \text{.....(3.2)}$$

Pour une classification binaire, le taux de reconnaissance peut aussi être calculé en termes de positifs et de négatifs comme suit :

$$Taux\ de\ reconnaissance = \frac{VP+VN}{TP+TN+FP+FN} \text{.....(3.3)}$$

Taux d'erreur de classification

Succès: la classe prédite correctement (Vrai Positive (VP) / VN)

Erreur: la classe est incorrectement prédite (faux positifs (FP) / négatifs (FN))
Taux d'erreur de classification: la proportion des exemples mal classés sur l'ensemble total des exemples.

$$Taux\ d'erreur = \frac{FP+FN}{TP+TN+FP+FN} \text{.....(3.4)}$$

3.4.3. Résultats expérimentaux

Nous avons implémenté en premier lieu les deux classifieurs classiques: SVM linéaire et réseau de neurones perceptron multi couches (MLP). En second lieu, nous avons implémenté les deux classifieurs incrémentaux individuellement (SVM de cauwenberghs et Learn++) et leur combinaison et de comparer les performances de ces deux classifieurs et la performance de combinaison en faisant des expérimentations sur les trois bases de données (décrites précédemment).

Remarque: Dans la sous section suivante, nous allons présenter les résultats de tous les classifieurs sur la base de données ionosphère comme un exemple.

3.4.3.1. Pour les classifieurs classiques

Le premier classifieur classique est un réseau de neurones de type perceptron multi couches MLP (Multi Layer Perceptron).

Nous avons obtenu les résultats suivants:

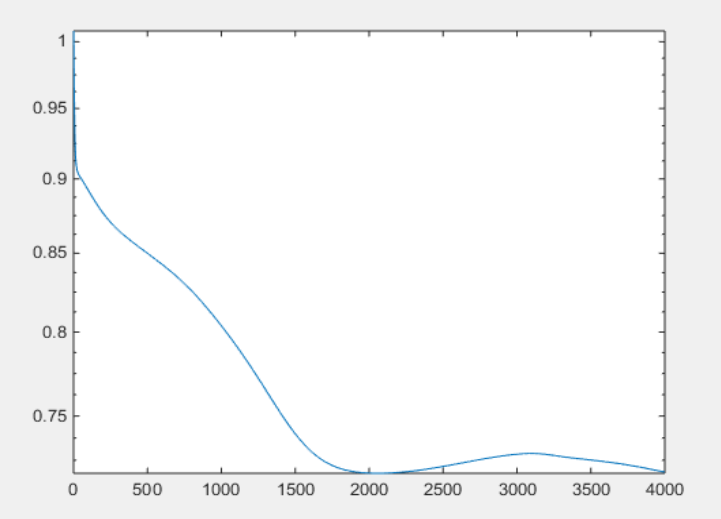

Figure 3.4. Taux d'erreur de MLP classique pour la base de données Ionosphère

Le deuxième classifieur classique est un SVM linéaire, le résultat de classification de la base de données ionosphère est illustré dans la figure 3.5:

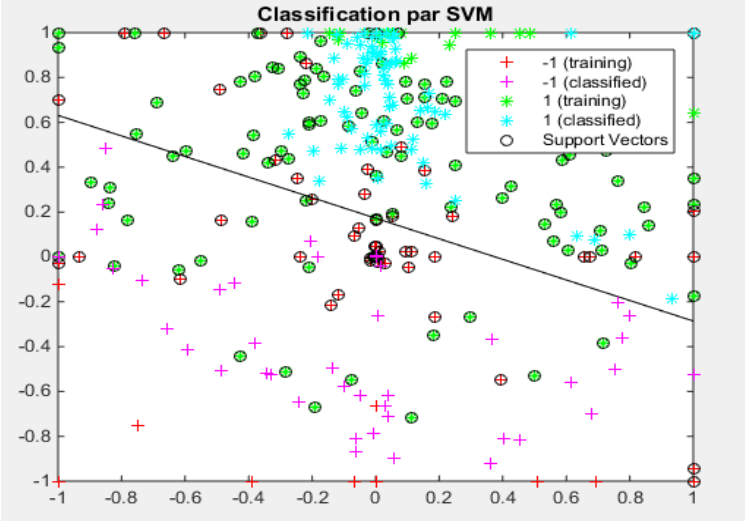

Figure 3.5. La classification par SVM linéaire pour la base de données Ionosphère

Le tableau 3.3 résume les résultats obtenus par l'application de MLP et SVM classiques sur les trois bases de données:

Bases de données	Taux de reconnaissance	
	SVM	MLP
Ionosphère	64.96	76.18
Blood Transfusion Service Center	75	86.21
Haberman's Survival	75.49	77.88

Tableau 3.3. Taux d'erreurs des classifieurs classiques

3.4.3.2. Pour les classifieurs incrémentaux

Nous avons divisé chaque base de données ionosphère en paquets de données et nous avons introduit à chaque classifieur un paquet et ainsi de suite....

La base de données est découpée en 2 paquets (BA1 et BA2) pour l'apprentissage et le 3ème pour le test (BT).

Le coefficient C est défini comme un paramètre de régularisation, il donne un compromis entre la marge et le nombre d'erreurs admissibles. C'est une constante qui représente un compromis entre la maximisation de la marge et la minimisation des erreurs.

Le noyau choisi est RBF, on choisit donc sigma =0.25 fixe, C=10:

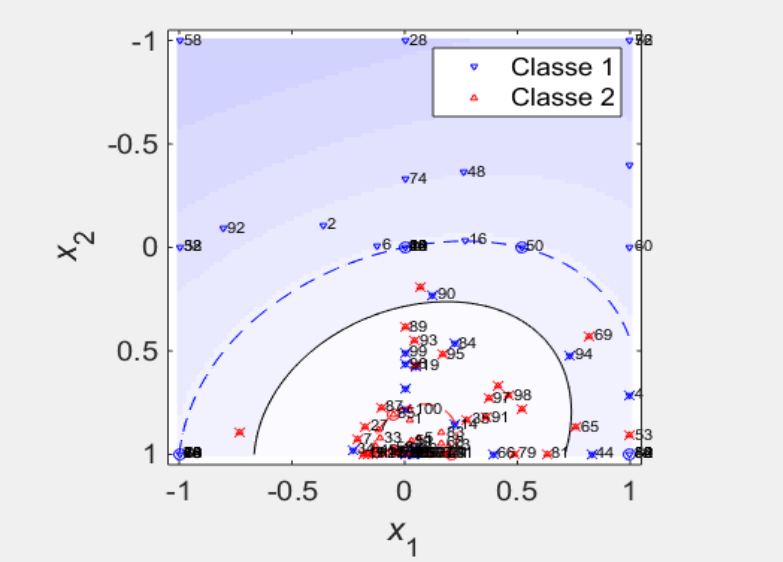

Figure 3.6. Classification par SVM incrémental BA1, C=10, Sigma=0.25, BA1=1-117, Vecteurs Erreurs=42, taux de reconnaissance=58%

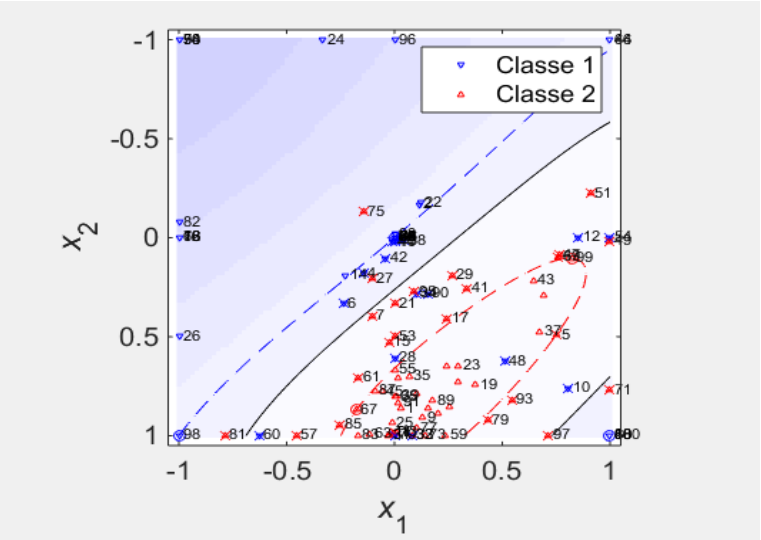

Figure 3.7. Classification par SVM incrémental BA2, C=10, Sigma=0.25, BA2=118-234, Vecteurs Erreurs=41, taux de reconnaissance=59%

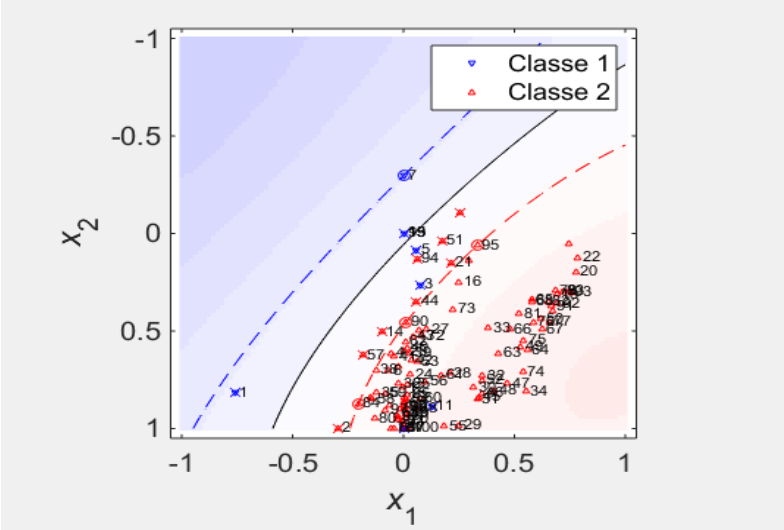

Figure 3.8. Classification par SVM incrémental BT, C=10, Sigma=0.25, BT=235-351, Vecteurs Erreurs=17, taux de reconnaissance=83%

Le tableau 3.4 résume les résultats obtenus par l'exécution de SVM incrémental et Learn++ sur les trois bases de données:

Pour la phase de test, nous avons pris 1/3 de chaque base de données, les taux de reconnaissances de chaque classe par rapport à chaque classifieur après la combinaison sont mentionnés dans le tableau suivant (tableau 3.4) :

Bases de données	SVM incrémental				Learn++		
	C (paramètre de régularisation)	Sigma	Vecteurs erreurs	Taux de reconnaissance (%)	T (Nombre d'itérations)	Taux d'erreur (%)	Taux de reconnaissance (%)
ionosphere_apprentissage1 (1_117)	10	0.25	42	58	3	28.53	71.47
ionosphere _ apprentissage2 (118_234)	10	0.25	41	59	3	16.38	83.62
Transfusion- apprentissage1 (1_88)	10	0.25	32	68	3	50.20	49.80
Transfusion- apprentissage2 (89_176)	10	0.25	20	80	3	38.43	61.57
Haberman- apprentissage1 (1_102)	10	0.25	20	80	3	21.67	78.33
Haberman_ apprentissage2 (103_204)	10	0.25	13	87	3	14.67	85.33

Tableau 3.4. Taux de reconnaissance et d'erreur des classifieurs incrémentaux (phase d'apprentissage)

Bases de données	SVM incrémental				Learn++		
	C (paramètre de régularisation)	Sigma	Vecteurs erreurs	Taux de reconnaissance (%)	T (Nombre d'itérations)	Taux d'erreur(%)	Taux de reconnaissance(%)
ionosphere _test (235_351)	10	0.25	17	83	3	9.32	90.68
Transfusion_test (177_264)	10	0.25	22	78	3	26.67	73.33
Haberman_test (205_306)	10	0.25	20	80	3	43.33	56.67

Tableau 3.5. Taux de reconnaissance et d'erreur des classifieurs incrémentaux (phase de test)

Interprétation des Résultats :

Pour le SVM incrémental: On voit dans les 2 premières lignes du tableau 3.4 que le nombre de vecteurs erreurs est grand (=42), ainsi que la marge augmente, donc la performance diminue =58%. Pour le 2ème paquet d'apprentissage, la performance est égale à 59%, elle est presque la même que pour le 1er paquet d'apprentissage. Si on observe le troisième cas (test), comme le montre la figure 3.8, on trouve que le nombre de vecteurs erreurs est 17, la marge est petite, la performance est meilleure (83%). Donc le meilleur cas est le troisième, on observe que le nombre d'erreurs égal 17% c'est-à-dire on a accepté d'avoir des points mal classifiés.

On a remarqué que les résultats de learn++ sont meilleurs que les résultats de SVM incrémental pour la base Ionosphère mais ce n'est pas toujours le cas pour les 2 autres bases.

3.4.3.3. Résultats de combinaison

Après avoir vu les résultats des deux classifieurs individuels, nous avons combiné les deux de manière parallèle en utilisant une combinaison par somme pondérée, nous avons pris les degrés de confiance de chacun des classifieurs qui indiquent les taux de reconnaissance pendant la phase de test, on appliquant la formule 3.1 sur la base de données ionosphère, on obtient les résultats suivants:

- Soient x_{1i}, x_{2i} les sorties associées à la classe i pour chaque classifieur qui prennent les valeurs mentionnées dans le tableau 3.5.
- Soient dc_1, dc_2, les degrés de confiance de chacun des classifieurs qui prennent les valeurs suivantes respectivement pour chacune des 3 bases de données (0.83- 0.9068), (0.78-0.7333), (0.80-0.5667).

Nous avons calculé les sorties pour chaque classifieur et pour chaque base de données, nous avons appliqué ensuite la méthode de la somme pondérée, en appliquant la formule (3.4) et nous avons obtenu les résultats suivants (voir le tableau 3.6):

Somme Pondérée1= $((0.83 * (\frac{44}{117}) + 0.83 * (\frac{56}{117})) + (0.9068 * (\frac{23}{117}) + 0.9068 * (\frac{94}{117})))/(0.83 + 0.9068) = 0.93$

Alors la performance obtenue après la combinaison des deux classifieurs sur la 1ère base de données est: 93% et 7% de rejet.

Somme Pondérée2= $((0.78 * (\frac{66}{88}) + 0.78 * (\frac{0}{88})) + (0.73 * (\frac{17}{88}) + 0.73 * (\frac{71}{88})))/(0.78 + 0.73) = 0.87$

Alors la performance obtenue après la combinaison des deux classifieurs sur la 2ère base de données est: 87% et 13% de rejet.

Somme Pondérée3= $((0.8 * (\frac{1}{102}) + 0.8 * (\frac{81}{102})) + (0.57 * (\frac{20}{102}) + 0.57 * (\frac{82}{102})))/(0.8 + 0.57) = 0.88$

Alors la performance obtenue après la combinaison des deux classifieurs sur la 3ère base de données est: 88% et 12% d'erreur.

Bases de données	SVM incrémental						Learn++						Combinaison par la somme pondérée (X)
	Vecteurs d'erreur	Taux de reconnaissance	Classe 1	N° de classe1	Classe 2	N° de classe2	Taux d'erreur	Taux de reconnaissance	Classe 1	N° de classe 1	Classe 2	N° de classe 2	
Ionosphère test 235_351	17	0.83	44	50	56	62	0.0932	0.9068	23	25	94	96	0.93
Transfusion test 177_264	22	0.78	66	83	0	4	0.2667	0.73	17	29	71	94	0.87
Haberman test 205_306	20	0.80	1	3	81	98	0.4333	0.57	20	31	82	99	0.88

Tableau 3.6. Résultats de la combinaison initiale

Selon notre architecture proposée (figure 3.1), la décision finale est la combinaison des combinaisons des classifieurs, alors on va appliquer la somme pondérée autre fois sur les résultats précédents (X du tableau 3.7):

Bases de données	**Combinaison des classifieurs (ISVM-Learn++)**		
	X	**Classe1**	**Classe2**
Ionosphère test 235_351	0.93	67	150
Transfusion test 177_264	0.87	83	71
Haberman test 205_306	0.88	21	163
Résultats de combinaison finale	**0.98**		

Tableau 3.7. Résultats de la combinaison finale

Classe1: Classe1 du 1er classifieur+ Classe1 du 2ème classifieur

Classe2: Classe2 du 1er classifieur+ Classe2 du 2ème classifieur

Somme pondérée finale=

$$\left(\left(0.93 * \left(\frac{67}{217}\right) + 0.93 * \left(\frac{150}{217}\right)\right) + \left(0.87 * \left(\frac{83}{154}\right) + 0.87 * \left(\frac{71}{154}\right)\right) + \left(0.88 * \left(\frac{21}{184}\right) + 0.88 * \left(\frac{163}{184}\right)\right)\right) / (0.93 + .0.87 + 0.88) = 0.98$$

Alors la performance obtenue après la combinaison des deux classifieurs sur la 3ère base de données est: **98%** et **2%** d'erreur.

Les résultats de performances des classifieurs individuels et du multi-classifieurs sont résumés dans le Tableau 3.8:

Classifieurs	**Taux de reconnaissance**
SVM incrémental et Learn++ (1ère BDD)	93%
SVM incrémental et Learn++ (2ère BDD)	87%
SVM incrémental et Learn++ (3ère BDD)	88%
ISVM-Learn++	98%

Tableau 3.8. Les résultats finaux de performances de combinaison

Interprétation des Résultats :

Le taux de reconnaissance a été augmenté avec la combinaison par 5% par rapport au meilleur classifieur et par 9% par rapport à la moyenne des trois combinaisons. On déduire qu'avec l'utilisation de la combinaison de combinaison des classifieurs, on bénéficie des puissances individuelles des classifieurs pour obtenir un taux de reconnaissance plus élevé.

3.4.4. Evaluation d'ISVM-Learn++ pour la reconnaissance de chiffres manuscrits

Dans le but de mieux évaluer la combinaison ISVM-Learn++ proposée dans le domaine de la reconnaissance de formes, nous avons utilisé une base de données de la reconnaissance de chiffres manuscrits afin de comparer nos résultats avec les résultats obtenus de Mohemmed et al. [MOH 12]: (voir section 3.1 et Tableau 3.1). Nous avons choisi d'utiliser ce travail de recherche dans la comparaison pour évaluer notre proposition car les auteurs utilisent les réseaux de neurones, en plus du fait que la base de données MNIST est très connue et utilisée dans le domaine de la reconnaissance de formes.

La base MNIST comporte les descriptions de chiffres manuscrits en format 28×28, soit 784 pixels, et les étiquettes associées. La base contient en tout 70 000 exemples, c'est-à-dire 7000 exemples de chaque chiffre. Les premiers 60000 forment l'ensemble d'apprentissage et les autres 10000 sont utilisés comme ensemble de test.

Base de données	**Taille d'image**	**Nombre de classes**	**Taille de l'ensemble d'apprentissage**	**Taille de l'ensemble de test**
MNIST	28×28	10	60000	10000

Tableau 3.9. Description de la base de données MNIST

Nous avons divisé le MNIST en six sous-ensembles d'apprentissage et un sous ensemble de test. Chaque sous-ensemble contient des images appartenant aux classes 0 à 9, avec 100 images dans chaque classe.

Les résultats de nos expérimentations, en utilisant les algorithmes existants et l'algorithme proposé et décrit dans ce chapitre: SVM classique, réseau de neurones classique, SVM incrémental, learn++ et ISVM-Learn++ proposé sont résumés dans les tableaux 3.10, 3.11, 3.12 respectivement.

Base de données	**SVM classique**		**SVM de C&P**	
MNIST	**Précision (%)**		**Précision (%)**	
	Apprentissage	**Test**	**Apprentissage**	**Test**
	95.67	89.64	96.7	89.52

Tableau 3.10. Performances des deux algorithmes : SVM classique et algorithme de C&P

Base de données	**MLP**		**Learn++**	
MNIST	**Précision (%)**		**Précision (%)**	
	Apprentissage	**Test**	**Apprentissage**	**Test**
	90.36	86.25	91.84	83.83

Tableau 3.11. Performances des deux algorithmes : MLP et Learn++

Base de données	**ISVM-Learn++**		**SPAN de Mohemmed et al. [MOH 12]**	
MNIST	**Précision (%)**		**Précision (%)**	
	Apprentissage	**Test**	**Apprentissage**	**Test**
	97.05	89.12	92	86.6

Tableau 3.12. Performances des deux algorithmes : ISVM-Learn++ et SPAN

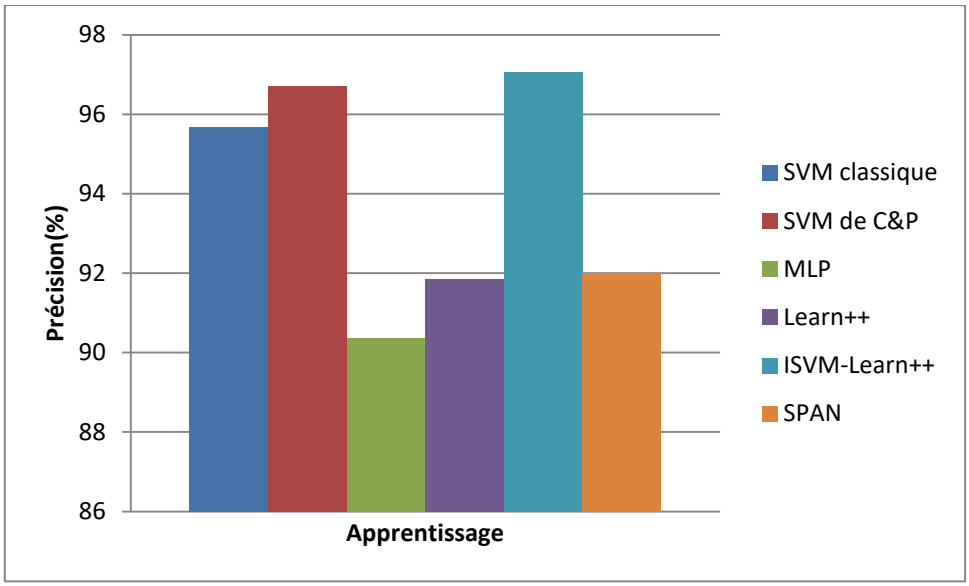

Figure 3.9. Comparaison de la précision de la base de données d'apprentissage MNIST

Comme nous pouvons le voir dans la figure 3.9, les meilleurs résultats de la précision de la base d'apprentissage sont obtenus par ordre décroissant par: notre algorithme proposé ISVM-Learn++, SVM de C &P, SVM classique, SPAN, Learn++. ISVM-Learn++ produit le taux d'erreur la plus faible pendant l'apprentissage (2.95%).

3.5. Conclusion

Dans ce chapitre, nous avons présenté la synthèse de notre étude bibliographique consacrée aux travaux relatifs à l'apprentissage supervisé incrémental dans le domaine de la reconnaissance des formes.

Cette étude nous a permis d'effectuer un ensemble de constats, particulièrement celui de l'intérêt de combiner des classifieurs incrémentaux. Dans cette optique, nous avons proposé la combinaison ISVM-Learn++ de deux classifieurs incrémentaux de référence: SVM incrémental de Cauwenberghs et Poggio [CAU 01] et Learn++ de Polikar et al. [POL 01].

Nous avons évalué notre proposition expérimentalement sur 3 bases de données publiques de l'UCI machine learning repository et une base de données de reconnaissance de formes (chiffres manuscrits).

Les résultats obtenus nous ont permis de confirmer la supériorité des performances de la combinaison proposée par rapport aux classifieurs individuels. Pour la reconnaissance de chiffres, les performances sont meilleures que celles d'un algorithme incrémental évalué sur la même base de données dans la littérature.

CHAPITRE 4

CONTRIBUTIONS DANS LE CLUSTERING INCREMENTAL

Ce chapitre présente la deuxième partie de nos contributions dans le cadre de l'apprentissage incrémental, plus précisément, le clustering ou apprentissage non supervisé. En effet, nous proposons une version améliorée de l'algorithme de clustering incrémental DBSCAN (Density-Based Spatial Clustering of Applications with Noise) que nous avons intitulée AMF-IDBSCAN [CHE 19c] et qui est basée sur le canopy clustering et la technique de filtrage médian adaptatif AMF (Adaptative Median Filtering).

Dans la section 4.2, nous introduisons la notion de filtrage des images qui est la source d'inspiration de notre proposition puis nous nous concentrons sur le filtrage médian adaptatif (AMF) dans la section 4.3.

Avant de détailler notre proposition, nous présentons dans la section 4.4 une synthèse de différentes propositions d'amélioration de l'algorithme DBSCAN dans la littérature puis nous décrivons, dans la section 4.5, celles que nous avons choisies pour une évaluation comparative.

Le reste du chapitre est dédié à la description détaillée de notre proposition ainsi que son évaluation.

4.1. Introduction

A la lumière de l'état de l'art présenté dans les premiers chapitres, nous pouvons placer la seconde partie de nos contributions dans le cadre des méthodes de clustering incrémental à base de densité. En effet, nous nous intéressons à l'algorithme DBSCAN (Density-Based Spatial Clustering of Applications with Noise) (voir section 1.4.3.4) qui permet l'identification de classes, c'est à dire le regroupement des objets d'une base de données en sous-classes significatives.

Les applications aux bases de données conséquentes augmentent les exigences des algorithmes de clustering telles que:

- Un minimum de connaissances sur les domaines afin de déterminer les paramètres d'entrée car les valeurs appropriées ne sont pas souvent connues à l'avance lorsque l'on travaille avec des bases de données importantes;
- La découverte de cluster de forme arbitraire car les formes de clusters dans les bases de données spatiales peuvent être sphérique, étiré, linéaire, allongé, etc.
- Une bonne efficacité sur les larges bases de données, c'est à dire celle contenant plus que quelques milliers d'objets.

Les raisons d'utiliser DBSCAN sont:

- Il permet de découvrir des clusters de forme arbitraire;
- Cet algorithme requiert seulement 2 paramètres d'entrée ($Eps\ and\ Minpts$) afin que l'utilisateur puisse spécifier une valeur appropriée;

- Il réduit l'espace de recherche et facilite la mise à jour incrémentale des clusters ;
- Il améliore le processus de mise en cluster en partitionnant de manière incrémentale l'ensemble de données afin de réduire l'espace de recherche du voisinage à une partition plutôt qu'à l'ensemble des données;
- Il est flexible et robuste aux données bruitées;
- Il s'adapte bien aux différents ensembles de données sans aucune information a priori [LIU 17];
- L'approche statique DBSCAN ne s'adapte pas avec les bases de données multidimensionnelles et volumineuses;
- Le DBSCAN avec le concept incrémental optimise dans le temps et l'effort alors que le DBSCAN statique souffre de certains inconvénients et ces problèmes sont principalement rencontrés dans les grandes bases de données dynamiques [SUT 13];

Dans ce chapitre, nous proposons un algorithme appelé AMF-IDBSCAN, une version améliorée du DBSCAN. Notre algorithme comprend trois phases principales. Après avoir importé la base de données d'origine en entrée, elle est d'abord prétraitée pour préparer l'étape de mise en cluster et pour réduire le volume du jeu de données à l'aide de la méthode canopy clustering. Ensuite, l'algorithme DBSCAN classique est appliqué aux résultats de la première étape et stocke le résultat dans une autre base de données. L'algorithme DBSCAN incrémental est appliqué ensuite à l'ensemble de données incrémental. La technique de filtre médian adaptatif est appliquée après sur les résultats de l'étape précédente pour supprimer le bruit et les valeurs aberrantes. Ensuite, les résultats sont comparés en évaluant également les performances.

Nous avons sélectionné la technique de filtre adaptatif médian (AMF) parmi les différentes techniques de filtrage car il supprime le bruit tout en préservant les détails de forme [VIJ 10]. La technique AMF est utilisée pour remplacer les valeurs aberrantes générées par DBSCAN incrémental par un cluster contenant un seul objet.

Nous avons choisi la méthode de canopy clustering pour le prétraitement des données car (1) elle est efficace lorsque le problème est important (2), elle peut considérablement réduire le nombre de calculs de distances requis pour la mise en cluster, en commençant par la partition des données en sous-ensembles qui se chevauchent, et alors seulement mesurer les distances entre des paires de points de données appartenant à un sous-ensemble commun [MCC 00].

Dans les sections suivantes, nous décrivons l'algorithme proposé AMF-IDBSCAN ainsi que les différents éléments qui le composent. Rappelons que les points forts des algorithmes fondés sur la densité est leur capacité à traiter de nouvelles données insérées, ou des données retirées de la base de données initiale (données obsolètes ou aberrantes). Ceci consiste à mettre à jour les clusters déjà obtenus sans avoir à relancer le processus global de classification. L'idée fondamentale est la suivante: affecter de nouvelles données à leurs clusters adéquats au moment où elles arrivent au système ou réarranger la partition quand des données existantes sont retirées de la base d'apprentissage. Ceci est réalisé avec un temps d'exécution minimisé tout en maintenant les propriétés de DBSCAN.

4.2. Filtrage

Dans notre proposition, nous avons utilisé l'idée et le principe de filtrage des images pour éliminer le bruit qui sont les données aberrantes dans notre classification. Nous abordons cette section à l'explication de la méthode AMF utilisée. Avant de commencer, il faut savoir c'est quoi le filtrage?

Une image dégradée peut avoir une cible négative sur son interprétation par l'être humain ainsi que sur ses utilisations dans des différents domaines. Pour supprimer l'information parasite que contient l'image, une opération est appliquée appelée « le filtrage ».

Le filtrage est une approche qui consiste à éliminer la présence d'information parasite qui s'ajoute de façon aléatoire à l'image, ce qui conduit à améliorer sa qualité [BOL 95]. Comme nous pouvons dire, le filtrage est une opération fondamentale en traitement d'image, il permet d'améliorer la perception de certains détails, de réduire le bruit, de compenser certains défauts du capteur, etc.

L'amélioration des images est essentiellement obtenue par la méthode de filtrage, qui consiste à modifier la valeur des pixels d'une image dans le but d'améliorer son aspect. Il s'agit de créer une nouvelle image en se servant des valeurs des pixels de l'image d'origine [BEN 17].

Il existe trois catégories de filtrages:

- Filtrage morphologique.
- Filtrage linéaire.
- Filtrage adaptatif.

4.2.1. Filtrage morphologique

Le concept de filtrage morphologique fait appel à des notions mathématiques morphologiques qui sont des transformations de type « Tout ou rien », qui utilisent directement les images numériques et ils sont développés à base d'ouverture et de fermeture « Morphologique » qui font appel aux notions de « Dilatation » et « Erosion ». [COS 89]

4.2.2. Filtrage linéaire

Il convertit un ensemble de données d'entrée en sorties suivant une opération mathématique nommée convolution. Quand il s'agit de données numérisées comme dans le cas du traitement d'image, la relation entre les valeurs des pixels de sortie et celle d'entrée est décrite par un tableau de nombres, souvent carré, appelé matrice de convolution. Le temps de calcul est généralement réduit lorsqu'on veut fractionner un filtre en deux filtres dont la convolution mutuelle permet de le reconstituer.

Cette technique provient de l'extension des méthodes mises au point pour le traitement du signal au traitement des images numériques. Il existe deux méthodes utilisées pour ce type de filtrage: [COS 89]

- Filtrage linéaire global qui consiste à prendre en compte la totalité des pixels de l'image au départ pour calculer chaque pixel de la nouvelle image, en utilisant les opérations sur les histogrammes ou les opérations qui nécessitent de passer dans l'espace de Fourier.
- Filtrage local qui consiste à prendre en compte seulement le voisinage du pixel correspondant dans l'image d'origine pour calculer chaque pixel de la nouvelle image, en choisissant le voisinage carré et symétrique autour du pixel considéré. Ces voisinages sont donc assimilables à des tableaux à deux dimensions (matrices) de taille impaire.

4.2.3. Filtrage adaptatif

Il peut être classé en deux catégories: les filtres d'ordre et les filtres de position. Les filtres d'ordre classent les niveaux de gris d'un voisinage et sélectionnent parmi ces quantités une certaine valeur, les filtres de position considèrent des sous voisinages du point.

Les filtres d'ordre sont de deux types: filtrage par la médiane (filtre médian) et filtrage par le plus proche voisin radiométrique. [COS 89]

Pour le filtrage par le plus proche voisin radiométrique, on attribue au pixel central la valeur moyenne des K pixels voisins dont les valeurs radiométriques « niveaux de gris » sont les plus proches.

Exemple:

$$\begin{matrix} 1 & 1 & 2 \\ 2 & 3 & 1 \\ 3 & 4 & 2 \end{matrix}$$

Pour K = 4,

$$\frac{(2+2+3+4)}{4} \approx 3$$

Dans ce cas, le point central gardera sa valeur.

Les filtres de position utilisent des voisinages 5×5. Le sous voisinage le plus homogène est sélectionné et son niveau de gris moyen est affecté au point central

$$\begin{matrix} 3 & 3 & 4 & 5 & 6 \\ 3 & 4 & 3 & 6 & 7 \\ 3 & 3 & 4 & 8 & 3 \\ 9 & 2 & 3 & 2 & 1 \\ 8 & 5 & 3 & 2 & 6 \end{matrix}$$

La valeur du point central passera de 4 à 3.

Les filtres de position présentent l'inconvénient d'avoir recours à un voisinage très grand (*en nombre de pixels)*, ce qui nécessite un temps d'exécution long c'est pour cette raison que nous avons préféré choisir un filtre d'ordre. Dans la section suivante, nous nous intéressons particulièrement au filtrage par la médiane qui est la technique utilisée dans notre proposition.

4.3. Filtrage par la médiane adaptative

Afin de comprendre le filtrage médian adaptatif, il faut d'abord comprendre ce qu'est un filtre médian et ce qu'il fait. L'opération principale appliquée dans le domaine du traitement d'images numériques est la suivante: à chaque pixel d'une image numérique on place un voisinage autour de ce point, on analyse les valeurs de tous les pixels du voisinage selon un algorithme, puis on remplace la valeur du pixel d'origine avec un autre basée sur l'analyse effectuée sur les pixels du voisinage. Le voisinage se déplace ensuite successivement sur chaque pixel de l'image, répétant le processus.

4.3.1. Principe du filtrage médian

La médiane est la valeur qui partage une population en deux parties de la même taille selon le tri par valeurs ordonnées de la variable considérée. Ce filtre est utilisé pour réduire des pixels isolés, d'une valeur très différente de leur voisinage. Il affecte à chaque pixel la valeur médiane de ses voisins de façon à éliminer les points isolés. Il est utilisé pour réduire le bruit dans une image, un peu comme le filtre moyen. Cependant, il est meilleur qu'un filtre moyen dans la préservation des détails utiles de l'image. Ce type de filtres appartient à la classe des filtres de lissage conservant les contours, qui sont des filtres non linéaires. Cela signifie que pour deux images $A(x)$ et $B(x)$:

$$median[A(x) + B(x)]) \neq median[A(x)] + median[B(x)] \ldots\ldots\ldots (4.1)$$

Ces filtres lissent les données tout en conservant les petits détails de l'image. La médiane correspond à la valeur médiane de toutes les valeurs des pixels du voisinage. La médiane est différente de la moyenne; c'est un "indicateur central" plus fort que la moyenne. En particulier, la médiane est affectée par un petit nombre de valeurs divergentes parmi les pixels du voisinage. Par conséquent, le filtre médian est très efficace pour éliminer divers types de bruit [CHA 16].

Comme le montre la figure 4.1 qui illustre un exemple de filtrage médian, ce filtre prend chaque pixel de l'image et garde ses voisins proches pour décider s'il est ou non représentatif de son environnement. Au lieu de remplacer la valeur de pixel par la moyenne des valeurs de pixels voisins, elle la remplace par la médiane de ces valeurs. La médiane est calculée en triant d'abord toutes les valeurs de pixel du voisinage dans l'ordre numérique, puis en remplaçant le pixel considéré par la valeur de pixel du milieu. (Si le voisinage considéré contient un nombre pair de pixels, la moyenne des deux valeurs de pixel du milieu est utilisée.)

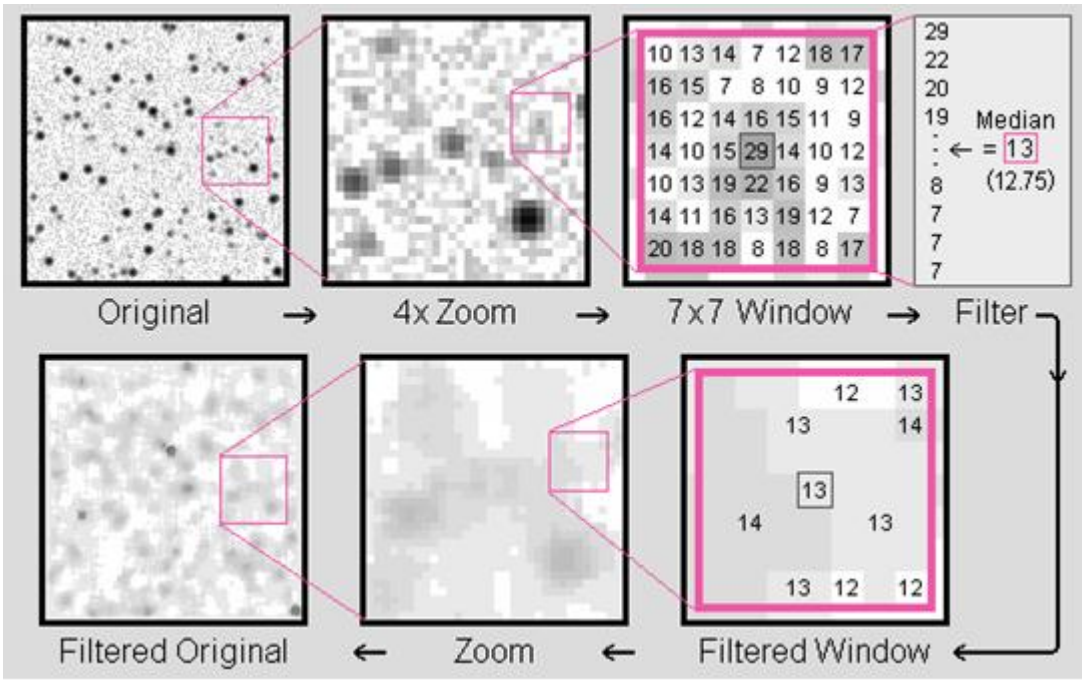

Figure 4.1. Exemple de filtrage médian [PEN 04]

Ainsi, le principe de filtre est de trier les pixels voisins par ordre croissant et de prendre la médiane.

La figure 4.2 illustre un exemple de calcul de médiane pour la valeur de pixel centrale 150, elle est non représentative des pixels alors elle est remplacée par la valeur médiane: 124, la fraction de 3 × 3 carrés est utilisé ici - les quartiers plus grands produiront un lissage plus sévère.

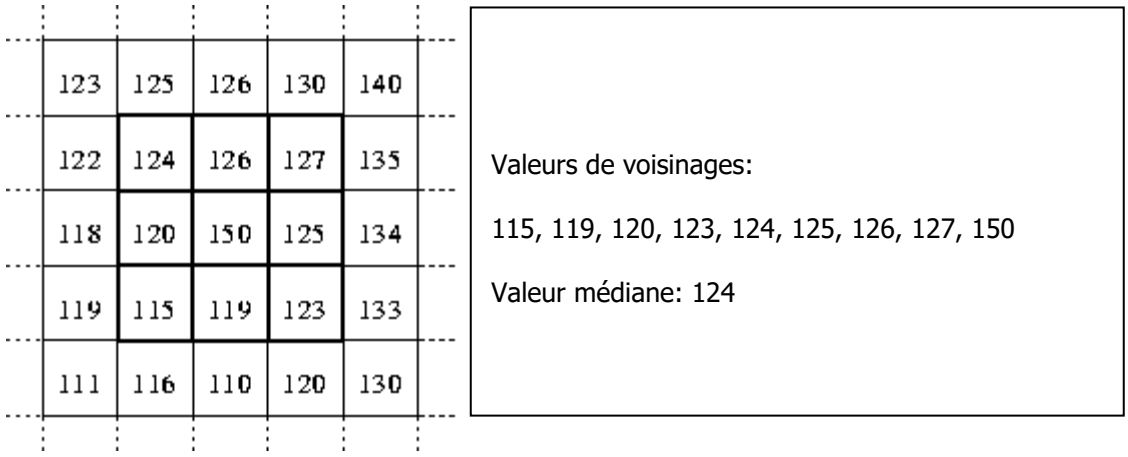

Figure 4.2. Exemple de calcul de la valeur médiane d'un voisinage de pixels [PEN 04]

Bien que le filtre médian soit une technique utile de lissage et d'amélioration des images non linéaires. Il présente également certains inconvénients: il élimine à la fois le bruit et le détail car il ne peut pas faire la différence entre les deux. Tout ce qui est de taille relativement petite par rapport à la taille de fenêtre (région) aura un impact qui minimise la valeur de la médiane et sera donc filtré. En d'autres termes, le filtre médian ne peut pas distinguer les détails les plus fins du bruit.

Par conséquent, le filtrage médian adaptatif a été largement appliqué en tant que méthode avancée par rapport au filtrage médian standard [PEN 04]. **Le filtre médian adaptatif** effectue un traitement spatial pour déterminer les pixels d'une image affectés (touchés) par le bruit impulsionnel. Le filtre médian adaptatif classe les pixels en tant que bruit en comparant chaque pixel de l'image à ses pixels voisins. La taille de la fenêtre est réglable, ainsi que le seuil pour la comparaison. Un pixel qui est différent de la majorité de ses voisins et qui n'est pas structurellement aligné avec les pixels auxquels il ressemble, est étiqueté comme du bruit impulsif. Ces pixels de bruit sont alors remplacés par la valeur médiane des pixels du voisinage qui ont passé le test d'étiquetage de bruit.

Les objectifs de l'utilisation de filtre médian sont:

1) Supprimer le bruit impulsif
2) Lissage des autres bruits
3) Réduire la distorsion, comme l'amincissement ou l'épaississement excessif des bordures de l'objet

4.3.2. Algorithme du filtrage médian adaptatif

Le filtre médian adaptatif modifie la taille de Sxy (la taille de fenêtre) pendant le fonctionnement.

Notations utilisées

$Zmin$ = valeur minimale prise par un bruit dans Sxy

$Zmax$ = valeur maximale prise par un bruit dans Sxy

$Zmed$ = valeur médiane de Sxy

Zxy = le bruit correspondant aux coordonnées (x, y)

$Smax$ = taille maximale autorisée de Sxy

Algorithme 4.1 (AMF)

Niveau A: $A1 = Zmed - Zmin$

$A2 = Zmed - Zmax$

Si $A1 > 0$ ET $A2 < 0$, Aller à B

Sinon augmenter la taille de la fenêtre

Si la taille de la fenêtre est $< Smax$, répéter le niveau A

Sinon la sortie est Zxy

Niveau B: $B1 = Zxy - Zmin$

$B2 = Zxy - Zmax$

Si $B1 > 0$ ET $B2 < 0$, Zxy est la sortie

Sinon la sortie est $Zmed$

- **Explication**

Niveau A: SI $Zmin < Zmed < Zmax$, et

- $Zmed$ n'est pas une impulsion

(1) aller à B pour tester si Zxy est une impulsion (bruit impulsif)...

AUTRE

- $Zmed$ est une impulsion

(1) la taille de la fenêtre est augmentée et

(2) le niveau A est répété jusqu'à ce que ...

(a) $Zmed$ n'est pas une impulsion, aller au niveau B ou

(b) $Smax$ atteint: la sortie est Zxy (le bruit reste)

Niveau B: SI $Zmin < Zxy < Zmax$, et

- Zxy n'est pas une impulsion

(1) la sortie est Zxy (réduction de la distorsion)

AUTRE

- soit $Zxy = Zmin$ ou $Zxy = Zmax$

(2) la sortie est $Zmed$ (filtre médian standard)

- $Zmed$ n'est pas une impulsion (du niveau A)

Le filtre médian standard ne fonctionne pas bien lorsque le bruit impulsif est:

a. Supérieur à 0,2, alors que le filtre médian adaptatif peut mieux gérer ces bruits
b. Le filtre médian adaptatif préserve les détails et lisse le bruit non impulsif, contrairement au filtre médian standard.

4.4. Synthèse de travaux des versions améliorées de DBSCAN

Suite à notre étude bibliographique de différentes propositions d'amélioration de DBSCAN, nous avons synthétisé certains des travaux dans le tableau 4.1. La première colonne contient des informations sur les auteurs et la référence du travail en question, alors que la 2ème colonne indique le nom de l'algorithme. La 3ème colonne présente le fonctionnement (définition) de chaque version améliorée:

Auteur et référence	Algorithme	Fonctionnement
Ram et al. [RAM 10]	DVBSCAN	Algorithme à densité variée, capable de gérer la variation de densité locale au sein d'un cluster. Le concept de moyenne de densité de cluster et de variance de densité de cluster sont introduits pour l'expansion de cluster.
Birant et Kut [BIR 07]	ST-DBSCAN	Clustering basé sur la densité spatio-temporelle. Il permet de regrouper les données en clusters en fonction de ses attributs spatiaux, non spatiaux et temporels. Il améliore DBSCAN de trois manières, de sorte qu'il regroupe les données spatio-temporelles correspondant à des attributs non spatiaux, spatiaux et temporels, détecte les clusters de densités différentes et résout les conflits dans les objets de bordure.
Liu et al. [LIU 07]	VDBSCAN	Clustering spatial basé sur une densité variable. Il permet de regrouper l'ensemble de données ayant des densités différentes. L'idée de base consiste à choisir des paramètres appropriés pour différentes densités à l'aide du tracé k-dist et à appliquer DBSCAN pour chaque paramètre choisi.
Elbatta [ELB 12], Elbatta at Ashour [ELB 13]	DMDBSCAN	Un nouvel algorithme proposé pour analyser les ensembles de données de densité variée. L'idée principale est qu'il utilise une méthode dynamique pour trouver la valeur appropriée d'Eps pour chaque niveau de densité de l'ensemble de données.
Viswanath et Pinkesh [VIS 06]	L-DBSCAN	Technique de clustering hybride, où L représente les leaders. Cet algorithme utilise le concept de Leader qui extrait les prototypes avec leurs voisins (enregistrement leaders) pour l'ensemble de données donné. Chaque enregistrement de leader contient le leader avec son voisinage situé à une distance égale ou inférieure au seuil de distance donné.
Uncu et al. [UNC 06]	GRIDBSCAN	Cet algorithme résout le problème de l'incapacité de DBSCAN à reconnaître des clusters de densités différentes dans la base de données spatiale par la sélection des grilles appropriées pour une densité homogène dans chaque grille, et la fusion des cellules de densités similaires et l'identification des paramètres d'entrée pour chaque grille.
Liu [LIU 06]	FDBSCAN	Cet algorithme a été introduit pour surmonter certaines des limitations de DBSCAN, notamment: (i) sa vitesse lente (décélération dans la recherche de voisinage due aux comparaisons impliquées pour chaque objet); et (ii) fixer la valeur seuil. Il s'agit d'un algorithme efficace dans le

		temps car il diminue le temps de calcul en ignorant les objets de région qui sont déjà regroupés.
He et al. [HE 11]	MR-DBSCAN	Une version parallèle de DBSCAN de manière MapReduce. Elle fournit une méthode pour diviser un grand ensemble de données en plusieurs partitions en fonction des dimensions de données. Cet algorithme basé sur un partitionnement spatial basé sur les coûts pour les données fortement asymétriques.
Wang et al. [WAN 16]	M-DBSCAN	Multi-Level DBSCAN. Dans cet algorithme, la recherche d'un objet voisin ne dépend pas d'un rayon fixé. La définition du rayon voisin est effectuée sur la base de la distribution des données autour du noyau en utilisant l'écart type et les valeurs moyennes.
Swathi et Thiagarasu [SWA 17]	FI-DBSCAN	Cet algorithme incorpore une arborescence ultra-métrique d'items fréquents avec regroupement spatial d'applications basé sur la densité (DBSCAN) et sur la structure MapReduce.
Mai et al. [MAI 16]	AnyDBC	un DBSCAN qui utilise à tout moment un apprentissage actif pour apprendre la structure de cluster actuelle, il ne peut donc sélectionner que les objets nécessaires pour développer les clusters.
Bakr et al. [BAK 15]	IDBSCAN	Une version améliorée de l'algorithme incrémental DBSCAN pour la construction et la mise à jour incrémentale de clusters de forme arbitraire dans des ensembles de données étendus. Il améliore le processus de clustering incrémental en limitant l'espace de recherche aux partitions, par opposition à l'ensemble de données entier, ce qui entraîne des améliorations significatives des performances par rapport aux algorithmes de clustering incrémental pertinents.
Yada et Sharma [YAD 16]	IDBSCAN	Un DBSCAN incrémental qui est fusionné avec une technique de suppression du bruit et de détection des valeurs aberrantes appropriée, inspirée de la méthode du box plot. Il a utilisé entre mesures de réseau et régions denses pour cadrer le dernier nombre de clusters. Cet algorithme partitionne de manière incrémentale l'ensemble de données pour réduire l'espace de recherche à chaque segment. Il structure et met à jour de manière incrémentale les régions denses dans chaque segment, en prenant en compte les zones épaisses imaginables dans chaque segment, l'algorithme utilise une mesure entre réseaux vers des régions denses pour encadrer le dernier nombre de clusters.

Tableau 4.1. Synthèse de travaux des versions améliorées de DBSCAN

4.5. Algorithmes utilisés pour la comparaison

L'un des principaux problèmes de DBSCAN est qu'il présente une grande variation de densité au sein d'un cluster. C'est pour cette raison que nous avons choisi de sélectionner deux algorithmes pour faire la comparaison avec notre algorithme proposé, l'algorithme DMDBSCAN d'Elbatta [ELB 13] et l'algorithme DBSCAN incrémental de Bakr et al. [BAK 15]. Le choix de ces

deux algorithmes est dû à leur proximité de principe par rapport à l'algorithme proposé. Dans cette section, nous allons expliquer chaque algorithme:

4.5.1. Algorithme DMDBSCAN

Afin de résoudre le problème d'utilisation d'une valeur globale du paramètre Eps pour toutes les densités de l'ensemble de données. Une nouvelle méthode DMDBSCAN a été proposée par Elbatta [ELB 13]. C'est une méthode dynamique qui vise à rechercher la valeur appropriée d'eps pour chaque niveau de densité de l'ensemble de données.

Cette méthode présente des avantages tels que:

1. Les clusters sont faciles à comprendre;
2. Elle ne se limite pas aux formes des clusters.

L'idée de base de DMDBSCAN est de trouver les valeurs appropriées du paramètre eps pour différents niveaux de densités selon le graphe $k\text{-}dist$. L'algorithme DBSCAN traditionnel est utilisé en premier lieu pour rechercher les clusters. Pour chaque valeur d'eps, l'algorithme DBSCAN est adopté pour rechercher tous les clusters par rapport au niveau de densité correspondant. Ensuite, à l'étape suivante, tous les points mis en cluster sont ignorés. Le résultat final évitera de marquer les zones les plus denses et les plus comptées comme un seul cluster.

Pour déterminer les paramètres eps et $MinPts$, le comportement de la distance du point au $k^{ème}$ voisin le plus proche, appelé $k\text{-}dist$ est examiné. Ces $k\text{-}dists$ sont calculés pour tous les points de données pour certains (k), puis le graphe trie les valeurs en ordre croissant, ce qui entraine un changement net dans le graphe tracé. Ce changement à la valeur de $k\text{-}dist$ correspond à une valeur appropriée de eps pour chaque niveau de densité de l'ensemble de données.

L'algorithme DMDBSCAN [ELB 13] pour trouver un eps approprié pour chaque niveau de densité est présenté sous forme de pseudo-code dans l'algorithme 4.2:

Algorithme 4.2 (DMDBSCAN)	
Objectif	Trouver les valeurs appropriées d'eps
Entrées	Ensemble de données de taille n
Sortie(s)	Eps pour chaque densité variée
Procédure	Pour $i{:=}1$ à n faire Pour $j{:=}1$ à n faire $D(i,j) := trouver\ distance(xi, xj)$ Trouver les valeurs minimales des distances au plus 3 proches voisins Fin pour; Fin pour; Trier les distances par ordre croissant et tracer pour trouver chaque valeur; eps correspond au changement critique des courbes.

4.5.2. Algorithme IDBSCAN

Un algorithme de classification incrémental basé sur la densité est introduit par Bakr et al. [BAK 15] pour créer et mettre à jour, de manière incrémentale des clusters dans des ensembles de données. L'algorithme partitionne progressivement les données pour réduire l'espace de recherche sur chaque partition au lieu d'analyser l'intégralité du jeu de données. Après cela, l'algorithme incrémente et met à jour les régions denses de chaque partition. Après avoir identifié les régions denses possibles dans chaque partition, l'algorithme utilise une mesure d'interconnectivité pour fusionner les régions de densité afin de former le nombre final de clusters. L'algorithme IDBSCAN a une bonne précision et présente des améliorations significatives sur le temps d'exécution avec un facteur d'accélération de 3,2. Il fonctionne mieux avec des jeux de données volumineux et des dimensions élevées.

L'algorithme IDBSCAN proposé par Bakr et al. [BAK 15] est présenté sous forme de pseudo-code dans l'algorithme 4.3:

Algorithme 4.3 (IDBSCAN)	
Entrées	Ensemble de nouveaux objets P, centroïdes.
Sortie(s)	Créer / mettre à jour des clusters après l'insertion de P
Procédure	M:= Liste des objets pouvant modifier leurs centroïdes; D:= Liste des régions denses mises à jour; Pour chaque P_i dans P faire C:= *le plus proche_centroide*(P_i); P:= *mettre à jour les centroïdes*(p_i,c); Ajouter P_i à M; Fin pour; Pour chaque P_i dans M faire C_n:=r_i; nouveau centroid; C_0:=r_i; ancien centroid; Appliquer *ncDbscanDel* pour retirer r_i de c_0; Appliquer *ncDbscanAdd* pour insérer r_i de c_n; Ajouter des régions denses mises à jour à D; Fin pour; Pour chaque d_i dans D faire Pour chaque d_j dans D et $i \neq j$ faire Si interconnectivité $(d_i, d_j) > \alpha$ fusionner Fusionner (d_i, d_j) Fsi;

	Fin pour; Fin pour; Fin.

4.6. Algorithme AMF-IDBSCAN proposé

Le processus global de l'algorithme proposé AMF-IDBSCAN pour l'amélioration de l'algorithme DBSCAN basée sur l'apprentissage incrémental peut être représenté par la figure 4.3. L'algorithme commence par une phase de pré-clustering en utilisant l'algorithme de canopy clustering. Ensuite, il applique l'algorithme de DBSCAN incrémental inspiré de l'algorithme de Chakraborty et al. [CHA 14]. Puis, il applique la méthode de filtrage médian adaptatif dans la phase de post-clustering pour réduire le nombre de valeurs aberrantes et de bruits en les remplaçant par les médianes choisies. La dernière phase est utilisée pour évaluer les performances des algorithmes de clustering à l'aide de différentes métriques d'évaluation:

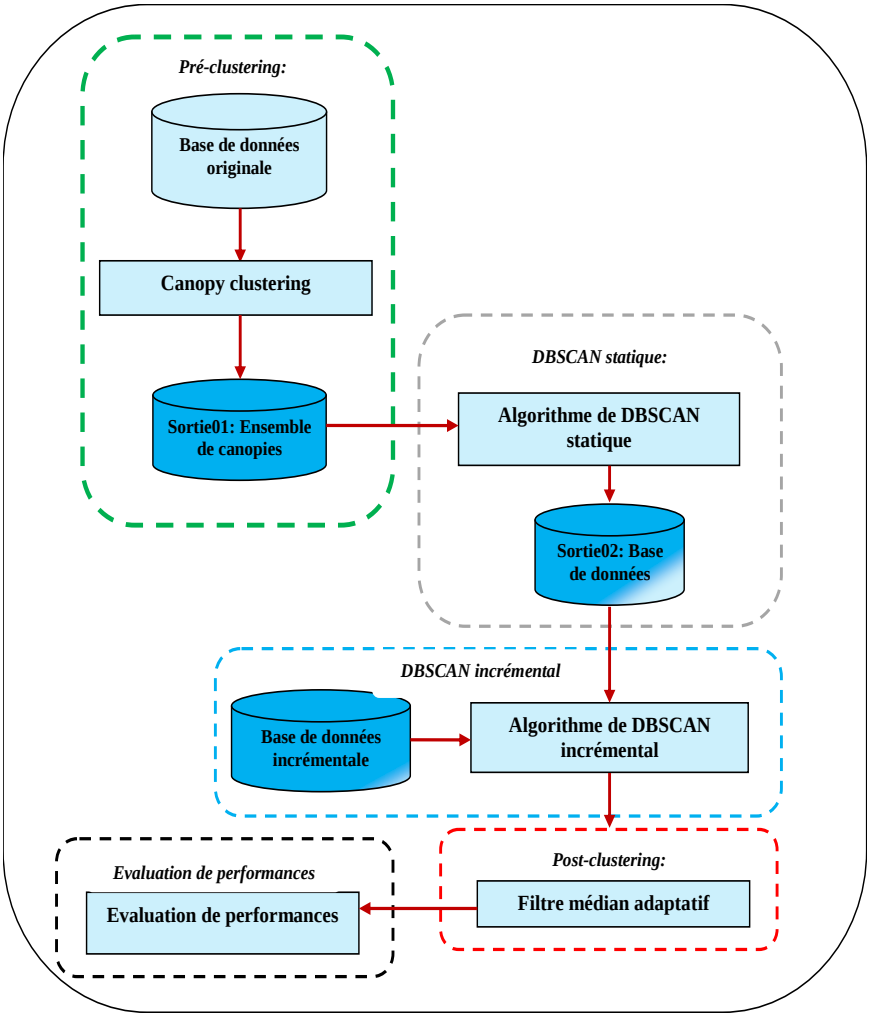

Figure 4.3. Architecture globale de l'algorithme proposé AMF-IDBSCAN

4.6.1. Pré-clustering

Le transfert de la forme depuis l'objet jusqu'à la cible se produit avec un certain bruit. Lors de la première phase, l'algorithme de canopy clustering est appliqué pour réduire considérablement

le nombre de calculs de distance requis pour la classification en commençant par partitionner la base de données en sous-ensembles qui se chevauchent, puis en mesurant uniquement la distance entre les paires de données d'une même canopée [MCC 00] .

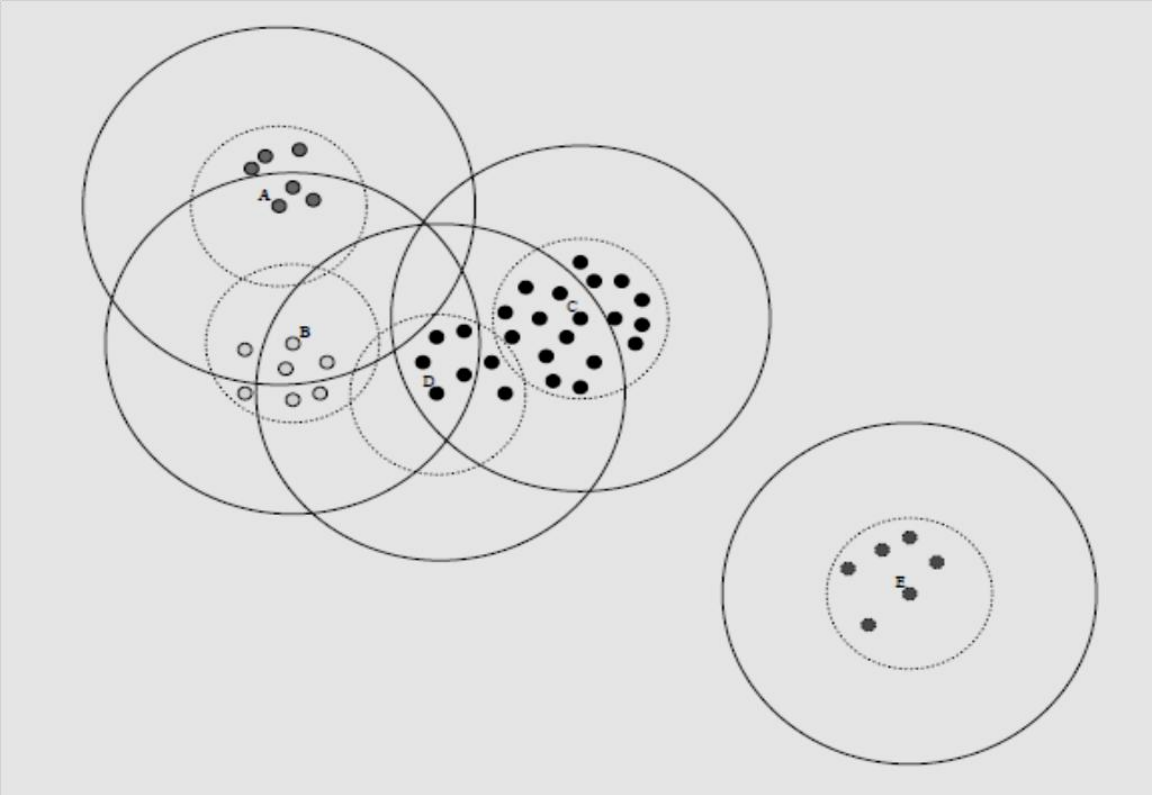

Figure 4.4. Exemple de mise en cluster de canopée [MCC 00]

La figure 4.4 montre quatre groupes de données et des canopées qui les couvrent. C et D sont dans la même canopée, A, B, E sont trois autres canopées.

Dans la figure 4.4, l'objet "A" a été choisi au hasard pour former une canopée composée de tous les objets du cercle extérieur. Les objets à l'intérieur du cercle sont exclus forment des canopées. En pratique, il n'est pas difficile de créer une mesure de distance pour les propriétés des canopées.

Algorithme 4.4 (canopy clustering)	
Entrées	Un ensemble de points S, deux seuils $T1$ et $T2$ $(<T1)$ $C=\varnothing$: ensemble de canopées; $\sum \leftarrow S$, $\sum$ est un ensemble des centres candidats;
Procédure	Tant que $\sum \neq \varnothing$ faire c est un point choisi aléatoirement-> c'est le centre d'une canopée Pour chaque point p dans S faire Si la distance $(p, c) \leq T1$ alors $C=C\cup\{p\}$ ->le canopée C comprend le point p; Si la distance $(p, c) \leq T2$ alors $\sum=\sum-\{p\}$ -> supprimer p des centres candidats;

	Fin-pour; Passer au point suivant Fin-tant que;

Selon l'algorithme (4.4), il existe trois entrées: un ensemble de points de données et deux seuils $T1$ et $T2$ $(<T1)$. Le premier seuil $T1$ influe sur le nombre de points inclus dans les canopées et $T2$ affecte le nombre de canopées créées. Dans un premier temps, l'algorithme initialise les centres candidats en tant que jeu de données en entrée. Ensuite, un point central c est choisi au hasard parmi les candidats. Une canopée autour de ce point central doit être créée dans les étapes suivantes. Une canopée comprend un point de données x si la distance $d(x, c)$ entre x et c est inférieure ou égale au seuil $T1$. En même temps, l'inégalité $d(x, c) \leq T2$ est évaluée et, si elle est vérifiée, le point de données x est supprimé des candidats centraux. Ces étapes de création des canopées se poursuivent jusqu'à ce qu'il n'y ait plus de candidats centraux.

La sortie de pré-clustering est l'entrée au DBSCAN statique...

4.6.2. DBSCAN statique

Lorsqu'on utilise l'algorithme DBSCAN avec l'algorithme de canopy clustering, l'étape de calcul du rayon (eps) peut être réduit et délimite la zone de voisinage d'un point, améliorant ainsi l'efficacité de l'algorithme. Les implémentations de l'algorithme DBSCAN avec canopy clustering impliquent les étapes suivantes (voir la figure 4.5):

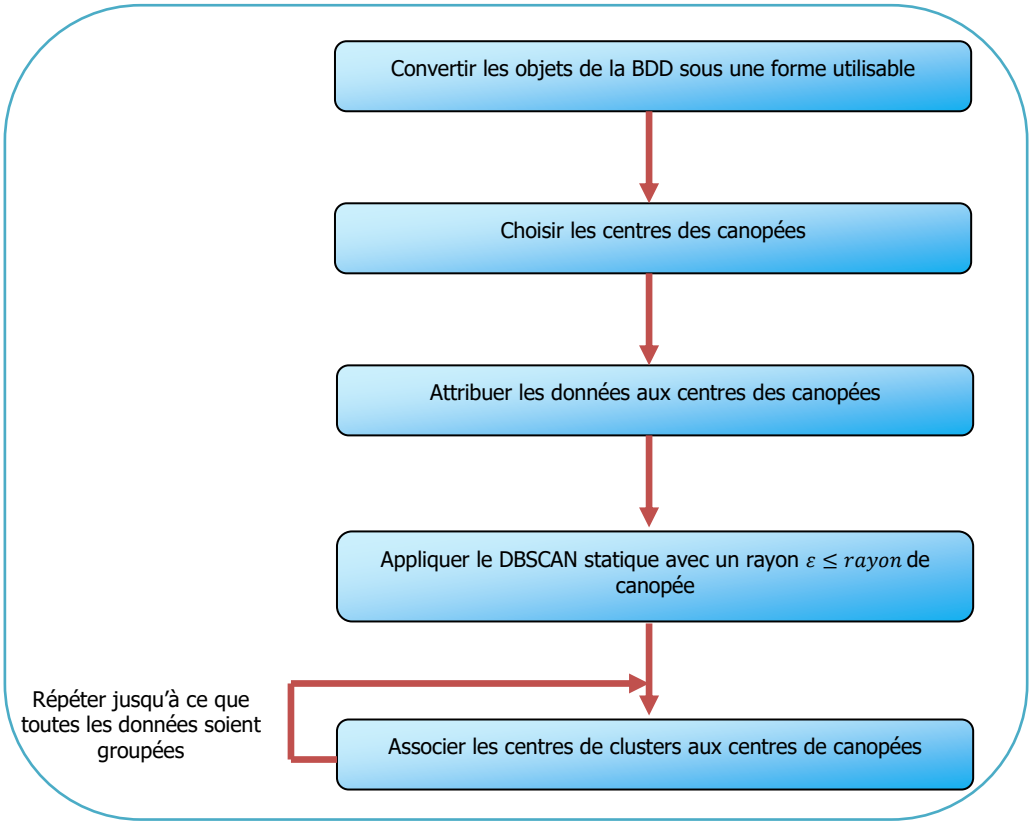

Figure 4.5. L'algorithme DBSCAN avec le canopy clustering

1. Préparation des points de données: Les données d'entrée doivent être transformées en un format approprié et utilisable pour les mesures de distance et de similarité;
2. Définir les centres des canopées (générés par l'étape de pré-clustering);
3. Attribuer des points de données choisis aléatoirement aux centres des canopées: l'étape d'attribution de canopée consiste simplement à affecter des points de données aux centres des canopées générés;
4. Associer les centroides des clusters aux centres des canopées. Les points de données choisis sont maintenant dans des ensembles des clusters.
5. Appliquer l'algorithme DBSCAN sur le reste de l'ensemble de données à partir de chaque centre de canopée avec le rayon ε <= rayon de la canopée et effectuer une itération jusqu'à la mise en cluster. Le calcul de nombre minimal de points voisins (Minpts) est considérablement réduit car nous calculons uniquement la distance entre un centre de clusters et un point de données s'ils partagent la même canopée. Répéter l'itération jusqu'à ce que toutes les données soient en cluster;

Un résumé de l'algorithme DBSCAN [ELG 07]:

1. Choisir aléatoirement un point P;
2. Rassembler dans un groupe c, tous les points de densité-accessible à partir de P en utilisant eps et $MinPts$;
3. si P est un point noyau, alors c forme un cluster;
4. si P est un point limite, alors aucun point n'est atteignable à partir de P;
5. L'algorithme sélectionne un autre point P et reprend en 2 jusqu'à avoir balayer tous les points de P.

4.6.3. DBSCAN incrémental

Dans notre étude, nous avons appliqué l'algorithme de DBSCAN incrémental de Chakraborty et Nagwani [CHA 14].

Cet algorithme fonctionne en deux étapes:

Étape 1. calculer les moyennes entre chaque point noyau ou central de clusters et les nouvelles données ajoutées. Insérer les nouvelles données dans un cluster spécifique en fonction de la distance moyenne minimale. Considérer les données en tant que point aberrant ou point bordure si elles ne peuvent pas être insérées dans des clusters;

Étape 2. former de nouveaux points centraux ou groupes lorsque les points aberrants ou les points de bordure remplissent les critères de $Minpts$ (nombre minimal de points) et de rayon ε.

Parfois, DBSCAN peut être appliqué à une base de données dynamique fréquemment mise à jour par l'insertion ou la suppression de données. Après les insertions et les suppressions dans la base de données, la mise en cluster localisée par DBSCAN doit être mise à jour. Le regroupement incrémental pourrait augmenter les chances d'engendrer des partitions sous-optimales. Dans cette approche, tout d'abord, des clusters basés sur les objets initiaux et un rayon donné (eps) et des $Minpts$ seront formés. Ainsi, l'algorithme obtient enfin des clusters remplissant les conditions et des valeurs aberrantes.

Lorsque de nouvelles données sont insérées dans la base de données existante, les clusters existants doivent être mis à jour à l'aide de DBSCAN statique:

Nous avons utilisé la distance euclidienne car il s'agit actuellement de la métrique le plus utilisée pour les algorithmes de classification établis [GU 17].

L'algorithme correspondant est donc le suivant:

Algorithme 4.5 (DBSCAN incrémental)	
Entrées	D: Ensemble d'individus contenant n objets $\{X_1, X2, X3 \ldots, Xn\}$; n: nombre de données; $Minpts$: Nombre minimal des objets voisins ; $Eps\ ou\ \varepsilon$: rayon de cluster.
Sortie(s)	K: l'ensemble de clusters.
Procédure	C_i (oú $i=1, 2, 3 \ldots$) est la nouvelle donnée. 1. Exécuter l'algoritme **DBSCAN statique** et regrouper la nouvelle donnée C_i en se basant sur le rayon ε et le $Minpts$. Répéter jusqu'à ce que toutes les données soient regroupées. **Pseudo code de l'algorithme DBSCAN incrémental:** **Début:** 2. a> K représente les clusters existants. b>Si un nouvel objet arrive, calculer la distance minimale $mean\ (M)$ entre cet objet et les objets noyaux (centres) des clusters existants; et le grouper dans le cluster de mean minimale. Pour i allant de 1 à n faire Trouver le minimum M d'un cluster K_p dans K qui a une distance entre C_i et M minimal; Si $(dis\ (C_i, M)$ est minimal) et C_i dans le rayon $(C_i <= \varepsilon)$ et le nombre des objets dans K_p est supérieur à $Minpts\ (size(K_p)>=Minpts)$ alors regrouper C_i dans $Kp\ (K_p = K_p\ U\ C_i)$ Sinon Si $dis\ (C_i$ est différent à $M)$ ou $(C_i > \varepsilon)$ ou $(taille(K_p)<Minpts)$ alors C_i est considéré comme donnée aberrante (O_i) Sinon Si le nombre des données aberrantes O_i est supérieur au nombre de $Minpts$ $(Count(O_i)>= Minpts)$ alors O_i forme un nouveau cluster(M_i). C > Répéter l'étape b jusqu'à ce que toutes les données sont regroupées. Fin.

4.6.4. Post-clustering

Nous illustrons les clusters et les points aberrants dans une fenêtre rectangulaire. Nous les avons implémentés dans un hyperplan de n dimensions équivalent aux dimensions des données. Nous appliquons le filtrage adaptatif médian (AMF) pour réduire les données

aberrantes. Nous avons pris l'idée principale de cette méthode et nous l'avons appliquée à notre proposition.

Le filtrage médian adaptatif [CHE 01] a été largement appliqué en tant que méthode avancée par rapport au filtrage médian standard. Le filtre adaptatif fonctionne sur une région rectangulaire W (illustration de l'ensemble des clusters et des valeurs aberrantes générées par l'étape précédente sur un hyperplan). Il modifie la taille de W pendant le filtrage en fonction de certains critères énumérés ci-dessous. La sortie du filtre est une valeur unique qui remplace la valeur de données aberrantes actuelle en $(x, y, \ldots.)$ par le point central de W.

Soit $I_{x, y, \ldots..}$ les données aberrantes sélectionnées selon les dimensions, I_{min} la valeur minimale prise par un bruit et I_{max} la valeur maximale prise par un bruit dans la fenêtre, W la taille de la fenêtre actuelle appliquée, W_{max} la taille de la fenêtre maximale pouvant être atteinte et I_{med} est la médiane de la fenêtre désignée. L'algorithme de cette technique de filtrage se déroule en deux niveaux comme décrit dans [ZHA 07]:

Niveau A:

a) Si $I_{min} < I_{med} < I_{max}$ alors la valeur médiane n'est pas une impulsion (bruit impulsif), l'algorithme passe donc au niveau B pour vérifier si le bruit actuel $I_{x, y}$ est une impulsion
b) Sinon, la taille de la fenêtre est augmentée et le niveau A est répété jusqu'à ce que la valeur médiane ne soit plus un stimulus, de sorte que l'algorithme passe au niveau B; ou la taille maximale de la fenêtre est atteinte, auquel cas la valeur médiane est attribuée en tant que valeur de bruit sélectionnée filtrée.

Niveau B:

a) Si $I_{x, y}$ est une impulsion, elle sera remplacée par la valeur médiane I_{med} sinon le bruit sélectionné filtré reste inchangé.
b) Si les données de bruit sélectionnées sont soit égales à I_{max} ou à I_{min}, la valeur médiane du niveau A est attribuée aux données de bruit sélectionnées filtrées.

4.6.5. Algorithme AMF-IDBSCAN

Pour évaluer les performances, l'étape de pré-clustering est appliquée sur la base de données d'entrées. La sortie de cette étape est les canopées qui ont été stockées dans une autre base de données, puis l'algorithme DBSCAN est appliqué aux résultats de cette base de données. L'algorithme DBSCAN incrémental est appliqué à l'ensemble de données incrémental et la sortie de DBSCAN classique.

L'algorithme proposé est donc le suivant:

Algorithme 4.6 (AMF-IDBSCAN)	
Entrées	D: un ensemble de données qui contient n objets $\{X_1, X_2, X_3 \ldots, X_n\}$; n: nombre des objets; $Minpts$: nombre minimal des objets voisins ; eps: rayon de cluster. C_i: nouvelle donnée; CN: centres des canopées
Sortie(s)	K: ensemble de clusters.

	Une valeur unique: $I_{x,y,....}$ (la valeur de la donnée aberrante) ou I_{med} (la valeur de la médiane)
Procédure	**1. Exécuter Canopy clustering :** 1.1. Placer toutes les données dans une liste; initialiser deux seuils $T1$ et $T2$ $(T1 > T2)$. 1.2. Sélectionner aléatoirement un point comme premier centre de canopée et supprimer cet objet de la liste. 1.3. Prendre un autre point dans la liste et calculer sa distance d par rapport au centre de canopée. Si $d < T2$, le point appartient à cette canopée; si $T2 \leq d \leq T1$, ce point sera marqué avec une étiquette faible; Si la distance d est supérieure à $T1$, le point sera classé comme nouveau centre d'une canopée. Enfin, ce point devrait être éliminé de la liste; 1.4. Exécuter l'étape (1.3) répétitivement jusqu'à ce que la liste soit vide et recalculer les centres de canopées CN. **2. Exécuter l'algorithme statique DBSCAN** et regrouper correctement la nouvelle donnée C_i en fonction des critères: rayon (eps) et Minpts. Répéter jusqu'à ce que toutes les données soient en cluster: 2.1. Définir les centres des canopées CN 2.2. Attribuer les points de données de D choisis aléatoirement aux centres des canopées CN; 2.3. Appliquer l'algorithme DBSCAN sur le reste de l'ensemble de données à partir de chaque centre de canopée avec le $rayon\ \varepsilon <= rayon\ de\ la\ canopée$ et effectuer une itération jusqu'à la mise en cluster. Le calcul de nombre minimal de points voisins $(Minpts)$ est considérablement réduit car nous calculons uniquement la distance entre un centre de clusters et un point de données s'ils partagent la même canopée; 2.4. Répéter (2) jusqu'à ce que toutes les données soient groupées. **3. Exécuter le DBSCAN incrémental:** 3.1. Soit, K représente les clusters déjà existants. 3.2. Lorsqu'un nouvel objet arrive dans la base de données, calculer la distance moyenne minimale (M) entre cet objet et les objets noyaux de clusters existants; et le regroupé. pour $i = 1$ à n do trouver une moyenne de M dans un certain cluster K_p dans K telle que la distance entre C_i et M soit la plus petite; Si $(dis\ (C_i, M)\ est\ minimal)$ et Ci à l'intérieur du $rayon\ (C_i <= eps)$ et que le nombre d'objets dans le K_p est supérieur à $Minpts\ (taille\ (K_p) >= Minpts)$, alors grouper C_i à K_p $(K_p = K_p\ U\ C_i)$ Sinon

	Si *dis (C_i est différent de M)* ou *(C_i> eps)* ou *(taille (Kp) <Minpts)*, alors C_i est considéré comme objet aberrant (O_i) 3.3. Élimination des objets abberants O_i: Le nouvel ensemble de données contient O_i et les clusters les plus proches **3.4. Technique de filtre médian adaptatif:** Pour $i=1$ à m faire {oú m est le nombre des outliers} Illustrer un nouveau rectangle sur un hyperplan: $I_{x,y,...}$est l'outlier sélectionné (O_i) avec les coordoonés $(x,y,...)$; % correspondant aux dimensions des données; I_{min} être la valeur minimale de point aberrant; I_{max} être la valeur minimale de point aberrant dans la fenêtre; W être la taille actuelle de la fenêtre appliquée; % Il contient K clusters et O_i W_{max} être la taille maximale de la fenêtre pouvant être atteinte; I_{med} être la médiane de la fenêtre assignée **Algorithme** Niveau A: $A1 = I_{med} - I_{min}$ $A2 = I_{med} - I_{max}$ Si $A1 > 0$ et $A2 < 0$, passer au niveau B Sinon augmenter la taille de la fenêtre Si la taille de fenêtre $W <= I_{max}$ répéter le niveau A Sinon la sortie est $I_{x,y...}$ Niveau B: $B1 = I_{x,y,..} - I_{min}$ $B2 = I_{x,y,..} - I_{min}$ Si $B1 > 0$ et $B2 < 0$ sortie $I_{x,y,..}$ Sinon la sortie est I_{med}. 3.5. Répéter jusqu'à ce que toutes les données soient groupées. **4. Evaluer les performances.**

4.7. Évaluation d'AMF-IDBSCAN

L'évaluation est une phase essentielle à tout processus d'apprentissage. Elle consiste à vérifier que le modèle construit sur la base d'apprentissage est un classifieur performant et efficace c'est à dire qu'il permet de classer tout individu avec le minimum d'erreurs possibles.

La validation de l'algorithme AMF-IDBSCAN nécessite une validation de sa performance et de son efficacité. Dans cette section, nous nous intéressons particulièrement à l'aspect analytique d'AMF-IDBSCAN. Nous présentons une évaluation de notre algorithme proposé sur cinq bases

de données publiques. Nous avons utilisé des données provenant de l'UCI Machine Learning Repository.

L'UCI Machine Learning Repository contient une collection de bases et de générateurs de données qui sont utilisés par la communauté de l'apprentissage automatique (Machine Learning) pour l'analyse empirique des algorithmes d'apprentissage automatique. L'archive a été créée en 1987 à l'Université de Californie à Irvine (UCI)[5][MEN 15].

Afin de pouvoir évaluer les résultats de l'algorithme proposé, nous l'avons implémenté en utilisant l'outil Weka[6], et le langage de programmation MATLAB[7] .

4.7.1. Choix expérimentaux

4.7.1.1. Paramètres d'entrée

Les paramètres d'entrées ont été déterminés expérimentalement. Ils sont présentés dans les Tableaux 4.2 et 4.3, respectivement, et sont utilisés dans l'algorithme proposé et les algorithmes de comparaison (DBSCAN, IDBSCAN [BAK 15] et DMDBSCAN [ELB 13]:

Paramètres	**Valeurs**
Eps(Rayon de cluster)	0.9
Minpts(Nombre minimal de voisinages)	6

Tableau 4.2. Paramètres de DBSCAN

Paramètres	**Valeurs**
k-dist	3
Eps1(Rayon de cluster)	4.3
Eps2(Rayon de cluster)	4.9
Eps3(Rayon de cluster)	5.1

Tableau 4.3. Paramètres de DMDBSCAN

4.7.1.2. Mesures d'évaluation des performances de classification

Afin de valider correctement le processus de classification, nous utilisons des mesures de performances sur les résultats de la classification. L'efficacité peut se définir selon plusieurs critères. Toutes les mesures utilisées habituellement se basent sur la matrice de confusion dont un exemple est donné dans le Tableau 4.4:

Classe réelle		**Classe obtenue**	
Modèle		Positif	Négatif
	Positif	VP	FP
	Négatif	FN	VN

Tableau 4.4. Matrice de confusion

Dans cette sous-section, nous nous intéressons particulièrement à l'aspect analytique d'AMF-IDBSCAN. Plusieurs mesures d'évaluation, des résultats obtenus pour un apprentissage non supervisé incrémental, existent dans la littérature, en l'occurrence, les indices externes tels que: la précision, le rappel et F-mesure. Ces mesures sont calculées à base de quatre valeurs. Nous allons donner une définition formelle de ces mesures. Nous rappelons les quatre notions suivantes pour une classe donnée (décrites dans la sous section 1.4.4):

[5] http://archive.ics.uci.edu/ml/datasets http://archive.ics.uci.edu/ml/datasets
[6] http://www.cs.waikato.ac.nz/ml/weka
[7] https://fr.mathworks.com/products/matlab.html

Rappel:

VP: est un résultat où le modèle prédit correctement la classe positive

FP: est un résultat où le modèle prédit *incorrectement* la classe *positive*

FN: est un résultat où le modèle prédit *incorrectement* la classe *négative*.

VN: est un résultat où le modèle prédit correctement la classe négative.

$$Précision = \frac{VP}{VP+FP} \text{.....(4.1)}$$

$$Rappel = \frac{VP}{VP+FN} \text{.....(4.2)}$$

$$f - mesure = \frac{2 \times précision \times rappel}{précsion+rappel} \text{.....(4.3)}$$

4.7.2. Expérimentations d'AMF-IDBSCAN avec différentes bases de données

4.7.2.1. Description des bases de données

Dans le but de mieux évaluer l'algorithme AMF-IDBSCAN proposé dans ce chapitre, nous avons utilisé cinq bases de données publiques disponibles dans l'UCI Machine Learning Repository. Le Tableau 4.5 présente une brève description de ces bases:

Nom de la base de données	Nombre des instances	Nombre des attributs	Type des attributs	Types de données
Fisher's Iris	150	4	Réel	Multivariate
Wine	178	13	Entier, réel	Multivariate
Glass Identification	214	10	Réel	Multivariate
Adult	48842	15	Catégorique, entier, réel	Multivariate
Ionosphere	351	34	Entier, réel	Multivariate

Tableau 4.5. Description des bases de données

4.7.2.2. Résultats expérimentaux

Dans cette section, nous présentons les résultats de notre approche AMF-IDBSCAN avec les cinq bases de données décrites précédemment. Les tableaux 4.6 jusqu'à 4.9 présentent les résultats en termes de données, le nombre de clusters obtenus, le taux d'erreur de classification, la f-mesure et le temps de construction de modèle pour les cinq bases de données.

Lorsque nous appliquons l'algorithme DBSCAN à l'ensemble de cinq bases de données avec eps= 0,9 et $MinPts$= 6, nous obtenons les résultats résumés dans le tableau 4.6.

Bases de données	Nombre de clusters	F-mesure	Temps de construction de modèle	Taux d'erreur (%)
Fisher's iris	3	0.264	0.02	35.33
Wine	3	0.175	0.06	26.18
Glass identification	6	0.423	0.04	68.22
Adult	1216	0.475	1638,35	32
Ionosphere	2	0.854	0.23	31.62

Tableau 4.6. Résultats obtenus par DBSCAN

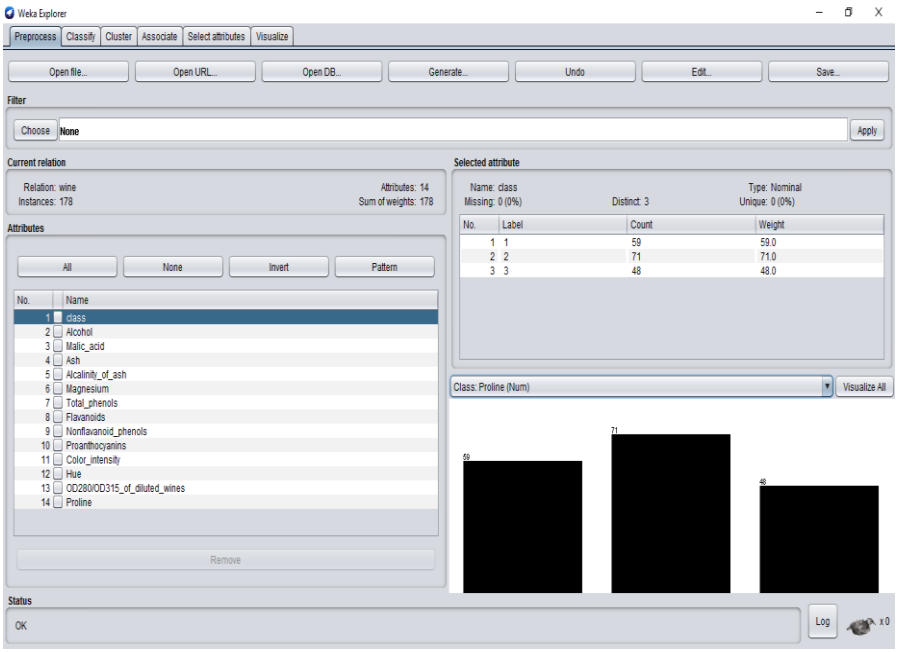

Figure 4.6. Distribution des trois classes de la base de données Wine

Le tableau 4.7 illustre les résultats obtenus lorsque nous appliquons l'algorithme avec *eps*= 0,9 et *MinPts*= 6.

Bases de données	**Nombre de clusters**	**F-mesure**	**Temps de construction de modèle**	**Taux d'erreur (%)**
Fisher's iris	3	0.354	0.03	28.54
Wine	4	0.274	0.05	23.45
Glass identification	8	0.323	0.09	49.52
Adult	1265	0.475	5476.9	27.96
Ionosphere	3	0.639	0.84	29.15

Tableau 4.7. Résultats obtenus par Incremental DBSCAN de Bakr et al. [BAK 15]

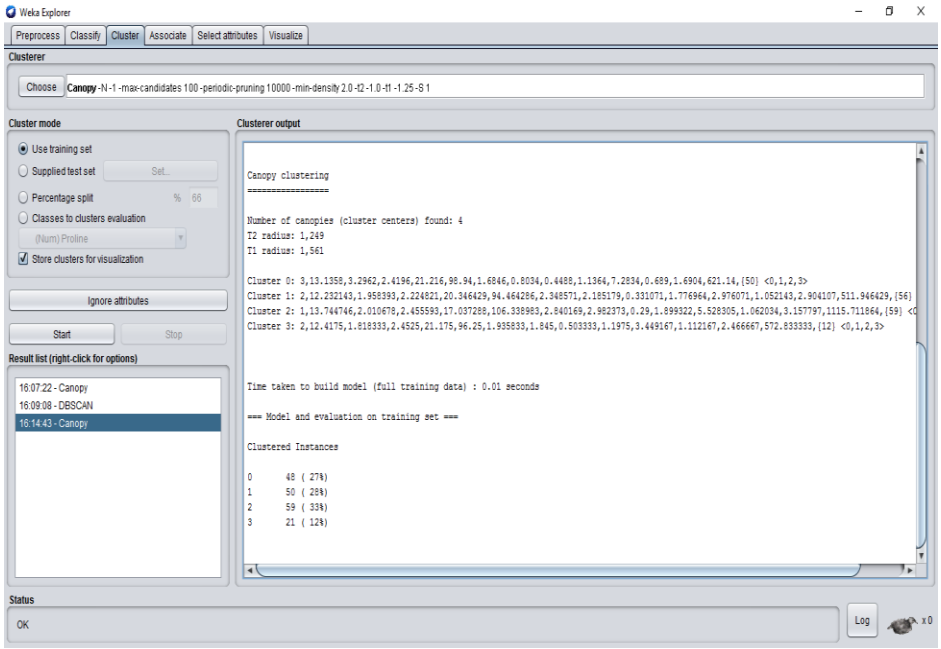

Figure 4.7. Les canopées de la base de données Wine

Bases de données	T1	T2	N° de canopées	Nombre de clusters	F-mesure	Taux d'erreur (%)	Temps de construction de modèle
Fisher's iris	1.092	0.874	3	4	0.798	25.63	0.01
Wine	1,561	1,249	4	4	0.354	18.25	0.02
Glass identification	1,237	0,989	8	6	0.695	35.96	0.04
Adult	2,020	1,616	100	1285	0.495	29.46	0.07
Ionosphere	2,700	2,160	11	5	0.821	27.64	0.04

Tableau 4.8. Résultats obtenus par notre algorithme proposé AMF-IDBSCAN

Dans nos expérimentations, nous avons utilisé pour implémenter le canopy clustering, l'outil Weka (Waikato Environment for Knowledge Analysis) [GOK 16], une application Java open source produite par l'Université de Waikato en Nouvelle-Zélande. Cela fonctionne comme les filtres de prétraitement, la sélection d'attribut, la classification / régression, le clustering, la découverte d'association et la visualisation. L'ensemble d'instances d'apprentissage doit être codé dans un fichier d'entrée avec l'extension .ARFF (Assign Relation File Format) à utiliser par l'outil Weka afin de générer les canopées qui seront utilisées comme entrées dans notre algorithme.

Le tableau 4.8 illustre les résultats obtenus lorsque nous appliquons notre algorithme AMF-IDBSCAN avec eps = 0,9 et $MinPts$ = 6.

Bases de données	Nombre de clusters	F-mesure	Taux d'erreur (%)	Temps de construction de modèle
Fisher's iris	3	0.293	38.46	0.08
Wine	3	0.125	23.15	0.13
Glass identification	6	0.623	58.39	0.24
Adult	1301	0.474	34.66	0.64
Ionosphere	6	0.754	30.04	0.09

Tableau 4.9. Résultats obtenus par DMDBSCAN [ELB 13]

Nous appliquons l'algorithme DMDBSCAN sur l'ensemble de données wine, et en appliquant *k-dist* pour les points les plus proches 3, nous avons 3 valeurs d' *eps* qui sont 4.3, 4.9 et 5.1. Nous obtenons un *f-mesure* = 0,125, un *taux d'erreur* de 23,15% et un *nombre de clusters* = 3. Lors de l'application de DMDBSCAN sur le jeu de données Iris et de l'application *k-dist* pour les points les plus proches à 3, nous avons 2 valeurs de *eps* qui sont 0,39 et 0,45. Nous obtenons un *taux d'erreur* de 38,46%, *f-mesure* = 0,295 et un *nombre de clusters* = 3.

Lorsque nous appliquons DMDBSCAN sur Glass et nous appliquons *k-dist* pour les points les plus proches, nous avons 3 valeurs de *eps* qui sont 0,89, 9,3 et 9,4. Nous obtenons *f-mesure* = 0,623, un *taux d'erreur* de 58,39% et le *nombre de clusters* = 6. Pour l'ensemble de données Adulte, nous obtenons une *f-mesure* = 0,474, le *taux d'erreur* de 34,66%, avec un *nombre de clusters* = 1301. Pour l'ionosphère *f-mesure* = 0,754, un *taux d'erreur* = 30,04% et le *nombre de clusters* = 6.

Le tableau 4.10 compare les résultats obtenus par notre algorithme proposé à ceux des trois algorithmes choisis (voir section 4.5): DBSCAN, IDBSCAN et DMDBSCAN:

Bases de données	DBSCAN			AMF-IDBSCAN			DMDBSCAN			IDBSCAN		
	Nombre de clusters	F-mesure	Taux d'erreur (%)	Nombre de clusters	F-mesure	Taux d'erreur (%)	Nombre de clusters	F-mesure	Taux d'erreur (%)	Nombre de clusters	F-mesure	Taux d'erreur (%)
Fisher's iris	3	0.264	35.33	4	0.798	25.63	3	0.293	38.46	3	0.354	28.54
Wine	3	0.175	26.18	4	0.354	18.25	3	0.125	23.15	4	0.274	23.45
Glass identification	6	0.423	68.22	6	0.695	35.96	6	0.623	58.39	8	0.323	49.52
Adult	1216	0.475	32	1285	0.495	29.46	1301	0.474	34.66	1265	0.475	27.96
Ionosphere	2	0.854	31.62	5	0.821	27.64	6	0.754	30.04	3	0.639	29.15

Tableau 4.10. Comparaison des différents résultats des différents algorithmes

D'après nos expérimentations et comme le montrent les tableaux (4.6 jusqu'à 4.9), nous effectuons les constats suivants:

- En utilisant l'algorithme DBSCAN pour des ensembles de données multi-densités, nous obtenons des résultats de faible qualité avec des durées longues. L'algorithme DBSCAN est un algorithme qui prend beaucoup de temps lorsqu'il s'agit de grands ensembles de données. Cela est dû aux valeurs des paramètres *eps* et *Minpts* qui sont très importantes pour l'algorithme DBSCAN, mais leurs calculs prennent beaucoup de temps. Autrement dit, une meilleure version de l'algorithme DBSCAN serait souhaitable pour traiter ces ensembles de données multi-densités spéciaux.
- DMDBSCAN donne de meilleurs résultats que l'algorithme de classification DBSCAN mais prend plus de temps par rapport aux autres algorithmes. Cela est dû au fait que l'algorithme doit exécuter l'algorithme DBSCAN pour chaque valeur de *Eps*.
- L'algorithme IDBSCAN est efficace en termes de *taux d'erreur* et de *f-mesure* par rapport à l'algorithme DBSCAN. En outre, cela prend plus de temps que DBSCAN, DMDBSCAN et AMF-IDBSCAN. Cela est dû au fait que l'algorithme doit appeler l'algorithme DBSCAN pour effectuer le clustering initial.
- AMF-IDBSCAN produit des meilleurs résultats par rapport à tous les algorithmes avec lesquels nous avons effectué la comparaison. Le tableau 4.9 présente les valeurs de *f-mesure* générées pour tous les jeux de données et tous les algorithmes. Une valeur élevée de *f-mesure* correspond à une meilleure qualité du processus de clustering. Une amélioration significative est constatée sur AMF-IDBSCAN et sur tous les jeux de données, à l'exception du jeu de données Ionosphere. L'augmentation maximale est observée dans les ensembles de données Iris et Glass. L'amélioration de *f-mesure* prouve que notre algorithme proposé est efficace en termes d'élimination du bruit et d'étiquetage des valeurs aberrantes. Outre *f-mesure*, notre méthode proposée nous permet d'obtenir de bons résultats de regroupement dans un temps raisonnable.

Comme la montre la figure (4.8), on peut facilement constater que notre proposition pour la suppression du bruit est bien adaptée aux ensembles de données de grandes dimensions et qu'elle permet d'obtenir de meilleurs résultats par rapport à des travaux de la littérature du domaine.

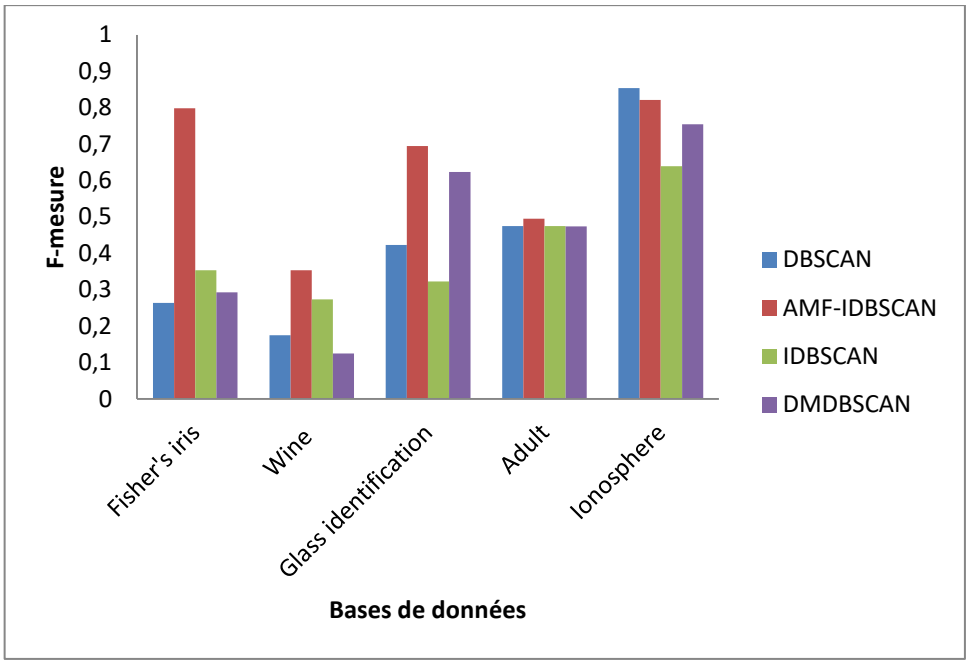

Figure 4.8. Comparaison de f-mesure de tous les algorithmes étudiés

4.8. Conclusion

Dans ce chapitre, nous avons présenté notre algorithme de clustering incrémental proposé AMF-IDBSCAN. Cet algorithme est basé sur l'utilisation de canopy clustering pour pré-grouper les données, le DBSCAN incrémental pour les apprendre et la technique de filtre médian adaptatif pour éliminer les données aberrantes générées par la classification. Les performances d'AMF-IDBSCAN sont comparées avec des extensions de DBSCAN proposées précédemment. Tous les algorithmes sont testés sur cinq bases de données publiques. Nos résultats expérimentaux montrent l'intérêt de l'algorithme proposé car il peut efficacement réduire les objets aberrants et améliorer la précision de la classification.

CONCLUSION ET PERSPECTIVES

Le développement de méthodes d'analyse dynamique de l'information devient une préoccupation centrale dans un grand nombre d'applications dont le but est de traiter de grands volumes de données dynamiques au cours du temps. Les tendances dans le domaine de l'apprentissage automatique se focalisent plus particulièrement sur les problèmes de traitement et d'évolution de grands volumes de données.

La majorité des systèmes de fouille de données ou de reconnaissance de formes sont limités par les données disponibles pour l'apprentissage automatique. En effet, si les données fournies au système ne sont pas représentatives du problème à modéliser, sa réponse ne sera pas fiable car le module d'apprentissage de ce système ne pourra pas fournir un modèle généralisant la réalité. Or, un tel ensemble n'est pas toujours disponible, il faudrait alors que le système soit capable d'utiliser les nouvelles données dont il disposera par la suite pour améliorer ses performances, on parle alors d'apprentissage incrémental…

L'apprentissage incrémental vise à résoudre les problèmes de volumétrie de données, d'évolutivité et d'adaptation dynamique. Cependant, la plupart des systèmes d'apprentissage automatique n'acquièrent, durant la phase d'apprentissage, que des connaissances statiques et limitées. Afin de résoudre ce problème, des chercheurs suggèrent que l'apprentissage soit considéré comme un processus incrémental qui permet de s'adapter automatiquement à l'environnement changeant au sein duquel il évolue.

Ainsi, dans le cadre de cette thèse, nous nous intéressons à l'apprentissage automatique incrémental. Nous avons commencé notre travail en étudiant les concepts et méthodes d'apprentissage classique (statique) supervisé et non supervisé. Nous nous sommes ensuite concentrés sur les concepts et méthodes de l'apprentissage incrémental.

Afin de mieux orienter nos travaux, nous avons effectué une étude bibliographique approfondie de différents travaux existants dans la littérature concernant les algorithmes d'apprentissage incrémental. Dans la plupart des travaux que nous avons étudiés, les auteurs présentent des méthodes d'apprentissage supervisé incrémental basées, entre autres, sur les réseaux de neurones et les SVM d'une part, et les méthodes de clustering incrémental basées sur le partitionnement et la densité d'autre part, faisant intervenir les performances individuelles des classifieurs utilisés.

Après l'analyse des travaux relatifs à l'apprentissage supervisé incrémental, que nous avons étudiés, notamment dans le domaine de la reconnaissance de formes, nous avons constaté que l'utilisation des classifieurs individuels ou simples donne des résultats plus ou moins satisfaisants mais que leur combinaison est considérée comme une perspective prometteuse. En d'autres termes, la tendance récente des recherches s'oriente vers l'hybridation ou la combinaison des méthodes d'apprentissage pour profiter des avantages de deux ou plusieurs méthodes et en atténuer les inconvénients.

Ainsi, nous nous sommes fixés comme but d'étudier l'apport de l'hybridation entre des méthodes d'apprentissage supervisé incrémental dans le domaine de la reconnaissance de formes et d'améliorer des algorithmes de clustering incrémental par l'intégration d'autres

techniques existantes inspirées de l'intelligence artificielle et du traitement de signal telles que le canopy clustering et le filtrage médian adaptatif.

Nous avons proposé la combinaison ISVM-Learn++, pour l'apprentissage supervisé incrémental. C'est la combinaison parallèle de deux classifieurs supervisés incrémentaux : le SVM incrémental (ISVM) et le réseau neuronal incrémental Learn++. Les résultats de nos expérimentations montrent que la combinaison a permis d'obtenir des résultats meilleurs que ceux des classifieurs individuels en termes de réduction d'erreur de classification.

La contribution finale de notre travail consiste en la proposition d'une version améliorée incrémentale, que nous avons appelée AMF-IDBSCAN, de l'algorithme de clustering DBSCAN (Density-Based Spatial Clustering of Applications with Noise). AMF-IDBSCAN est constitué de trois sous-systèmes: le pré-clustering qui utilise l'algorithme canopy clustering pour diminuer la taille de la base de données, l'application de DBSCAN classique pour grouper les objets, le DBSCAN incrémental pour grouper les nouveaux exemples et la technique de filtrage médian adaptatif (AMF) pour éliminer les données aberrantes. Nos résultats expérimentaux montrent l'intérêt de l'algorithme proposé car il peut efficacement réduire les données aberrantes et améliorer la précision de la classification.

Nous pouvons résumer le travail de recherche effectué dans cette thèse à travers les points suivants:

- Etude de concepts, méthodes et algorithmes de l'apprentissage automatique classique supervisé ou non supervisé (Chapitre 1).
- Etude de concepts, méthodes et algorithmes de l'apprentissage automatique incrémental supervisé ou non supervisé (Chapitre 2) [CHE 13].
- Synthèse comparative des méthodes supervisées d'apprentissage incrémental dans le domaine de la reconnaissance de formes (Chapitre 3)[CHE 19a] qui représente le centre d'intérêt de nos travaux antérieurs [CHE 14].
- Proposition de la combinaison ISVM-Learn++ pour l'apprentissage spervisé incrémental (Chapitre 3)[CHE 19b].
- Proposition de AMF-IDBSCAN (Chapitre 4), qui est l'extension d'un algorithme de clustering incrémental (basé DBSCAN) et son évaluation expérimentale comparative [CHE 19c].

Les directions que nous avons prises dans le cadre de notre travail de recherche nous ont permis d'aboutir à des résultats intéressants, mais nous ont aussi ouvert plusieurs voies pouvant être exploitées dans le futur. Certaines des perspectives envisagées sont des extensions relatives aux propositions émises dans cette thèse, telles que:

- L'utilisation d'autres mesures d'évaluation de clustering telles que: les indices relatives comme par exemple l'indice de Dunn's, Davies Bouldin et Silhouette pour évaluer les performances de notre algorithme de clustering incrémental.
- L'utilisation d'autres jeux de données pour la validation de nos propositions. Les données utilisées sont généralement de taille importante, nous avons choisi des bases de données publiques qui sont largement utilisées dans les travaux existants. Nous pouvons aussi appliquer les algorithmes proposés dans des domaines applicatifs précis, tels que la reconnaissance de l'écriture manuscrite, en utilisant des bases de données spécifiques avec de très grandes tailles.

- L’utilisation et, éventuellement, l’hybridation d’autres méthodes d'apprentissage supervisé incrémental et même de clustering incrémental. En effet, suite à notre étude et au constat de la rareté des travaux incluant l'hybridation des algorithmes incrémentaux, nous envisageons, éventuellement, de proposer des hybridations, intégrations ou d’autres combinaisons en utilisant les différentes approches existantes (série, parallèle et hybride) dans nos futurs travaux.

BIBLIOGRAPHIE

[ADE 13] Ade, R. R., Deshmukh, P. R. "Methods for incremental learning: a survey". *International Journal of Data Mining & Knowledge Management Process*, vol. 3, no 4, pp. 119-125, 2013.

[AHA 90] Aha, D. W. "A study of instance-based algorithms for supervised learning tasks: Mathematical, empirical, and psychological evaluations". *PhD Thesis*, California University, Irvine, 1990.

[AHA 91] Aha, D. W., Kibler, D., Albert, M. K. "Instance-based learning algorithms". *Machine learning*, vol. 6, no 1, pp. 37-66, 1991.

[ALM 08] Almaksour, A., Mouchère, H., Anquetil, E. "Apprentissage incrémental et synthèse de données pour la reconnaissance de caractères manuscrits en-ligne". *Actes du 8ème Colloque International Francophone sur l'Ecrit et le Document (CIFED'08),* vol. 1, pp. 55 – 60, 2008.

[ALM 10] Almaksour, A., Anquetil, E, Quiniou, S., Cheriet, M. " Evolving fuzzy classifiers: Application to incremental learning of handwritten gesture recognition systems". In: *Proceedings of the 20th international conference on pattern recognition*. IEEE, pp. 4056-4059, 2010.

[ALM 11] Almaksour, A. "Incremental learning of evolving fuzzy inference systems: application to handwritten gesture recognition". *PhD Thesis*, INSA de Rennes, France, 2011.

[ANK 99] Ankerst, M., Breunig, M. M., Kriegel, H.Peter., Sander, J. "OPTICS: ordering points to identify the clustering structure". *ACM Sigmod record,* pp. 49-60, 1999.

[BAI 15] Bai, X., Ren, P., Zhang, H., Zhou, J. "An incremental structured part model for object recognition". *Neurocomputing*, vol. 154, pp. 189-199, 2015.

[BAK 15] Bakr, A. M., Ghanem, N. M., Ismail, M. A. "Efficient incremental density-based algorithm for clustering large datasets". *Alexandria Engineering Journal*, vol. 54, no 4, pp. 1147-1154, 2015.

[BAL 94] Baluja, S. "Population-based incremental learning. a method for integrating genetic search based function optimization and competitive learning". *Technical Report*, CMU-CS-94-163, Carnegie-Mellon University, Pittsburgh, 1994.

[BAR 96] Barni, M., Cappellini, V., Mecocci, A. "Comments on a possibilistic approach to clustering". *IEEE Transactions on Fuzzy Systems*, vol. 4, no 3, pp. 393-396, 1996.

[BAR 15] Baroudi, R. "Minimisation des désagréments dans les clusters agrégés". *Thèse de doctorat*, Université d'Oran1 Ahmed Benbella, Oran, Algérie, 2015.

[BEL 17] Beloufa, F. "Conception d'un classifeur à base des règles floues". *Thèse de doctorat,* Université Abou bakr Belkaid, Tlemcen, Algérie, 2017.

[BEN 14] Bendjelloul, G. "Utilisation de wordnet dans le clustering des textes d'auteurs par les méthodes des réseaux de kohonen et k-means (étude comparative)". *Mémoire de master*, Université Dr. Tahar Moulay, Saida, Algérie, 2014.

[BEN 16] Benyahia, K. E., Atamnia, M. C. " Vers une approche de classification non supervisée des paramètres météorologiques". *Mémoire de master*, Université Mohamed Chérif Messaadia, Souk Ahras, Algérie, 2016.

[BEN 17] Bendaoud, M. H. "Développement de méthodes d'extraction de contours sur des images à niveaux de gris". *Thèse de doctorat*, Université Mohamed Boudiaf des Sciences et de la Technologie-Mohamed, Oran, Algérie, 2017.

[BEZ 92] Bezdek, J.C., Pal, S.K. *"Fuzzy models for pattern recognition: methods that search for structures in data*". *IEEE Press*, New York, 1992.

[BEZ 98] Bezdek, J.C., Pal, N.R. "Some new indices of cluster validity". *Transactions on Systems, Man, and Cybernetics—Part B*, Vol. 28, pp. 301-315, 1998.

[BIR 07] Birant, D., Kut, A. "ST-DBSCAN: An algorithm for clustering spatial–temporal data". *Data & Knowledge Engineering*, vol. 60, no 1, pp. 208-221, 2007.

[BIS 11] Bisimwa Mugisho, P. "Caractérisation et extraction informatique de la structure d'un tableau par une méthode implémentant un réseau de neurones*". Mémoire de Licence informatique de gestion*, Institut supérieur pédagogique, Kongo, 2011.

[BOL 95] Bolon, P., Chassery, J.Marc, Cocquerez, J.-P., Demigny, D., Graffigne, C., Montanvert, A., Philipp, S., Zéboudj, R., Zerubia, J., Maître, H. "Analyse d'images: filtrage et segmentation". Masson, *Enseignement de la physique*, 2-225-84923-4. hal-00706168, 1995.

[BOU 09] Boulemnadjel, A. "Partitionnement neuronal et validité des classes. Application à la ségmentation d'images". *Mémoire de magister*, université Mentouri, Constantine, 2009.

[BOU 12] Bouillon, M. "Apprentissage incrémental et décrémental: Classification avec un système d'inférence floue évolutif appliquée à la reconnaissance de gestes manuscrits". *Mémoire de Master*, INSA de Rennes, France, 2012.

[BOU 16] Bouillon, M. "Apprentissage actif en-ligne d'un classifieur évolutif, application à la reconnaissance de commandes gestuelles". *Thèse de doctorat*, INSA de Rennes, France, 2016.

[BRA 85] Bradshaw, G. L. "Learning to recognize speech sounds: A theory and model". *PhD Thesis*, Carnegie-Mellon University, Pittsburgh, 1985.

[BRY 17] Brynjolfsson, E., Mitchell, T. "What can machine learning do? Workforce implications". *Science*, vol. 358, no 6370, pp. 1530-1534, 2017.

[CAR 83] Carbonell, J. G., Michalski, R. S., Mitchell, T. M. "An overview of machine learning". *Machine learning*. Morgan Kaufmann, pp. 3-23, 1983.

[CAR 91] Carpenter, G. A., Grossberg, S., Rosen, D. B. "Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system". *Neural networks*, vol. 4, no 6, pp. 759-771, 1991.

[CAR 92] Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H., Rosen, D. B. "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps". *IEEE Transactions on neural networks*, vol. 3, no 5, pp. 698-713, 1992.

[CAU 01] Cauwenberghs, G., Poggio, T. "Incremental and decremental support vector machine learning". *Advances in neural information processing systems*, pp. 409-415, 2001.

[CHA 11a] Chakraborty, S., Nagwani, N. K. "Analysis and study of incremental k-means clustering algorithm". In: *Proceedings of the International Conference on High Performance Architecture and Grid Computing*. Springer, Berlin, Heidelberg, pp. 338-341, 2011.

[CHA 11b] Chakraborty, S., Nagwani, N. K. "Analysis and study of Incremental DBSCAN clustering algorithm". *International Journal of Enterprise Computing and Business Systems*, vol. 1, no. 2, 2011.

[CHE 01] Chen, T., Wu, H. R. "Adaptive impulse detection using center-weighted median filters". *IEEE signal processing letters*, vol. 8, no 1, pp. 1-3, 2001.

[CHE 07] Chen, Z., Huang L., Murphey, Y. L. "Incremental learning for text document classification". In: *Proceedings of the International Joint Conference on Neural Networks*. IEEE, pp. 2592-2597, 2007.

[CHE 12] Chergui, L. "*Combinaison de classifieurs pour la reconnaissance de mots arabes manuscrits"*. *Thèse de doctorat*, université Mentouri, Constantine, Algérie, 2012

[CHE 13] Chefrour, A., Souici-Meslati, L. "Un panorama de méthodes d'apprentissage incrémental". *13e Conférence Francophone sur l'Extraction et la Gestion des Connaissances EGC'2013, Atelier CIDN Classification Incrémentale et Détection de Nouveauté*, Toulouse, France, 2013.

[CHE 14] Chefrour, A., Souici-Meslati, L. "SVM incrémental pour la classification de chiffres manuscrits", SFC'2014, 21ème Rencontre de la Société Francophone de Classification, Rabat, Maroc, Septembre 2014.

[CHE 19a] Chefrour, A. "Incremental supervised learning: algorithms and applications in pattern recognition". *Evolutionary Intelligence*, vol. 12, no 2, pp. 97-112, 2019.

[CHE 19b] Chefrour, A., Souici-Meslati, L., Difi, I., Bekkouche, N. "A novel incremental learning algorithm based on incremental support vector machine and incremental neural network Learn++". *Revue d'Intelligence Artificielle*. Vol. 33, no. 3, pp. 181-188, 2019.

[CHE 19c] Chefrour, A. Souici-Meslati, L. "AMF-IDBSCAN: Incremental density based clustering algorithm using adaptive median filtering technique". *Informatica, an International journal of Computing and Informatics.* Vol. 43, no. 4, pp. 495-506, 2019.

[CHE 19d] Chen, C., Min, W., Li, X., Jiang, S. "Hybrid incremental learning of new data and new classes for hand-held object recognition". *Journal of Visual Communication and Image Representation*, vol. 58, pp. 138-148, 2019.

[COR 95] Cortes, C., Vapnik, V. "Support-vector networks". *Machine learning*, vol. 20, no 3, pp. 273-297, 1995.

[COS 89] Coster, M., Chermant, L. L. "Précis d'analyse d'images". *Presses du CNRS*, 1989.

[DAL 09] Dalton, L., Ballarin, V., Brun, M. "Clustering algorithms: on learning, validation, performance, and applications to genomics". *Current genomics*, vol. 10, no 6, pp. 430-445, 2009.

[DÉN 02] Déniz, O., Castrillon, M., Lorenzo, J., Hernández, M. "An incremental learning algorithm for face recognition". In: *Proceedings of the International Workshop on Biometric Authentication*. Springer, Berlin, Heidelberg, pp. 1-9, 2002.

[DID 71] Diday, E. "Une nouvelle méthode en classification automatique et reconnaissance des formes la méthode des nuées dynamiques". *Revue de statistique appliquée*, vol. 19, no 2, pp. 19-33, 1971.

[DIE 03] Diehl, C. P., Cauwenberghs, Gt. "SVM incremental learning, adaptation and optimization". In: *Proceedings of the International Joint Conference on Neural Networks.* IEEE, pp. 2685-2690, 2003.

[DIM 05] DI Mauro, N., Esposito, F., Ferilli, S., Basile, T. M. "Avoiding order effects in incremental learning". *Congress of the Italian Association for Artificial Intelligence*. Springer, Berlin, Heidelberg, pp. 110-121, 2005.

[DUD 12] Duda, R. O., Hart, P. E., Stork, D. G. "Pattern classification". John Wiley & Sons, 2012.

[DUN 73] Dunn, J. C. "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters". *Journal of Cybernetics*, vol. 3, pp. 32-57, 1973.

[ELB 12] Elbatta, M. "An improvement for DBSCAN algorithm for best results in varied densities". *Master thesis,* Islamic University, Gaza, 2012.

[ELB 13] Elbatta, M., Ashour, W. "A dynamic method for discovering density varied clusters". *International Journal of Signal Processing, Image Processing and Pattern Recognition* , vol. 6, no 1, pp. 123-134, 2013.

[ELG 07] Elghazel, H. "Classification et Prévision des données hétérogenes: Application aux trajectoires et séjours hospitaliers". *Thèse de doctorat*, université Lyon 1, France, 2007.

[ENE 05] Enembreck, F., Brathès, J.P. "ELA—a new approach for learning agents". *Autonomous Agents and Multi-Agent Systems*, vol. 10, no 3, pp. 215-248, 2005.

[ERD 05a] Erdem, Z., Polikar, R., Guergen, F., Yumusak, N."Ensemble of SVMs for incremental learning". In : *Proceedings of the International Workshop on Multiple Classifier Systems*. Springer, Berlin, Heidelberg, pp. 246-256, 2005.

[ERD 05b] Erdem, Z., Polikar, R., Guergen, F., Yumusak, N."Reducing the effect of out-voting problem in ensemble based incremental support vector machines". In : *Proceedings of International Conference on Artificial Neural Networks*. Springer, Berlin, Heidelberg, pp. 607-612, 2005.

[EST 96] Ester, Ma., Kriegel, H. P, Sander, J., Xu, X. "A density-based algorithm for discovering clusters in large spatial databases with noise". In : *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 226-231, 1996.

[FAH 90] Fahlman, S.E., Lebiere, C. "The cascade-correlation learning architecture". *Advances in neural information processing systems*. pp. 524-532, 1990.

[FRE 99] French, R.M. "Catastrophic forgetting in connectionist networks". *Trends in cognitive sciences*, vol. 3, no 4, pp. 128-135, 1999.

[FRI 95] Fritzke, B. "A growing neural gas network learns topologies". *Advances in neural information processing systems*, pp. 625-632, 1995.

[FUN 02] Fung, G., Mangasarian, O. L. "Incremental support vector machine classification". In: *Proceedings of the SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics, pp. 247-260, 2002.

[GAS 05] Gasmi, I., Merouani, H. F, Souici-Meslati, L. "Combinaison de classifieurs", In: *Proceeding of the 3rd International Conference : Sciences of Electronic, Technologies of Information and Telecommunication (SETIT)*, Sousse-Tunisia, 2005.

[GEO 99] George, K., Han, E. H., Kumar, V. "CHAMELEON: a hierarchical clustering algorithm using dynamic modeling". *IEEE Computer*, vol. 27, no 3, pp. 329-341, 1999.

[GHA 07] Ghassabeh, Y. A., Moghaddam, H. A. "A face recognition system using neural networks with incremental learning ability". In: *Proceedings of the International Symposium on Computational Intelligence in Robotics and Automation*. IEEE, pp. 291-296, 2007.

[GIR 07] Girard, A. "Exploration d'un algorithme génétique et d'un arbre de décision à des fins de catégorisation". *Thèse de doctorat*, Université du Québec à Trois-Rivières, Québec, Canada, 2007.

[GON 93] Gonzalez, A. J., Dankel, D.D. "The engineering of knowledge-based systems: theory and practice". *Prentice-Hall*, Inc., 1993.

[GOK 16] Gokilam, G. G., Shanthi, K. "Performance Analysis of Various Data mining Classification Algorithms on Diabetes Heart dataset". *International Journal of Pharmaceutical Research and Medicinal Plants*, vol. 1, no 1, 2016.

[GOW 86] Gower, J. C., Legendre, P. "Metric and Euclidean properties of dissimilarity coefficients". *Journal of classification*, vol. 3, no 1, pp. 5-48, 1986.

[GRA 08] Granger, E., Connolly, J. F., Sabourin, R. "A comparison of fuzzy ARTMAP and Gaussian ARTMAP neural networks for incremental learning". In : *Proceedings of the IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pp. 3305-3312, 2008.

[GRE 97] Greene, J. R. "A role for simple, robust 'Black-Box'optimisers in the evolution of engineering systems and artefacts". In: *Proceedings of the International Conference on genetic algorithms in engineering systems : innovations and application* , pp. 427- 432, 1997.

[GRO 76] Grossberg, S. "Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors". *Biological Cybernetics*, vol. 23, no 3, pp. 121-134, 1976.

[GRO 82] Grossberg, S. "How does a brain build a cognitive code?. In: Studies of mind and brain". *Springer*, pp. 1-52, 1982.

[GU 17] Gu, X., Angelov, P. P., Kangin, D., Principe, J. C. "A new type of distance metric and its use for clustering". *Evolving Systems*, vol. 8, no 3, pp. 167-177, 2017.

[GUE 15] Guermoudi, M., Fekih, M. "Fusion des classifieurs supervisés: Application sur la classification pixellaire des images microscopiques ". *Mémoire de master*, université Abou Bakr Belkaid, Tlemcen, Algérie, 2015.

[GUE 19] Guebaili, N. "Combinaison des classifieurs non supervisés". *Mémoire de Master*, Université Mohamed Chérif Messaadia, Souk Ahras, 2019.

[GUH 00] Guha, S., Rastogi, R., Shim, K. "Rock: A robust clustering algorithm for categorical attributes". *Information systems*, vol. 25, no 5, pp. 345-366, 2000.

[GUP 13] Gupta, N., Ujjwal, R. L." An efficient incremental clustering algorithm". *World of Computer Science & Information Technology Journal*, vol. 3, no 5, pp. 23-29, 2013.

[HAC 17] Hacene, G. B., Gripon, V., Farrugia, N., Arzel, M., Jezequel, M. "Incremental Learning with Pre-Trained Convolutional Neural Networks and Binary Associative Memories". In: *Proceedings of the International Workshop on Signal Processing Systems*, pp. 1 - 4, 2017.

[HAN 01] Han, J., Kamber, M., Tung, A. K. "Spatial clustering methods in data mining". *Geographic data mining and knowledge discovery*, pp. 188-217, 2001.

[HAN 11] Han, J., Pei, J., Kamber, M. "Data mining: concepts and techniques". *Elsevier*, 2011.

[HAN 16] Han, S., Meng, Z., Khan, A. S., Tong, Y. "Incremental boosting convolutional neural network for facial action unit recognition". *Advances in neural information processing systems*, pp. 109-117, 2016.

[HE 11] He, Y., Tan, H., Luo, W., Mao, H., Ma, D., Feng, S., Fan, J. "MR-DBSCAN: an efficient parallel density-based clustering algorithm using mapreduce". In : *Proceedings of the 17th International Conference on Parallel and Distributed Systems*. IEEE. pp. 473-480, 2011.

[HEB 99] Hebert, J. F., Marizeau, M., Ghazzali, N. "An hybrid architecture for active and incremental learning: The self-organizing perceptron (SOP) network". In : *Proceedings of the International Joint Conference on Neural Networks*. EEE, pp. 1646-1651, 1999.

[HÉB 00] Hébert, J.F. "Architecture neuronale hybride pour l'apprentissage incrémental des connaissances: application à la reconnaissance d'écriture cursive". *Thèse de doctorat*, université Laval, Québec, Canada, 2000.

[HEB 08] Hebboul, A. "Approches incrémentales pour l'apprentissage en reconnaissance de formes". *Mémoire de magister*, université Mentouri, Constantine, Algérie, 2008.

[HUA 07] Huang, C., Ai, H., Yamashita, T., Lao, S., Kawade, M. "Incremental learning of boosted face detector". In: *Proceedings of the 11th International Conference on Computer Vision. IEEE*, pp. 1-8, 2007.

[HUB 85] Hubert, L., Arabie, P. "Comparing partitions". *Journal of classification.,* vol. 2, no 1, pp. 193-218, 1985.

[HUL 94] Hull, D. "Improving text retrieval for the routing problem using latent semantic indexing". In: *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pp. 282-291, 1994.

[HUL 07] Hulley, G., Marwala, T. "Evolving classifiers: Methods for incremental learning". *Computing Research Repository, abs/0709.3965*, 2007.

[HWA 95] Hwang, H., Haddad, R. A. "Adaptive median filters: new algorithms and results". *IEEE Transactions on image processing*, vol. 4, no 4, pp. 499-502, 1995.

[IAN 00] Ianakiev, K., Govindaraju, V. "Architecture for classifier combination using entropy measures". In: *Proceedings of the International Workshop on Multiple Classifier Systems*. Springer, Berlin, Heidelberg, pp. 340-350, 2000.

[JAB 88] Jabbour, K., Riveros, J. F. V., Landsbergen, D., Meyer, W. "ALFA: Automated load forecasting assistant". *IEEE Transactions on Power Systems*, vol. 3, no 3, pp. 908-914, 1988.

[JAI 88] Jain, A. K., Dubes, R. C. "Algorithms for clustering data". *Prentice-Hall,Inc*, ISBN 0-13-022278-X, 1988.

[KAR 04] Karray, F., Karray, F. O., De Silva, C. W. "Soft computing and intelligent systems design: theory, tools, and applications". *Pearson Education*, 2004.

[KAU 09] Kaufman, L., Rousseeuw, P. J. "Finding groups in data: an introduction to cluster analysis". *John Wiley & Sons*, 2009.

[KAW 13] Kawewong, A., Pimup, R., Hasegawa, O. "Incremental learning framework for indoor scene recognition". In: *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, pp. 496-502, Bellevue, 2013.

[KOU 11] Koudri, M. "Modèle de mélange Gaussien. Application sur image cytologique". *Mémoire de master*, université AbouBakr Belkaid, Tlemcen, Algérie,2011.

[LAD 19] Laadjailia, A. "Analyse et reconnaissance des activités humaines des séquences vidéo". *Thèse de doctorat*, Université Badji Mokhtar, Annaba, Algérie, 2019.

[LAM 17] Lamraoui, D., Slimani, H. "Filtrage des images par différentes approches ". *Mémoire de master*, université M'hamed Bougara, Boumerdes, Algérie, 2017.

[LAN 95] Langley, P. "Order effects in incremental learning. Learning in humans and machines: Towards an interdisciplinary learning science". vol. 136, pp. 137, Pergamon, 1995.

[LAW 17] Lawal, I. A., Abdulkarim, S. A. "Adaptive SVM for data stream classification". *South African Computer Journal*, vol. 29, no 1, pp. 27-42, 2017.

[LEC 95] Lecourtier, Y., Ennaji, A., Stpcker, E., Gilles. F. "Yprel networks, classification and incremental learning". *Traitement du signal*, vol. 12, no 6, pp. 597-608, 1995.

[LEN 83] Lenat, D. B. "Eurisko: a program that learns new heuristics and domain concepts: the nature of heuristics III: program design and results". *Artificial intelligence*, vol. 21, no 1-2, pp. 61-98, 1983.

[LIU 06] Liu, B. "A fast density-based clustering algorithm for large databases. In : *Proceedings of the International Conference on Machine Learning and Cybernetics*. IEEE, pp. 996-1000, 2006.

[LIU 07] Liu, P., ZHOU, D., WU, N." VDBSCAN: varied density based spatial clustering of applications with noise". In : *Proceedings of the International Conference on service systems and service management*. IEEE, pp. 1-4, 2007.

[LIU 15] Liu, Z., Li, X., Luo, P., Loy, C. C., Tang, X. "Semantic image segmentation via deep parsing network". In: *Proceedings of the IEEE international Conference on computer vision*. pp. 1377-1385, 2015.

[LIU 17] Liu, X., Yang, Q., He, L. "A novel DBSCAN with entropy and probability for mixed data". *Cluster Computing*, vol. 20, no 2, pp. 1313-1323, 2017.

[LOO 06] Loosli, G. "Méthodes à noyaux pour la détection de contexte". *Thèse de doctorat,* INSA de Rouen, France, 2006.

[LU 14] Lu, Y., Boukharouba, K., Boonært, J., Fleury, A., Lecoeuche, S. "Application of an incremental SVM algorithm for on-line human recognition from video surveillance using texture and color features". *Neurocomputing*, vol. 126, pp. 132-140, 2014.

[LUO 07] Luo, J., Pronobis, A., Caputo, B., Jensfelt, P. "Incremental learning for place recognition in dynamic environments". In: *Proceedings of the International Conference on Intelligent Robots and Systems*. IEEE, pp. 721-728, 2007.

[MAI 16] Mai, S.T., Assent, I., Storgaard, M. "AnyDBC: an efficient anytime density-based clustering algorithm for very large complex datasets". *In : Proceedings of the 22nd ACM SIGKDD International Conference on knowledge discovery and data mining*. pp. 1025-1034, 2016.

[MAL 03] Mali, K., Mitra, S. "Clustering and its validation in a symbolic framework". *Pattern Recognition Letters*, vol. 24, no 14, pp. 2367-2376, 2003.

[MAM 13] Mamine, S. "Apprentissage hybride adaptatif pour les systèmes évolutifs". *Mémoire de magister*, université Badji Mokhtar, Annaba, algérie, 2013.

[MAN 02] Mandziuk, J., Shastri, L. "Incremental class learning approach and its application to handwritten digit recognition". *Information Sciences*, vol. 141, no 3-4, pp. 193-217, 2002.

[MAR 13] Marref, N. "*Apprentissage Incrémental &Machines à Vecteurs Supports"*. *Mémoire de magister*. Université Mustafa Ben Boulaid, Batna, Algérie, 2013.

[MCC 00] McCallum, A., Nigam, K., Ungar, L. H. "Efficient clustering of high-dimensional data sets with application to reference matching". In: *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge discovery and data mining*, pp. 169-178, 2000.

[MCD 89] McDonald, C. "Machine learning: a survey of current techniques". *Artificial Intelligence Review*, vol. 3, no 4, pp. 243-280, 1989.

[MEN 15] Menghour, K. "Approches Bio-inspirées pour la Sélection d'Attributs". *Thèse de doctorat,* Université Badji Mokhtar, Annaba, Algérie, 2015.

[MER 12] Merabet, R. "Une approche automate cellulaire pour L'apprentissage d'un agent". *Thèse de doctorat*, Université Mohamed khider, Biskra, Algérie, 2012.

[MIC 86] Michalski, R.S., Winston, P.H. "Variable precision logic". *Artificial intelligence*, vol. 29, no 2, pp. 121-146, 1986.

[MIS 06] Misina, S. "Incremental learning for e-mail classification". *Computational Intelligence, Theory and Applications*. Springer, Berlin, Heidelberg, pp. 545-553, 2006.

[MIT 83] Mitchell, T.M., Utgoff, P. E., Banerji, R. "Learning by experimentation: Acquiring and refining problem-solving heuristics". *Machine learning*. Springer, Berlin, Heidelberg. pp. 163-190, 1983.

[MIT 90] Mitchell, T. M., Buchanan, B., Dejong, G., Dietterich, T., Rosenbloom, P., Waibel, A. "Machine learning". *Annual review of computer science*, vol. 4, no 1, pp. 417-433, 1990.

[MIT 06] Mitchell, T. M., "The discipline of machine learning". *CMU-ML-06-108*, vol. 9, Carnegie Mellon university, Pittsburgh, 2006.

[MOH 12] Mohemmed, A., LU, G., Kasabov, N. "Evaluating SPAN incremental learning for handwritten digit recognition". In: *Proceedings of the International Conference on Neural Information Processing*. Springer, Berlin, Heidelberg, pp. 670-677, 2012.

[MOL 14] Molina, J. F. G., Zheng, L., Sertdemir, M., Dinter, D. J., Schönberg, S., Rädle, M "Incremental learning with SVM for multimodal classification of prostatic adenocarcinoma". *PloS one*, vol. 9, no 4, pp. e93600, 2014.

[MUH 04] Muhlbaier, M. D., Topalis, A., Polikar, R. "Learn++. MT: A new approach to incremental learning". In: *Proceedings of the International Workshop on Multiple Classifier Systems*. Springer, Berlin, Heidelberg, pp. 52-61, 2004.

[MUH 09] Muhlbaier, M.D., Topalis, A., Polikae, R. "Learn. NC: Combining Ensemble of Classifiers With Dynamically Weighted Consult-and-Vote for Efficient Incremental Learning of New Classes". *IEEE transactions on neural networks*, vol. 20, no 1, pp. 152-168, 2009.

[MUR 08] Murphey, Y.L., Chen, Z. H., Feldkamp, L. A. "An incremental neural learning framework and its application to vehicle diagnostics". *Applied Intelligence*, vol. 28, no 1, pp. 29-49, 2008.

[MUS 01] Mustière, S. "Apprentissage supervisé pour la généralisation cartographique". *Thèse de doctorat*, université de Paris 6, Paris, France, 2001.

[NEM 10] Nemouchi, S. Farah, N. "Reconnaissance de l'Ecriture Arabe par Systèmes Flous". *Mémoire de magister,* université Badji Mokhtar, Annaba, Algérie, 2010.

[NG 02] Ng, R. T., Han, J. "CLARANS: A method for clustering objects for spatial data mining". *IEEE Transactions on Knowledge & Data Engineering*, no 5, pp. 1003-1016, 2002.

[NGO 15] Ngo Ho, A. K. "Méthodes de classifications dynamiques et incrémentales: application à la numérisation cognitive d'images de documents". *Thèse de doctorat*, université de Tours, France, 2015.

[OZA 05] Ozawa, S., Toh, S. L., Abe, S., Pang, S., Kasabov, N "Incremental learning of feature space and classifier for face recognition". *Neural Networks*, vol. 18, no 5-6, pp. 575-584, 2005.

[OZA 08] Ozawa, S., Pang, S., Kasabov, N. "Incremental learning of chunk data for online pattern classification systems". *IEEE Transactions on Neural Networks*, vol. 19, no 6, pp. 1061-1074, 2008.

[PEN 04] Peng, L. "Adaptive median filtering". ~~In:~~ *Seminar Report, Machine Vision*, vol. 140, pp. 1-13, 2004.

[PIN 03] Pinker, S. "The blank slate: The modern denial of human nature". *Penguin*, 2003.

[POL 01] Polikar, R., Upda, L., Upda, S. S., Honavar, V. "Learn++: An incremental learning algorithm for supervised neural networks". *IEEE transactions on systems, man, and cybernetics, part C (applications and reviews)*, vol. 31, no 4, pp. 497-508, 2001.

[PRU 06] Prudent, Y. "Système d'apprentissage incrémental et hybride". *Thèse de doctorat*, université de Rouen, France, 2006.

[QUI 86] Quinlan, J. R. "Induction of decision trees". *Machine learning*, vol. 1, no 1, pp. 81-106, 1986.

[QUI 93] Quinlan, J. R. "C4.5: programs for machine learning", *Morgan Kaufmann San Mateo*, California, 1993.

[QUI 14] Quinlan, J. R. "C4.5: programs for machine learning". *Elsevier*, 2014.

[RAL 01] Ralaivola, L., D'alché-buc, F. "Incremental support vector machine learning: A local approach". In: *Proceedings of the International Conference on Artificial Neural Networks*. Springer, Berlin, Heidelberg, pp. 322-330, 2001.

[RAM 10] Ram, A., Jalal, S., Jalal, A. S., Kumar, M. "DVBSCAN: A density based algorithm for discovering density varied clusters in large spatial databases". *International Journal of Computer Applications*, pp. 0975-8887, 2010.

[RED 09] Reddy, K. K., Liu, J., Shah, M. "Incremental action recognition using feature-tree". In: *Proceedings of* the *12th international conference on computer vision*. IEEE, pp. 1010-1017, 2009.

[RIS 87] Rissland, E. L., Stillings, N. "Artificial Intelligence: knowledge representation". In: *Chapter 4 in Cognitive science: an introduction, second printing*. Massachusetts Institute of Technology, 1987.

[ROM 14] Romano, S., Bailey, J., Nguyen, V., Verspoor, K. "Standardized mutual information for clustering comparisons: one step further in adjustment for chance". *In : Proceedings of the International Conference on Machine Learning*. pp. 1143-1151, 2014.

[ROU 87] Rousseeuw, P. J. "A graphical aid to the interpretation and validation of cluster analysis". *Journal of computational and applied mathematics*, vol. 20, pp. 53-65, 1987.

[RUM 85] Rumelhart, D. E., Hinton, G. E., Williams, R.J. "Learning internal representations by error propagation". California University San Diego La Jolla Inst for Cognitive Science, 1985.

[RUP 01] Ruping, S.. "Incremental learning with support vector machines". In: *Proceedings of the International Conference on Data Mining*. IEEE, pp. 641-642, 2001.

[RUS 16] Russell, S. J., Norvig, P. "Artificial intelligence: a modern approach". *Malaysia; Pearson Education Limited*, 2016.

[SAL 10] Salperwyck, C., Lemaire, V., De Bois, D. U. D. P. "Classification incrémentale supervisée: un panel introductif*". Revue des Nouvel les Technologies de l'Information, Numéro spécial sur l'apprentissage et la fouille de données* . pp. 121-148, 2010.

[SAL 12] Salperwyck, C. "Apprentissage incrémental en ligne sur flux de données". *Thèse de doctorat,* Université Charles de Gaulle-Lille III, France, 2012.

[SAM 59] Samuel, A. L. "Some Studies in Machine Learning Using the Game of Checkers". *IBM journal of Research and Development 3*, vol. 44, no 1.2, pp. 206-226, 1959.

[SAR 17] Sarwar, S. S., Ankit, A., Roy, K. "Incremental learning in deep convolutional neural networks using partial network sharing". *CoRR, vol. abs/1712.02719*, 2017.

[SAS 07] Sasaki, Y. "The truth of the F-measure". *Teach Tutor mater*, vol. 1, no 5, pp. 1-5, 2007.

[SCH 86] Schlimmer, J. C., Fisher, D. H. "A case study of incremental concept induction". In : *Proceedings of the 5th National Conference on Artificial Inteligence*, pp. 496-501, 1986.

[SHA 90] Shavlik, J. W., Dietterich, T., Dietterich, T. G. "Readings in machine learning". *Morgan Kaufmann*, 1990.

[SHI 18] Shivarudrappa, S. "Real-time physiological identification using incremental learning and semi-supervised learning". *master thesis*, Michigan Dearborn university, 2018.

[SIL 17] Silhavy, R., Senkerik, R., Oplatkova, Z. K., Prokopova, Z., Silhavy, P. "Artificial Intelligence Trends in Intelligent Systems".In: *Proceedings of the 6th Computer Science On-line Conference* ,Springer, vol. 1, 2017.

[SIM 83] Simon H. A. "Why should machines learn?". *Machine learning*, pp. 25-37, 1983.

[SIM 13] Simon, P. "Too big to ignore: the business case for big data". *John Wiley & Sons*, 2013.

[SOW 10] Sowjanya, A. M., Shashi, M. "Cluster feature-based incremental clustering approach (CFICA) for numerical data". *International Journal of Computer Science and Network Security*, vol. 10, no 9, pp. 73-79, 2010.

[SUT 13] Suthar, N., Indr, P., Vinit, P. "A Technical Survey on DBSCAN Clustering Algorithm". *International Journal of Scientific and Engineering Research*, vol. 4, no 5, 2013.

[SWA 17] Kiruthika, V. S., & Thiagarasu, V. " FI-DBSCAN: Frequent Itemset Ultrametric Trees with Density Based Spatial Clustering Of Applications with Noise Using Mapreduce in Big Data". *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 5, pp. 56-64, 2017.

[SYE 99] Syed, N. A., Huan, S., Kah, L., Sung, K. "Incremental learning with support vector machines". In: *Proceedings of the Workshop on Support Vector Machines at the International Joint Conference on Articial Intelligence*, Stockholm, Sweden. 1999.

[TJH 09] Tjhi, W. C., Chen, L. "Dual fuzzy-possibilistic coclustering for categorization of documents". *IEEE Transactions on Fuzzy Systems*, vol. 17, no 3, pp. 532-543, 2009.

[TOH 03] Toh, S. L., Ozawa, S. "A face recognition system using neural networks with incremental learning ability". In: *Proceedings of the 8th Australian and New Zealand Conference on Intelligent Information Systems*, pp. 389-394, 2003.

[UNC 06] Uncu, O., Gruver, W. A., Kotak, D. B., Sabaz, D., Alibhai, Z., Ng, C. "Gridbscan: Grid density-based spatial clustering of applications with noise". In : *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pp. 2976-2981, 2006.

[UTG 89] Utgoff, P. E. "Incremental induction of decision trees". *Machine learning*, vol. 4, no 2, pp. 161-186, 1989.

[VAN 04] Van Rijsbergen, C. J. "The geometry of information retrieval". *Cambridge University Press*.

[VAP 13] Vapnik, V. "The nature of statistical learning theory". *Springer science & business media*, 2013.

[VAP 15] Vapnik, V. N., Chervonenkis, A. Y. "On the uniform convergence of relative frequencies of events to their probabilities". *In: Vovk V., Papadopoulos H., Gammerman A. (eds) Measures of Complexity,* Springer, pp. 11-30, 2015.

[VIJ 10] Vijaykumar, V. R., Jothibasu, P. "Decision based adaptive median filter to remove blotches, scratches, streaks, stripes and impulse noise in images". In : *Proceedings of the IEEE International Conference on Image Processing*, pp. 117-120, 2010.

[VIN 09] Vinh, N.Xuan., Epps, J., Bailey, J. "Information theoretic measures for clusterings comparison: is a correction for chance necessary*?".* In : *Proceedings of the 26th annual International Conference on machine learning,* pp. 1073-1080, 2009.

[VIS 06] Viswanath, P., Pinkesh, R. "I-dbscan: A fast hybrid density based clustering method". In : *Proceedings of the 18th International Conference on Pattern Recognition*. IEEE, pp. 912-915, 2006.

[WAN 97] Wang, W., Yang, J., Muntz, R. "STING: A statistical information grid approach to spatial data mining*".* In: *Proceedings of the 23rd International Conference on Very Large Data Bases*, pp. 186-195, 1997.

[WAN 16] Wang, S., Liu, Y., Shen, B. "MDBSCAN: Multi-level density based spatial clustering of applications with noise". In: *Proceedings of the The 11th International Knowledge Management in Organizations Conference on The changing face of Knowledge Management Impacting Society*. pp. 21, 2016.

[WEI 10] Weimer, M. "Machine Teaching--A Machine Learning Approach to Technology Enhanced Learning". *PhD Thesis*, Technische Universität, Brunswick, 2010.

[WEN 07] Wen, Y. M., Lu, B. L. "Incremental learning of support vector machines by classifier combining". In: *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, Berlin, Heidelberg, pp. 904-911, 2007.

[WEN 16] Wen, H., Xie, W., Pei, J., Guan, L. "An incremental learning algorithm for the hybrid RBF-BP network classifier". *EURASIP Journal on Advances in Signal Processing*, vol. 2016, no 1, pp. 57, 2016.

[WIT 16] Witten, I. H., Frank, E., Hall, M. A. "Data Mining: Practical machine learning tools and techniques". *Morgan Kaufmann*, 2016.

[WU 07] Wu, H., Wang, Y., Huai, X. "AttributeNets: an incremental learning method for interpretable classification". In: *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, Berlin, Heidelberg, pp. 940-947, 2007.

[YAD 16] Yada, P., Sharma, P. "An Efficient Incremental Density based Clustering Algorithm Fused with Noise Removal and Outlier Labelling Technique". *Indian Journal of Science and Technology*, vol. 9, pp. 1-7, 2016.

[YEN 99] Yen, O., Meesad, P. "Pattern classification by an incremental learning fuzzy neural network". In: *Proceedings of the International Joint Conference on Neural Networks*. IEEE, pp. 3230-3235, 1999.

[YEN 01] Yen, G. G., Meesad, P.. "An effective neuro-fuzzy paradigm for machinery condition health monitoring". *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 31, no 4, pp. 523-536, 2001.

[ZHA 96] Zhang, T., Ramakrishinan, R., Livny, M. "BIRCH: an efficient data clustering method for very large databases". *ACM Sigmod Record,* pp. 103-114, 1996.

[ZHA 06] Zhao, H., Yuen, P.C., Kwok, J. T. "A novel incremental principal component analysis and its application for face recognition". *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 36, no 4, pp. 873-886, 2006.

[ZHA 07] Zhao, Y., LI, D., LI, Z.i. "Performance enhancement and analysis of an adaptive median filter". In: *Proceedings of the Second International Conference on Communications and Networking in China*. IEEE, pp. 651-653, 2007.

[ZIA 16] Zia, M. S. "Incremental Learning Based Classification for Facial Expression Recognition". *PhD Thesis*. National University of Computer and Emerging Sciences, Islamabad, Pakistan, 2016.

[ZOU 09] Zou, L., Zhang, T., Cao, Z. "An incremental learning algorithm based on Support Vector Machine for pattern recognition". *Pattern Recognition and Computer Vision*. International Society for Optics and Photonics, pp. 74961H, 2009.

[ZRI 16] Zribi, M., Boujelbene, Y. "The neural networks with an incremental learning algorithm approach for mass classification in breast cancer". *Biomedical Data Mining*, vol. 5, no 1, 2016.

[ZUR 92] Zurada, J. M. "Introduction to artificial neural systems". *West publishing company*, 1992.

A PROPOS DE L'AUTEUR

BIOGRAPHIE DE L'AUTEUR

Aida Chefrour est née à Souk Ahras (Algérie) le 24 Octobre 1984, où elle a poursuivi ses études jusqu'à l'obtention du Baccalauréat en 2001, option sciences de la nature. Elle a rejoint l'université Badji Mokhtar, Annaba, pour suivre des études d'ingénieur d'état en informatique, spécialité intelligence artificielle, jusqu'à l'année 2006. Ayant réussi au concours de Magister option Technologies de l'Information et de la Communication (TIC) qui s'est déroulé en 2007, elle accède aux études post graduées dans la même université. Après quatre années de recherche, elle obtient le diplôme de Magister en informatique. Suite à cela, elle devient enseignante à l'Ecole Normale Supérieure d'enseignement Technologique (ENSET) à Skikda depuis 2011. Elle a effectué une mutation à l'université de Mohamed Chérif Messaadia de Souk Ahras en 2014. Elle a enseigné en qualité de chargé de cours de plusieurs modules tels que: Interaction Homme Machine (IHM), Technologies de l'Information et de la Communication (TIC) et Terminologie scientifique et expression écrite et orale. Elle a participé, en qualité de membre, à deux projets de recherche CNEPRU intitulés, "Classification et apprentissage dans les systèmes complexes", agréés de 2012 jusqu'à 2018. Elle a également participé à plusieurs conférences internationales. Depuis 2013, elle est membre du Laboratoire d'Ingénierie des Systèmes Complexes (LISCO) de l'université d'Annaba. Elle est actuellement membre d'un projet de recherche PRFU intitulé, "Apprentissage artificiel et BIo-inspiré dans les Systèmes COmplexes (ABISCO)", agréé pour 2019-2022.

CONTRIBUTIONS SCIENTIFIQUES

1. Articles parus dans des journaux internationaux

1 Chefrour, A., (2019). "Incremental supervised learning: algorithms and applications in pattern recognition". *Evolutionary Intelligence*. Vol. 12, no. 2, pp. 97-112.

Informations sur le journal Evolutionary Intelligence

ISSN: 1864-5909 (Print) 1864-5917 (Online)
Fréquence de publication: 4 numéros par an (depuis 2008)
Publié par: Springer Verlag
Site Web: https://link.springer.com/journal/12065
Indexé par : SCOPUS, **Emerging Sources Citation Index**, INSPEC, Zentralblatt Math, Google Scholar, DBLP, EBSCO Discovery Service, EI Compendex, Expanded Academic, Gale, Gale Academic OneFile, OCLC WorldCat Discovery Service, ProQuest Advanced Technologies & Aerospace Database, ProQuest SciTech Premium Collection, ProQuest Technology Collection, ProQuest-ExLibris Primo, ProQuest-ExLibris Summon.
HIndex: 19
(https://www.scimagojr.com/journalsearch.php?q=14500154734&tip=sid&clean=0)

2 Chefrour, A. Souici-Meslati, L., (2019). "AMF-IDBSCAN: Incremental density based clustering algorithm using adaptive median filtering technique". *Informatica, an International journal of Computing and Informatics.* Vol. 43, no. 4, pp. 495-506.

Informations sur le journal Informatica (Ljubljana)

ISSN: 0350-5596
Fréquence de publication: 4 numéros par an (depuis 1977)
Publié par: Slovensko Drustvo Informatika
Site Web: http://www.informatica.si/index.php/informatica/index
Indexé par : SCOPUS, Emerging Sources Citation Index, ACM Digital Library, Citeseer, COBISS, Compendex, Computer & Information, Systems Abstracts, Computer Database, Computer Science Index, dLib.si, DBLP Computer Science Bibliography, Directory of Open Access Journals, Google Scholar, InfoTrac OneFile, Inspec, Linguistic and Language Behaviour Abstracts, Mathematical Reviews, MatSciNet, MatSci on SilverPlatter and Current Mathematical Publications.
HIndex: 31
(https://www.scimagojr.com/journalsearch.php?q=25507&tip=sid&clean=0)

3 Chefrour, A., Souici-Meslati, L., Difi, I., Bekkouche, N., (2019). "A novel incremental learning algorithm based on incremental support vector machine and incremental neural network Learn++". *Revue d'Intelligence Artificielle*. Vol. 33, no. 3, pp. 181-188.

Informations sur le journal Revue d'Intelligence Artificielle (RIA)

ISSN: 0992-499X (print); 1958-5748 (online)
Fréquence de publication: 6 numéros par an (depuis 2010)
Publié par: Lavoisier
Site Web: http://www.iieta.org/Journals/RIA
Indexé par: SCOPUS, SCImago (SJR), Ei Compendex, EBSCOhost, Google Scholar, CrossRef, Microsoft Academic
HIndex: 13
(https://www.scimagojr.com/journalsearch.php?q=24847&tip=sid&clean=0)

2. Communications avec comité de lecture international

1. Aida CHEFROUR, Siham AMROUCH and Labiba SOUICI-MESLATI : Reconnaissance de noms de wilayas Algériennes en arabe par les arbres de décision. ICIST'2011, the first International Conference on Information Systems and Technologies, Tébessa, Algeria, 2011.
2. Aida CHEFROUR, Siham AMROUCH and Labiba SOUICI-MESLATI: Recognition of handwritten Arabic words with structural features and decision tree learning. CISC'2011, the Second International Conference on Complex Systems, Jijel, Algeria, 2011.
3. Aida CHEFROUR, Siham AMROUCH and Labiba SOUICI-MESLATI: Decision trees for handwritten Arabic words recognition, ACIT'2011, the International Arab Conference on Information Technology, Saudi Arabia, 2011.
4. Aida CHEFROUR, Labiba SOUICI-MESLATI : Un panorama de méthodes d'apprentissage incrémental, EGC'2013, 13ème Conférence Francophone sur l'Extraction et la Gestion des Connaissances, Atelier CIDN (Classification Incrémentale et Détection de Nouveauté), Toulouse, France, 2013.

5. Aida CHEFROUR, Labiba SOUICI-MESLATI : SVM incrémental pour la classification de chiffres manuscrits, SFC'2014, 21ème Rencontre de la Société Francophone de Classification, Rabat, Maroc, Septembre 2014.

5. Communications avec comité de lecture national

1. Aida CHEFROUR, Labiba SOUICI-MESLATI : Arbres de décision pour la reconnaissance des mots arabes manuscrits, JSPMC, Premières Journées Scientifiques de Physique, Mathématiques et Chimie, Skikda, Algérie, 2012.

Résumé

Le travail présenté dans ce document se situe dans le cadre général de l'apprentissage automatique et, plus précisément, celui de l'apprentissage incrémental qui propose une alternative d'évolutivité et d'adaptation dynamique pour intégrer de nouvelles connaissances, de nouvelles données ou pour restructurer des problèmes déjà partiellement appris. L'objectif principal de l'apprentissage incrémental est de disposer d'un système capable d'apprendre de nouvelles informations (nouvelles données, nouvelles classes...) sans pour autant oublier les connaissances déjà acquises. Suite à notre étude des concepts et méthodes de l'apprentissage incrémental ainsi que des travaux relatifs à l'apprentissage incrémental supervisé, notamment dans le domaine de la reconnaissance de formes, nous avons constaté le manque de propositions d'hybridation et de combinaison de ces méthodes. Dans ce cadre, nous comparons et combinons deux algorithmes d'apprentissage supervisé le SVM incrémental et le réseau neuronal incrémental Learn++. Le système proposé, que nous avons nommé ISVM-Learn++, a montré des bonnes capacités d'apprentissage incrémental sur plusieurs jeux de données et a permis d'obtenir des résultats satisfaisants. Après une étude approfondie des méthodes d'apprentissage non supervisé ou clustering, notamment celles proposées dans un cadre incrémental, nous nous sommes intéressés particulièrement au DBSCAN (Density-Based Spatial Clustering of Applications with Noise). Nous proposons AMF-IDBSCAN, une version améliorée de cet algorithme qui construit de manière incrémentale les clusters de formes et de tailles différentes dans de grands ensembles de données et élimine la présence de bruit et de valeurs aberrantes. L'algorithme AMF-IDBSCAN proposé utilise le canopy clustering pour pré-regrouper les ensembles de données afin de réduire le volume de données, applique un DBSCAN incrémental pour regrouper les exemples d'apprentissage et la technique AMF (Adaptive Median Filtering) pour réduire les données aberrantes ou bruits en les remplaçant par les médianes choisies. Les résultats expérimentaux obtenus, sur plusieurs bases de données, montrent que notre algorithme donne de bons résultats par rapport au DBSCAN et à certaines de ses extensions proposées dans la littérature.

Mots clés : Apprentissage incrémental; Apprentissage supervisé ; Clustering; DBSCAN; AMF-IDBSCAN; ISVM-Learn++.

Abstract

Our research work is in the general framework of machine learning and, more specifically, incremental learning which offers an alternative scalability and dynamic adaptation to integrate new knowledge, new data or to restructure already partially learned problems. Therefore, the main objective of incremental learning is to have a system capable of learning new information (new data, new classes ...) without forgetting the knowledge already acquired. After a detailed study of the concepts and methods of incremental learning as well as work relating to supervised incremental learning, in particular in the field of pattern recognition, we noted the lack of proposals for hybridization and combination of these methods. In this framework, we compare and combine two supervised learning algorithms for the incremental SVM and the incremental neural network Learn ++. The proposed system, which we named ISVM-Learn ++, has shown good incremental learning capabilities on several datasets and has achieved satisfactory results. After study on the unsupervised learning methods or clustering, those proposed in an incremental framework, we were particularly interested in DBSCAN (Density-Based Spatial Clustering of Applications with Noise). We offer AMF-IDBSCAN, an improved version of this algorithm that incrementally builds clusters of different shapes and sizes in large data sets and eliminates the presence of noise and outliers. The proposed AMF-IDBSCAN algorithm uses canopy clustering to pre-cluster the data sets to reduce the volume of data, applies an incremental DBSCAN to group the learning examples and the AMF (Adaptive Median Filtering) technique to reduce the outliers or noises by replacing them with the chosen medians. The experimental results obtained, on several databases, show that our algorithm gives good results compared to DBSCAN and to some of its extensions proposed in the literature.

Keywords: Incremental learning; supervised learning; clustering; dbscan; AMF-IDBSCAN; ISVM-Learn ++.

الملـخص

العمل المقدم في هاته الوثيقة يندرج في الإطار العام للتعلم الآلي وبالتحديد التعلم الإضافي الذي يوفر قابلية بديلة للتكيف والتكيف الديناميكي لدمج المعرفة أو البيانات الجديدة أو لإعادة هيكلة المشكلات التي سبق تعلمها جزئيًا. لذلك فإن الهدف الرئيسي للتعلم التدريجي هو أن يكون لديك نظام قادر على تعلم معلومات جديدة (بيانات جديدة، فصول جديدة....) دون أن ينسى المعرفة المكتسبة بالفعل. بعد دراسة مفصلة لمفاهيم وأساليب التعلم التدريجي وكذلك العمل المتعلق به الخاضع للإشراف وخاصة في مجال التعرف على الأنماط لاحظنا عدم وجود مقترحات لتهجين هذه الأساليب ومزيجها. في هذا الإطار نقوم بمقارنة ودمج خوارزميتي تعليمي خاضعتين للإشراف التدريجي SVM والشبكة العصبية الإضافية..++ Learn. أظهر النظام المقترح الذي أطلقنا عليه اسم++ ISVM-Learn، قدرات تعليمية تدريجية جيدة وحقق نتائج مرضية على العديد من مجموعات البيانات .بعد دراسة أساليب التعلم غير الخاضعة للرقابة أو ألمجموعات تلك المقترحة في إطار تدريجي انصب اهتمامنا بشكل خاص بالتجميع المكاني القائم على الكثافة للتطبيقات ذات الضوضاء. اقترحنا AMF-IDBSCAN، كنسخة محسنة من هذه الخوارزمية التي تصنع بشكل تدريجي مجموعات من الأشكال والأحجام المختلفة انطلاقا من مجموعات البيانات الكبيرة، وتزيل وجود الضوضاء والقيم المتطرفة. تستخدم خوارزمية AMF-IDBSCAN المقترحة مجموعات المظلات لتجميع مجموعات البيانات مسبقًا لتقليل حجمها وتطبق ايضا خوارزمية DBSCAN التزايدي لتجميع أمثلة التعلم وتقنيةAMF للتصفية الوسيطة التكيفية للحد من القيم المتطرفة أو الضوضاء من خلال استبدالها بالقيم الوسيطة المختارة. أجريت التجارب على عدة قواعد بيانات عامة، وأظهرت نتائج الخوارزمية المقترحة والتي تم تقييمها اعطت نتائج جيدة مقارنةً بـ DBSCAN وبعض امتداداتها المقترحة.

الكلمات المفتاحية : التعلم الآلي، التعلم التدريجي، DBSCAN ،++ISVM-Learn،AMF-IDBSCAN ، التعرف على الشكل.