ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN GÉNIE DE LA PRODUCTION AUTOMATISÉE
M.Ing.

PAR
GUILLAUME TREMBLAY

OPTIMISATION D'ENSEMBLES DE CLASSIFIEURS NON PARAMÉTRIQUES AVEC APPRENTISSAGE PAR REPRÉSENTATION PARTIELLE DE L'INFORMATION

MONTRÉAL, LE 16 DÉCEMBRE 2004

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

M. Robert Sabourin, directeur de mémoire
Département de génie de la production automatisée, École de technologie supérieure

M. Rachid Aissaoui, président du jury
Département de génie de la production automatisée, École de technologie supérieure

M. Éric Granger, examinateur
Département de génie de la production automatisée, École de technologie supérieure

IL A FAIT L'OBJET D'UNE PRÉSENTATION DEVANT JURY ET PUBLIC

LE 6 DÉCEMBRE 2004

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

OPTIMISATION D'ENSEMBLES DE CLASSIFIEURS NON PARAMÉTRIQUES AVEC APPRENTISSAGE PAR REPRÉSENTATION PARTIELLE DE L'INFORMATION

Guillaume Tremblay

SOMMAIRE

L'un des défis de la reconnaissance de formes (RF) est de concevoir des systèmes à la fois simples (peu de paramètres, faible coût de calcul) et performants (haut taux de reconnaissance). Il est démontré que les ensembles de classifieurs (**EoC**) peuvent permettre d'obtenir de meilleures performances qu'un classifieur unique, d'où la recherche d'un compromis entre simplicité et performance. L'utilisation de classifieurs non paramétriques de type *k*-**NN** ayant une représentation partielle de l'information favorise toutefois la simplicité d'un système de RF. Dans le présent travail, nous avons utilisé un tel ensemble de *k*-**NN** pour vérifier s'il était possible de concevoir des **EoC** par sélection de classifieurs pour améliorer la simplicité tout en augmentant la performance du système.

L'utilisation d'un algorithme d'optimisation pouvant explorer de grands espaces mal définis est nécessaire pour atteindre cet objectif. Afin d'aider la recherche, différentes mesures de «diversité» sont proposées dans la littérature. Nous avons tenté d'optimiser un **EoC** à l'aide de différentes méthodes de recherche et avons testé l'effet de la maximisation conjointe de la performance avec un échantillon des mesures de diversité les plus populaires. Toutes les expériences ont été répétées 30 fois de façon à pouvoir comparer, à l'aide de tests statistiques, les différentes approches évaluées.

Nous avons découvert que la maximisation conjointe de la simplicité et de la performance était la meilleure façon de créer un ensemble optimisant ces deux objectifs. Par contre, pour générer des ensembles ayant une performance maximale, l'utilisation d'un algorithme de recherche à un seul objectif est préférable. Contrairement à nos attentes, il n'a pas été possible de démontrer un avantage significatif à l'utilisation d'une mesure de diversité comme critère d'optimisation.

À notre connaissance, c'était la première fois qu'était étudiée de manière exhaustive la façon de faire de la sélection de classifieurs de type *k*-**NN** basés sur le paradigme des sous-espaces aléatoires. L'application systématique de tests statistiques pour valider les résultats des stratégies de sélection de classifieurs a été rendue possible grâce à l'utilisation d'une grappe d'ordinateurs et à la création de base de données de votes précalculés. Cette validation statistique est rarement mise en oeuvre dans le domaine.

OPTIMIZING ENSEMBLES OF NONPARAMETRIC CLASSIFIERS TRAINED WITH PARTIAL INFORMATION REPRESENTATION

Guillaume Tremblay

ABSTRACT

A challenge in pattern recognition (PR) is to design systems that are simple (few parameters, low calculation costs) and achieve high-performance (high recognition rate). It has been shown that ensembles of classifiers (**EoC**) can achieve better performances than those of a single classifier thus being the source of a trade off between simplicity and performance. However, nonparametric classifiers such as *k*-**NN** trained with partial information representation help keeping PR systems simple. In this work, an ensemble of such *k*-**NN** was used to assess whether **EoC** could be designed by classifiers selection, improving simplicity while increasing performance of the system.

A search algorithm which can explore large and poorly understood spaces is necessary to achieve this goal. Moreover, several "diversity" measures have been proposed in recent literature to help this kind of search. As a consequence, we tried to optimize an **EoC** with several search methods and we tested the effect of joint maximization of both performance and diversity measures selected among the most popular ones. All the experiments were repeated 30 times in order to compare the various approaches with statistical tests.

We discovered that joint maximization of simplicity and performance was the best way of optimizing both of these objectives. On the other hand, mono-objective search algorithms are more suitable to generate **EoC** having the highest recognition rates. Contrary to our expectations, it has not been possible to show that the use of a diversity measure, as an optimization criterion, gives any significant advantage.

To the best of our knowledge, it is the first time that random subspaces based *k*-**NN** selection has been studied in such an exhaustive way. Furthermore, systematic application of statistical tests, made possible by cluster computing, had been seldomly implemented in experimental protocols of the PR community.

REMERCIEMENTS

La réalisation de ce mémoire a été rendue possible grâce à la participation financière de RDDC-Valcartier ainsi qu'aux bourses du *Programme d'incitation pour diplômés de l'ÉTS à poursuivre des études de 2e et 3e cycles à l'ÉTS.*

Ce projet a été réalisé dans le cadre du contrat W7701-024425/001/QCA entre DRDC-Valcartier et l'École de technologie supérieure (ÉTS) sous l'autorité scientifique de Patrick Maupin.

Je tiens à remercier mon directeur de mémoire, Robert Sabourin pour la confiance qu'il m'a accordé en m'accueillant au sein de son équipe ainsi que pour la grande disponibilité dont il a fait preuve tout au long du projet.

Beaucoup de gratitude va aussi à Patrick Maupin, qui a constamment veillé à augmenter la qualité de ce travail jusque dans ses menus détails. Sans son investissement et sa patience, le présent mémoire serait plus difficile à lire tout en étant scientifiquement moins complet.

Je veux remercier sincèrement Luiz Soares Oliveira et Eulanda Miranda dos Santos pour leur aide. Les expériences réalisées au cours de ce travail leurs doivent beaucoup.

Merci à tous les membres du LIVIA, avec qui j'ai partagé des moments enrichissants tant sur le plan humain que sur le plan intellectuel.

Finalement, merci Sophie pour toutes tes relectures, corrections, attentions mais aussi et surtout de m'avoir accompagné et encouragé tout au long de ce parcours. Merci également à notre fils, Victor, de m'avoir si patiemment attendu.

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

LISTE DES FIGURES

Page

LISTE DES ALGORITHMES

LISTE DES ABRÉVIATIONS ET SIGLES

ACP	Analyse en composantes principales
$\widetilde{A}$	Ambiguïté (mesure de diversité)
AG	Algorithme génétique
AGMO	Algorithme génétique multi-objectifs
CA	Caractéristiques absentes
$\widetilde{E}$	Entropie selon la définition classique en théorie de l'information (mesure de diversité)
E_0	Erreur à zéro rejet
E_b	Erreur bayesienne
EI	Écart interquartile
$\widetilde{E_K}$	Entropie selon la définition de Kuncheva (mesure de diversité)
EoC	Ensemble of classifiers
FDP	Fonction de densité de probabilité
FM	Fault Majority (mesure de diversité)
k-**NN**	k-Nearest Neighbours (règle des k- plus proches voisins)
`MLP`	Multi-layer perceptron (réseau neuronal multicouche de type perceptron)
NFM	Normalized Fault Majority (mesure de diversité)
NIST	National Institute of Standards and Technology
NSGA	Non-Dominated Sorting Genetic Algorithm
$P(C_i)$	Probabilité a priori de la classe C_i
$P(C_i\|x)$	Probabilité de la classe C_i étant donné x
$P(x\|C_i)$	Probabilité d'observer x étant donné C_i

Q_{avr}	Mesure de diversité
q.e.d.	Quod erat demonstrandum (ce qu'il fallait démontrer)
RF	Reconnaissance de formes
SEA	Sous-espace aléatoire
TI	Théorie de l'information
UC	Usable Classifiers (stratégie de gestion des caractéristiques absentes)
VA	Valeur d'adaptation
VAF	Valeur d'adaptation fantôme
σ_{share}	paramètre contrôlant le diamètre de la niche dans NSGA

CHAPITRE 1

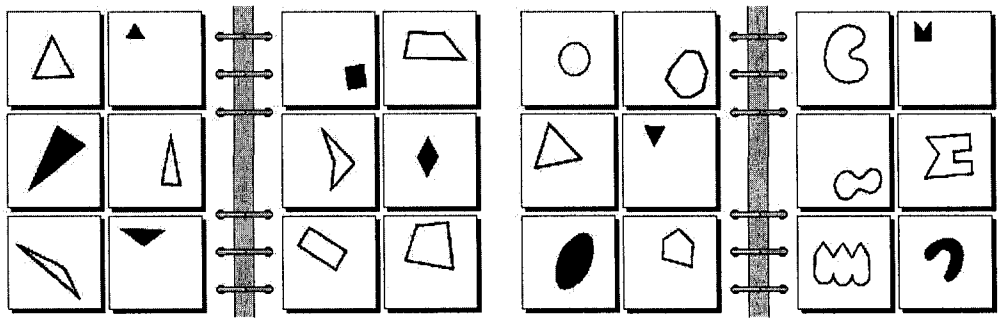
LA RECONNAISSANCE DE FORMES

Ce projet s'est déroulé dans le contexte de la reconnaissance de formes (RF) en général et plus particulièrement de la reconnaissance de chiffres manuscrits isolés. Pour effectuer la tâche de reconnaissance, de plus en plus d'auteurs proposent l'utilisation d'ensembles de classifieurs (**EoC**), c'est-à-dire le regroupement de plusieurs «machines» à reconnaître en comité d'experts. Cette façon de procéder permet souvent d'augmenter le taux de reconnaissance d'un système. Ce mémoire se propose d'explorer de nouvelles façons de créer un tel comité d'experts afin que ce dernier soit performant tout en conservant une relative simplicité.

1.1 Introduction

La classification est un domaine d'application et de recherche où des automates, souvent des programmes informatiques, ont à prendre des décisions quant à l'appartenance d'un objet à une classe. Ce problème est considéré être un sous-domaine de la RF, cette dernière étant une branche de l'intelligence artificielle [37]. Un système de reconnaissance de visage permet de décider si celui-ci appartient ou non à une personne donnée, tandis que certains moteurs de recherche tentent de déterminer la langue dans laquelle un texte électronique a été écrit. Ces deux exemples sont des domaines où l'être humain excelle à reconnaître des formes. D'autres applications ne sont cependant pas envisageables sans automatisation : la recherche de patrons répétitifs dans des séquences d'ADN en est un exemple. Même lorsque l'humain est performant à reconnaître certains types de formes, il peut être avantageux d'automatiser la tâche surtout en cas de fort volume de données à traiter : le tri des enveloppes postales en est une bonne illustration.

L'émergence des ordinateurs et de l'informatique au milieu du XX^e siècle a permis au domaine de la RF d'évoluer parallèlement. Un des premiers problèmes à faire surface



a) b)

Figure 1 **Problèmes de Bongard [7].** Dans le problème a), tous les exemples de la page de gauche sont des triangles alors que sur la page de droite, les exemples ne sont pas des triangles (en l'occurrence, ce sont des quadrilatères). La solution au problème b) est laissée en exercice au lecteur. Reproduit à partir de [22].

concerne la représentation du monde réel. En effet, tout dans un ordinateur est ultimement représenté par une série d'unités binaires, ce qui n'est pas nécessairement la façon dont l'humain procède. Dans les années 50, Glantz se demande s'il ne faudrait pas tenter de mieux comprendre les processus mentaux sous-jacents à la reconnaissance de formes par l'humain pour faire progresser la RF automatique [26]. Dans les années 60, Bledsoe fait apparaître le problème de la RF dans le contexte très fréquent et très réel où un grand nombre de classes sont possibles : contrairement aux lettres de l'alphabet latin, le nombre d'empreintes digitales ou de visages humains différents est énorme [5]. À la même époque, Prather [61] explore des méthodes d'induction et d'apprentissage automatique dans le but de créer un éventuel automate de reconnaissance «tout usage.»

Les problèmes de Bongard, popularisés par Hosfstadter [34], synthétisent à eux seuls tous ces aspects du problème de la reconnaissance de formes (voir les deux exemples de la figure 1). Ils posent à la fois le problème de la représentation des objets, de l'apprentissage et du nombre illimité de formes possibles. À la fin des années 60, Mikhail Moiseevich Bongard proposa 100 problèmes qui devaient être un défi pour l'intelligence artificielle [7]. La résolution de ces problèmes par des automates demeure toujours très délicate [52]. Ces

derniers se présentent sous la forme de deux ensembles de six images. Celles du côté gauche sont toutes conformes à une même règle ou sont créées à partir d'un même patron, alors que l'ensemble des figures du côté droit n'obéissent pas à cette règle. L'objectif du *reconnaisseur* de formes (humain ou machine) est de trouver cette règle, c'est-à-dire de déterminer en quoi un ensemble de formes est différent d'un autre. Si les solutions des exemples de la figure 1 sont plutôt triviales pour un humain, il n'en est pas de même pour un automate. Les problèmes de Bongard, par leur généralité, nous renvoient à l'essence même de l'intelligence humaine.

Le problème de l'intelligence artificielle en général et de la RF en particulier est si vaste qu'il est nécessaire de l'aborder par sous-problèmes. La classification est le domaine de la RF auquel le présent travail s'est intéressé. Dans ce domaine, on appelle classifieur[1], parfois expert, un programme informatique dont le rôle est de décider de quel type est un objet (quelle est sa classe) en fonction des données fournies en entrée. Par exemple, en communication numérique, le classifieur détermine quel symbole a été émis en observant le signal capté par le récepteur, tandis qu'en reconnaissance de chiffres manuscrits, il doit décider quel est le chiffre qui lui est présenté à partir d'une image.

Pour prendre une décision, l'expert doit se baser sur de la connaissance *a priori* qui prend souvent la forme d'exemples connus, appelés prototypes. L'ensemble des prototypes est appelé la base d'apprentissage ou base d'entraînement. Ces prototypes sont souvent représentés par des vecteurs de caractéristiques où chaque composante est une mesure faite sur les objets réels (amplitude et phase d'un signal électrique, nombre de pixels noirs d'une image, hauteur d'un vérin dans un système mécanique) ou un attribut qualitatif (vert, rond, chaud, etc.). Chaque caractéristique devient donc un axe dans un espace ayant autant de

[1] Bien que le terme classifi cateur soit aussi parfois employé, l'Offi ce québécois de la langue française préconise le terme classifi eur dans le contexte d'un «instrument de l'étape de reconnaissance qui affecte le caractère à une classe de formes par l'identifi cation de certaines caractéristiques». Pour l'OQLF, un classifi cateur est une personne qui effectue de la classifi cation tel un taxinomiste.

dimensions que la cardinalité de l'ensemble des caractéristiques, tandis qu'un prototype est un point dans cet espace.

Le livre de Duda, Hart et Stork constitue une excellente référence sur le domaine de la RF [21].

1.2 Le problème de la reconnaissance de formes

Comment peut-on faire reconnaître des formes à une machine ? La figure 2 montre un schéma général d'un système de classification. La connaissance *a priori* prend la forme d'une base de données d'entraînement sur laquelle le système de reconnaissance se base pour comparer une observation dont on cherche la classe d'appartenance. Cette connaissance est fournie par l'expert humain qui structure et les données et les étiquettes. Habituellement, on demande à un tel système de ne prendre de décision que si un certain seuil de certitude a été atteint, dans le cas contraire l'observation n'est pas classée (elle est rejetée). Ceci est vrai en mode opérationnel (lors de la mise en service du système de RF) et peut aussi l'être au moment de l'apprentissage. Si le seuil est atteint, une décision est prise et l'observation est assignée à une classe. Cette décision est bonne si l'observation est assignée à sa classe réelle et mauvaise sinon. Lorsqu'on mesure les performances d'un système, on utilise une autre base de données étiquetées (c'est-à-dire, une base dont on connaît la classe de chacune des observations). Cette nouvelle base, appelée base de test, ne doit pas avoir servi de connaissance *a priori* pour ne pas biaiser la mesure. Le taux de reconnaissance est défini comme étant le rapport entre le nombre de données reconnues correctement et le nombre de données présentées à l'entrée du système.

$$\textit{Taux de reconnaissance} = \frac{\textit{Bonnes décisions}}{\textit{Nombre total d'exemples}} \times 100 \tag{1.1}$$

Lorsqu'on veut tenir compte des observations rejetées, on mesurera la fiabilité, c'est-à-dire le rapport entre le nombre de bonnes décisions et le nombre de données pour lesquelles

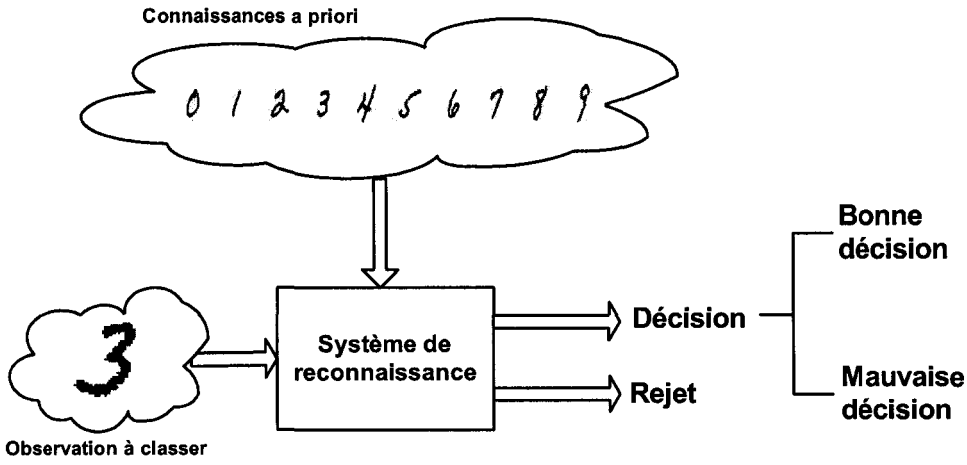


Figure 2 **Procédé de la RF.** Principaux concepts autour d'un système de reconnaissance de formes. L'entrée est composée d'une observation à classer. À l'aide de connaissance *a priori*, le système va rendre une décision qui peut être acceptée ou rejetée. En cas d'acceptation, deux cas sont possibles : une bonne ou une mauvaise décision.

une décision a été prise.

$$\textit{Taux de fiabilité} = \frac{\textit{Bonnes décisions}}{\textit{Nombre de décisions}} \times 100 \tag{1.2}$$

Le système de reconnaissance peut être décomposé en plusieurs étapes tel qu'illustré à la figure 3. Les différentes parties du système sont décrites ci-dessous.

- **La détection**

  La détection consiste à convertir en signaux électriques, à l'aide de transducteurs, des grandeurs physiques mesurées sur l'objet à reconnaître. Par exemple, un microphone convertit une variation de pression de l'air en tension électrique. Ce signal, une fois numérisé, sert d'entrée au système. L'étude et la conception de transducteurs est un vaste domaine regroupant plusieurs disciplines, dont la physique et l'ingénierie.

- **Le prétraitement**

  Lors de l'étape de prétraitement, les signaux d'entrées sont mis en forme et traités

de façon à rendre possible l'étape subséquente d'extraction de caractéristiques. Par exemple, c'est à cette étape qu'un signal sera filtré, qu'une image sera binarisée, normalisée ou redimensionnée. Avant de passer à l'étape suivante, on devra aussi très souvent segmenter le signal. La segmentation consiste à séparer le signal en unités de base indivisibles. Par exemple, en reconnaissance de la parole, la segmentation se fait souvent au niveau des phonèmes qui constituent les mots à reconnaître alors qu'en reconnaissance d'écriture cette segmentation pourrait se faire au niveau des lettres. On comprend tout de suite que l'étape de segmentation est intrinsèquement liée à tout le processus de RF. En effet, comment peut-on segmenter un mot en lettres sans reconnaître d'abord un ensemble de pixels comme étant une lettre ? Prendre les arêtes ou toute autre unité de base plus petite ne fait que repousser le problème à un autre niveau. L'étape de segmentation est donc elle-même un processus de reconnaissance de formes de plus bas niveau ; processus qui, à son échelle, devra sans doute lui-même passer par une étape de segmentation. Ce paradoxe fait de la segmentation un des grands problèmes de la RF puisqu'il pose celui ontologique de l'existence *a priori* d'unités indivisibles dans les formes à reconnaître. Et si de telles unités n'existent pas, une question épistémologique s'impose : «Quel est le rôle de la segmentation ?» Or, expérimentalement, on sait que cette étape, bien que nécessairement imparfaite, facilite généralement le processus de RF. Ces questions dépassent les domaines généralement abordés par les membres de la communauté de la RF mais constituent le coeur du problème. Bien qu'une branche de la philosophie, appelée méréologie, se penche spécifiquement sur l'étude des relations entre les touts et leurs parties, en pratique l'automatisation de la segmentation se fait généralement à partir d'heuristiques déterminées empiriquement pour maximiser la performance d'un système.

- **L'extraction de caractéristiques**

  Dans les années 70, Zobrist réfléchissait à la question de la représentation des données [79] et concluait que la reconnaissance de formes serait mieux servie par des

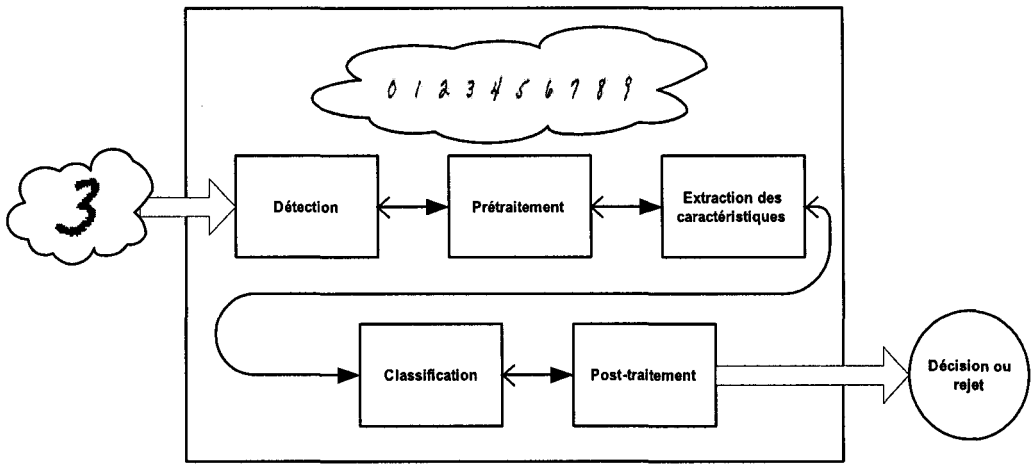


Figure 3 **Schéma d'un système de reconnaissance de formes.** Précision sur le fonctionnement d'un système de reconnaissance de formes, ici divisé en 5 étapes. Le sens des flèches indique la principale direction du flux de données. Notons toutefois qu'un rétroaction est possible entre toutes ces étapes et que le schéma pourrait se complexifier avec des flèches partant de toutes les étapes pour revenir à n'importe quelle étape précédente.

machines utilisant une représentation interne des objets à reconnaître qui leur est propre, en opposition avec représentation compréhensible par les humains. Bien que la question ne soit certainement pas entièrement réglée, c'est aujourd'hui la façon de faire la plus courante. L'extraction de caractéristiques consiste donc à transformer une forme en une représentation interne à l'automate de classification. Elle est constituée plus précisément d'une série de mesures faites sur des objets (où des segments d'objets) à reconnaître. Par exemple, on peut mesurer la puissance d'un signal à des fréquences bien définies de son spectre, on peut aussi compter le nombre de pixels d'une certaine intensité ou calculer les moments statistiques d'une distribution (une distribution spatiale de pixels ou une distribution temporelle de fréquences par exemple). L'ensemble de ces mesures forme une représentation des objets qui sera utilisée à l'étape de classification. La difficulté ici est de trouver de bonnes caractéristiques. De « bonnes » caractéristiques permettent aux classifieurs de reconnaître facilement les différentes classes d'objets ; on dit alors qu'elles sont discri-

minantes. Elles doivent aussi être invariantes à certaines transformations (la lettre **A** appartient à la même classe quelle que soit sa taille). À la limite, des caractéristiques parfaitement discriminantes permettraient de déterminer automatiquement à quelle classe appartient un objet. Mais alors, le module d'extraction de caractéristiques deviendrait lui-même un module de classification. En fait, la distinction entre la partie d'extraction de caractéristiques et de classification est arbitraire d'un point de vue conceptuel mais utile en pratique dans l'implantation de systèmes réels. Un extracteur de caractéristiques peut alors prendre la forme d'un programme qui produira des représentations de données qu'un classifieur pourra utiliser ultérieurement. Ainsi, pour comparer différents classifieurs on choisira la même représentation des même données ; pour comparer le pouvoir discriminant de différentes caractéristiques, on utilisera les mêmes classifieurs avec différentes représentations d'une même base de données.

- **La classification**

  Le classifieur a pour fonction d'assigner une étiquette de classe à l'objet qui lui est présenté par l'extracteur de caractéristiques. Un avantage pratique de la séparation entre l'extraction de caractéristiques et la classification est que de cette façon, il est possible d'étudier les classifieurs hors de leur domaine d'application spécifique ; ceci grâce au niveau d'abstraction suffisamment élevé produit par la représentation des objets sous forme de vecteurs de caractéristiques. Le travail du classifieur consiste à séparer l'espace de représentation par des frontières[2] et à assigner des étiquettes de classe aux régions ainsi formées. La région où se trouve un vecteur de caractéristiques détermine donc sa classe d'appartenance. Généralement, il n'est pas possible de bâtir une partition parfaite de l'espace, aussi le rôle du classifieur sera souvent de donner une probabilité d'appartenance d'un objet à une classe. De plus, dans les applications réelles, le système doit composer avec un certain bruit

[2] Ces frontières peuvent avoir une forme analytique (hyperplans, hyperparaboloïdes, etc.) ou arbitraire telle une «hyperpatatoïde».

ou biais (qu'il soit gaussien et dû aux transducteurs de l'étape de détection, ou qu'il soit de toute autre nature comme un dérèglement progressif, une interférence avec un autre appareil, etc). Souvent, il prendra la forme de caractéristiques absentes (un capteur peut être défectueux, une donnée peut ne pas être disponible). Le classifieur doit pouvoir gérer ces imperfections, et la façon d'entraîner des classifieurs pour y faire face est un domaine de recherche en soi.

- **Le post-traitement**

  L'étape ultime du processus de RF, appelée post-traitement, regroupe toutes les actions à prendre lors de la classification ou non de l'objet à classer (fusion des sorties de plusieurs classifieurs, évaluation d'un seuil de confiance, décision de classer ou de rejeter un objet). S'il y a classification, on assignera l'objet à une classe alors qu'on le rejettera dans le cas contraire. Cette décision peut se prendre à partir des probabilités émises par le classifieur mais aussi grâce à d'autres sources d'information émanant du contexte par exemple. Le post-traitement peut aussi être l'étape de fusion de multiples résultats de classification dans un contexte d'ensembles de classifieurs.

On remarquera, à la figure 3, que les flèches sont bidirectionnelles. En effet, une flèche plus petite relie les différentes étapes du processus de RF dans le sens opposé au sens présenté plus haut. Ceci a pour but d'illustrer qu'il y a souvent une rétroaction entre les différentes étapes de reconnaissance et que les résultats obtenus en aval peuvent être réinjectés en amont. De tout cela, il faut donc retenir que les différents étages d'un système de RF n'ont pas nécessairement des frontières nettes et que chacune des étapes peut avoir une influence sur toutes les autres.

Le présent travail peut être vu comme portant principalement, mais non exclusivement, sur les étapes de post-traitement ou d'extraction de caractéristiques. En effet, il s'agit essentiellement d'étudier des façons de générer des ensembles de classifieurs et donc de mesurer leur impact au niveau du post-traitement, tous les classifieurs étant identiques. D'un autre

côté, chaque classifieur utilise une représentation différente des objets, chacun ayant accès à des sous-ensembles de caractéristiques lui étant propres (représentation partielle de l'information). Dans ce cas, la création d'ensembles de classifieurs devient indirectement une sélection de caractéristiques.

1.2.1 Les approches de classification

On peut décomposer les classifieurs en deux grandes familles, l'une utilisant une approche dite structurelle et l'autre une approche statistique. L'approche structurelle [53, 56, 21] exploite la topologie des structures élémentaires des objets pour définir un certain nombre de règles définissant l'appartenance ou non à une classe. Font ainsi partie de cette catégorie : les arbres de décision, les systèmes experts et les programmes d'analyse syntaxique. Il est intéressant de noter que les bases qui ont permis l'émergence de ces derniers ne viennent non pas de la communauté de la RF ni même de celle de l'informatique au sens large, mais de Noam Chomsky qui introduisit dans les années 50 l'utilisation de règles dans les grammaires formelles permettant la génération de phrases et de syntagmes. Ses travaux portaient sur la linguistique et le langage naturel [11]. Dans les années 60, Rosenfeld prédit que le développement de «langages graphiques»[3] était en voie de devenir le thème majeur de la recherche en reconnaissance de formes graphiques [62]. Si l'approche syntaxique n'a pas rempli toutes les promesses attendues à l'époque, elle n'en demeure pas moins une manière intéressante de représenter des objets pour permettre la classification.

Les caractéristiques d'entrée des classifieurs basés sur l'approche structurelle -appelées primitives- ne sont pas constituées de vecteurs de valeurs réelles mais plutôt d'une liste d'attributs semi-objectifs (grand, ouvert, bleu pâle, etc.), de valeurs booléennes ou d'une décomposition de l'objet en parties élémentaires idéalisées (ligne verticale, cercle, phonème, etc.) Ces caractéristiques, tout comme les règles régissant leurs interactions, peuvent être définies par un expert humain ou apprises automatiquement à l'aide de techniques

[3] *picture languages*

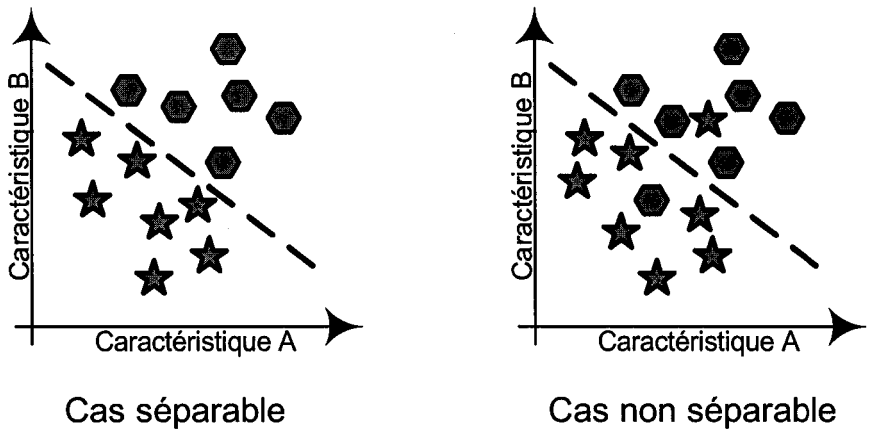


Figure 4 **Frontières de décision.** Deux ensembles sont linéairement séparables lorsqu'on peut trouver un hyperplan H tel que de part et d'autre de H il n'y ait que des éléments du même ensemble.

d'inférence. Il en est de même pour les classifieurs statistiques de sorte que les deux approches ne sont pas fondamentalement différentes.

L'approche statistique est basée sur des caractéristiques ayant la forme d'un vecteur de d valeurs numériques (généralement réelles mais pas toujours) [38]. Un objet, une observation à classer, devient donc un point dans $\mathbb{R}^d$, l'espace des caractéristiques. La connaissance de la distribution spatiale des classes permettrait théoriquement de catégoriser -donc de reconnaître- instantanément les objets ainsi représentés. En pratique, il n'est pas possible d'extraire des caractéristiques parfaitement discriminantes pour des problèmes non-triviaux. Les frontières séparant les classes dans l'espace des caractéristiques, appelées frontières de décision, doivent donc être le résultat d'un certain compromis entre le pouvoir de généralisation et celui de mémorisation[4]. Le cas simple -linéairement séparable- et le cas réel -non linéairement séparable- sont illustrés à la figure 4 dans un espace à deux caractéristiques.

[4] Par généralisation, on entend la capacité qu'a un classifi eur de reconnaître correctement des nouvelles observations alors que la mémorisation est sa capacité à bien classer les exemples qui lui ont servi à l'apprentissage.

L'apprentissage des classifieurs statistiques est donc une recherche de ces frontières de décision. Le terme statistique attribué à ce type de classifieurs provient de ce qu'ils sont basés sur la théorie bayesienne de la décision [75, 21, 56, 38]. En effet, si l'on connaît la fonction de densité de probabilité (FDP) conditionnelle des classes à traiter, la théorie bayesienne fournit tous les outils nécessaires pour prendre la meilleure décision possible en minimisant la probabilité d'erreur. On note $P(C_i)$ la probabilité *a priori* d'observer la classe i, elle est indépendante de toute mesure faite sur l'objet à classer. On considère le vecteur de caractéristiques x comme étant une variable aléatoire dont la distribution dépend de la classe observée. Les différentes distributions de x sont notées $P(x|C_i)$ et sont appelées probabilités conditionnelles puisque qu'il s'agit de la probabilité de mesurer le vecteur x sur une instance de la classe i. Dans un problème à N classes, si on connaît les probabilités *a priori* $P(C_i)$ et les probabilités conditionnelles $P(x|C_i)$ pour $i = 1, 2, ..., N$, alors, en mesurant effectivement le vecteur x, on peut transformer ces connaissances en probabilités *a posteriori*. Cette probabilité $P(C_i|x)$ d'observer la classe C_i sachant que l'on a mesuré les caractéristiques x est celle qui permet au classifieur de prendre un décision. En effet, la règle de décision de Bayes pour minimiser la probabilité d'erreur est la suivante :

Le classifieur choisit la classe C_j tel que :

$$P(C_j|x) = \max_{i=1}^{N} P(C_i|x) \tag{1.3}$$

L'équation 1.4 montre comment obtenir $P(C_i|x)$ à partir de $P(C_i)$ et de $P(x|C_i)$.

$$P(C_i|x) = \frac{P(x|C_i) \cdot P(C_i)}{\sum_{j=1}^{N} P(x|C_j) \cdot P(C_j)} \tag{1.4}$$

La figure 5 montre la frontière de décision séparant deux classes dont les probabilités conditionnelles $P(x|C_i)$ sont distribuées selon une loi gaussienne. Cette figure illustre

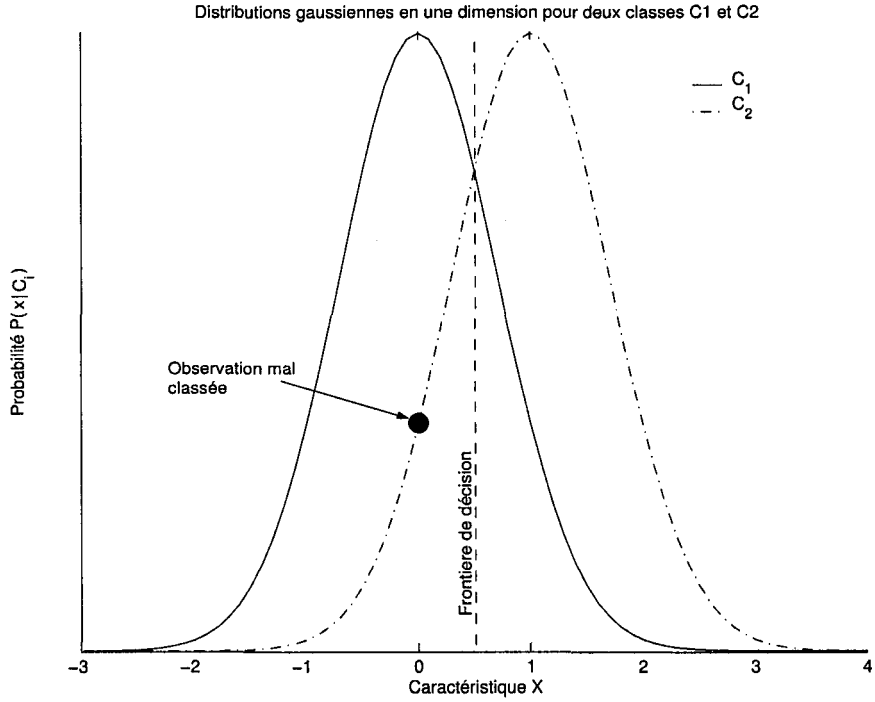


Figure 5 **Illustration de l'origine de l'erreur bayesienne.** Dans le cas de probabilités *a priori* identiques, si la caractéristique x mesurée sur la classe C_2 a une probabilité conditionnelle $P(x|C_2)$ inférieure à $P(x|C_1)$, alors le classifieur bayesien choisira la classe C_1 et l'observation sera mal classée. C'est un phénomène inévitable lorsqu'il y a recouvrement dans les distributions de $P(x|C_i)$.

l'origine de l'erreur bayesienne, c'est-à-dire la probabilité d'erreur minimisée par la règle de l'équation 1.3.

1.2.2 Les techniques non paramétriques

Dans le cas général, qui est aussi le cas le plus fréquemment rencontré en pratique, on ne connaît pas la FDP des différentes classes. Pour contourner ce problème, une approche consiste à faire l'hypothèse d'une FDP connue (souvent une gaussienne dans $\mathbb{R}^d$). On

parle alors de classifieurs paramétriques puisque leur apprentissage consiste à estimer les paramètres de la FDP [5].

Dans les faits, il est plutôt rare de rencontrer les distributions classiques [6] et elles estiment souvent mal les FDP réelles. Les méthodes non paramétriques sont utilisées lorsqu'on ne possède aucune connaissance *a priori* ou qu'on ne fait aucune hypothèse sur les FDP des données à classer. On peut alors soit estimer la FDP réelle, soit tenter d'obtenir directement la probabilité *a posteriori* $P(C_i|x)$, c'est à dire la probabilité qu'une observation appartienne à la classe C_i après en avoir extrait le vecteur de caractéristiques x. Cette estimation de $P(C_i|x)$ peut se faire en échantillonnant un certain volume de l'espace des prototypes et en établissant un simple rapport. Supposons un volume V centré sur x englobant k prototypes, dont k_i appartiennent à la classe C_i. On considère alors qu'une bonne estimation de $P(C_i|x)$ est :

$$P(C_i|x) = \frac{k_i}{k} \tag{1.5}$$

En déplaçant le volume V dans l'espace des prototypes, on peut estimer $P(C_i|x)$ en tout point et ainsi reconstruire une FDP. Le choix de V peut se faire selon deux approches : fixer la taille et la forme du volume ou fixer le nombre de prototypes à l'intérieur d'une hypersphère dont le rayon dépendra de la région échantillonnée. On montre que lorsque le nombre d'échantillons tend vers l'infini, les deux approches sont équivalentes puisqu'elle tendent toutes deux vers l'estimation de la vraie FDP. L'estimateur de Parzen est inspiré de la première approche. Plutôt que de prendre comme volume une hypersphère ou un hypercube, cet estimateur utilise une fenêtre (par exemple une gaussienne) qui module la contribution de tous les prototypes en fonction de leur distance au point d'analyse x. Un inconvénient de cette méthode est qu'il faille faire le choix d'une fonction pour la fenêtre et que cet ajustement demande un apprentissage.

[5] On sait, par exemple, qu'une gaussienne est entièrement caractérisée par ses deux premiers moments statistiques (moyenne et variance). L'estimation de ces deux paramètres par une méthode d'apprentissage fournit donc une estimation de la FDP.

[6] Gaussienne, exponentielle, Rayleigh, etc.

L'estimation de $P(C_i|x)$ par la seconde approche, celle des k-plus proches voisins, nécessite aussi l'ajustement d'un paramètre : celui du nombre de prototypes k contenus dans V. L'utilisation de cette méthode donne cependant une règle très simple pour la classification. En effet, maximiser la probabilité *a posteriori* $\frac{k_i}{k}$ revient à choisir la classe C_i la plus représentée parmi les k exemples du volume V. C'est la règle des k-plus proches voisins, classifieurs aussi appelé k-**NN** pour k-Nearest Neighbour.

Notons que dans le contexte d'ensemble de classifieurs, la fusion des différentes $P(C_i|x)$ dépend souvent d'une fonction basée sur le produit. Il suffit alors qu'un estimateur produise une valeur nulle pour que l'estimation de l'ensemble soit nulle. Pour minimiser ce phénomène de veto, Alkoot a proposé une façon différente d'estimer $P(C_i|x)$ qui ne permet pas de valeur nulle [1] :

$$P(C_i|x) = \frac{k_i + 1}{k + M} \tag{1.6}$$

avec $M =$ au nombre de classes.

1.3 La règle du plus proche voisin

Le classifieur utilisant la règle du plus proche voisin (voir algorithme 1) a été formellement proposé dès 1967 par Cover et Hart [13]. Depuis, ce classifieur a fait l'objet de nombreuses études et a été généralisé au k-**NN**. Un historique détaillé en est donné dans une monographie de Dasarathy [16], document regroupant plus d'une centaine d'articles portant sur les techniques reliées au k-**NN**.

On sait depuis l'article initial de Cover et Hart que si l'on dispose d'un ensemble de n prototypes pour l'entraînement et que $n \to \infty$, alors la probabilité d'erreur $P(E_{1-NN})$ d'un classifieur basé sur la règle du plus proche voisin (k-**NN** avec $k = 1$) est bornée par les valeurs suivantes : $P(E_b) \leq P(E_{1-NN}) \leq 2P(E_b)$, où $P(E_b)$ est la probabilité d'erreur bayesienne[7]. On peut donc dire que la moitié de l'information est contenue dans

[7] La probabilité d'erreur bayesienne est optimale car elle minimise la probabilité de mauvaise décision, connaissant les distributions des classes.

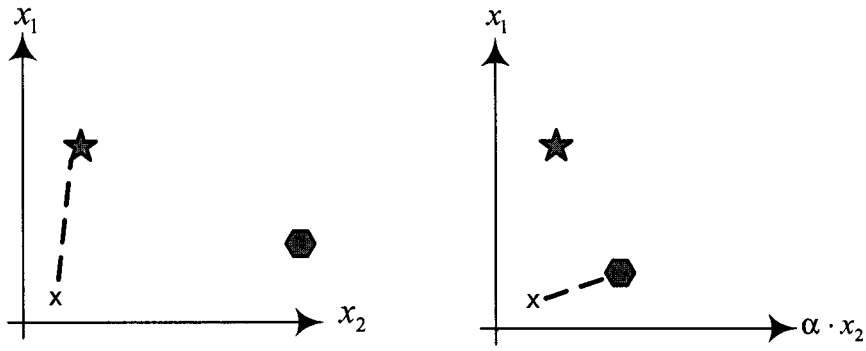


Figure 6 **Sensibilité à l'échelle des axes d'un *k*-NN utilisant la distance euclidienne comme métrique.** La distance la plus courte entre l'observation x et les deux plus proche prototypes n'est pas la même selon l'échelle des axes. Ce changement d'échelle est fréquent puisqu'il est la conséquence directe de la normalisation des données.

le plus proche voisin, si on dispose d'un ensemble d'entraînement infini. En pratique, cela demeure valable si n est *suffisamment grand* mais il n'y a pas de méthode pour déterminer à partir de quelle taille n peut être considéré suffisamment grand.

Le paramètre k joue un rôle au niveau de l'estimation de la probabilité *a posteriori*. À mesure que k augmente, l'estimation de $P(C_i|x)$se rapproche de la valeur réelle. À la limite, quand $k \to \infty$, l'estimation de la probabilité *a posteriori* est strictement égale à $P(C_i|x)$, ce qui entraîne que $P(E_{1-NN}) = P(E_b)$. On préférera donc des valeurs élevées de k pour augmenter la précision de l'estimation de $P(C_i|x)$. D'un autre côté, puisque les données d'entraînement sont en nombre fini, il faut s'assurer que les k voisins x' soient assez près de x de sorte que $P(C_i|x') \approx P(C_i|x)$. Le choix de k résulte donc d'un compromis et doit être ajusté à chaque problème.

On notera l'importance de la métrique ainsi que le choix ou non de normaliser les données. La distance euclidienne D_e, qui est certainement la plus utilisée, est définie en (1.7) pour un espace de dimension d. Cette distance est particulièrement sensible à la normalisation des données comme illustré à la figure 6 où la distance la plus courte change selon qu'il y

ait normalisation ou non de l'axe x_2. D'autres mesures de distance sont parfois utilisées, mentionnons la distance de Mahalanobis, distance pondérée par l'inverse de la matrice de covariance, puis les distances de Hamming et de Tanimoto pour des vecteurs binaires [21].

$$D_e = \sqrt{\sum_{0<f\leq d} (x'_f - x_f)^2} \tag{1.7}$$

Supposons n prototypes d'apprentissage dans un espace de caractéristiques de dimension d. Le calcul de la distance euclidienne entre une observation à classer et un prototype est de complexité $O(d)$. Pour trouver le prototype le plus proche d'une observation, la méthode la plus simple consiste à calculer la distance entre cette observation et les n prototypes et à comparer chaque distance obtenue avec celle que l'on considère la plus proche. La complexité est donc d'ordre $O(d \times n^2)$. Il existe néanmoins des techniques pour réduire la complexité [46, 54]. Utiliser des distances partielles, structurer les données sous forme d'arbre de recherche ou élaguer la base de données sont les points de départ de la plupart des méthodes visant à réduire la complexité des k-**NN**. Il faut cependant noter que souvent ces méthodes de réduction n'assurent pas de trouver le ou les plus proches voisins. La diminution de la complexité peut donc signifier l'augmentation de la probabilité d'erreur du système.

Malgré l'inconvénient de sa complexité élevée, le k-**NN** a l'avantage de se prêter particulièrement bien à l'apprentissage en-ligne [16]. En effet, sa complexité tient du fait que l'apprentissage se résume à stocker des données en mémoire sans aucun traitement, tous les calculs ne s'effectuant qu'au moment de classer une observation. Pour cette même raison, l'ajout de données supplémentaires acquises alors que le système est à l'oeuvre ne nécessite que de pouvoir ajouter ces dernières dans une base de données, contrairement à des systèmes basés sur une descente de gradient qui nécessitent un réapprentissage complet. De plus, la simplicité de sa mise en oeuvre et l'abondance de travaux à son sujet font du k-**NN** un classifieur intéressant pour fin de comparaisons avec d'autres systèmes pour

lesquels on dispose de moins de bases théoriques au niveau de la probabilité d'erreur par exemple.

Algorithme 1 : Classification à l'aide d'un k-**NN** [21]

Données :
O : l'observation à classer
D : l'ensemble des n prototypes et de leur étiquette, notés respectivement X^i et Y_i
k : le nombre de plus proches voisins à considérer
$knn(O, D, k)$: classifieur de type Plus Proche Voisin (PPV)
$Vote(Liste)$: une fonction de vote qui retourne la classe la plus représentée dans une liste et qui départage arbitrairement les cas d'égalité
$d(a, b)$: une fonction qui calcule la distance selon une métrique quelconque (souvent la distance euclidienne)

début
  $i \longleftarrow 0$
  répéter
    $i \longleftarrow i + 1$
    $dist \longleftarrow d(X^i, O)$
    $L_i \longleftarrow \{dist, Y_i\}$
  jusqu'à $i = k$
  Trier la liste L par ordre croissant des distance
  $PPV_i \longleftarrow L_i | i \in \{1, ..., k\}$
fin
retourner $Vote(PPV)$

1.4 Au-delà du classifieur

L'étude d'un système de RF ne se limite pas aux seuls classifieurs en tant qu'unité individuelle. L'erreur de reconnaissance peut être décomposée et ainsi permettre une caractérisation du système de RF ne dépendant pas du type de classifieur utilisé. Aussi, un système n'est pas nécessairement composé d'un seul classifieur mais peut être construit sur un ensemble de ceux-ci. Ces deux approches sont discutées dans les parties qui suivent.

1.4.1 Biais et variance

La probabilité d'erreur $P(E)$ d'un classifieur statistique est supérieure ou égale à la probabilité d'erreur bayesienne $P(E_b)$ puisque cette dernière est optimale. Cette optimalité

signifie qu'aucun classifieur ne peut avoir une probabilité d'erreur inférieure à $P(E_b)$, la figure 5 illustrant pourquoi. En général, les classifieurs ont une probabilité d'erreur non pas égale mais supérieure au cas bayesien idéal car ils doivent estimer $P(C_i|x)$ et que cette estimation est forcément imparfaite. Traditionnellement, dans le domaine de la régression statistique, on décompose cette erreur d'estimation $E(X)$ en biais $B(X)$ et en variance $V(X)$, X étant un vecteur d'échantillons de la courbe à estimer (équation 1.8).

$$E(X) = B(X) + V(X) \tag{1.8}$$

Cette façon de décomposer l'erreur a été adaptée au domaine de la classification. Le passage de la régression à la classification n'est pas direct et il a fallu attendre les travaux de Domingos pour avoir une technique de décomposition générale applicable à ce domaine [20]. Valentini et Dietterich utilisent depuis technique dans leur propres travaux pour caractériser des machines à vecteurs de support [74].

La décomposition de l'erreur en biais et en variance est utile pour comprendre le comportement d'un algorithme d'apprentissage et surtout la nature de ses erreurs. La variance correspond aux erreurs dues aux données d'entraînement. Ces données sont constituées d'un nombre fini d'échantillons et par conséquent l'estimation de $P(C_i|x)$ dépend directement de ces données. Un même algorithme entraîné avec deux bases de données d'entraînement différentes ne produira pas exactement les mêmes résultats. C'est cette différence que mesure la variance. Le biais est intrinsèque à un type de classifieur. C'est l'erreur produite indépendamment de la base d'apprentissage. Celle-ci dépend de la capacité du modèle à créer des frontières de décisions complexes. Pour augmenter cette capacité, on doit généralement augmenter le nombre de paramètres du classifieur, ce qui a pour conséquence de nécessiter plus de données d'entraînement pour en permettre une juste estimation. En diminuant la complexité du classifieur et en gardant constant le nombre de données d'entraînement, on augmente la précision de l'estimation de ses paramètres, ce qui diminue la variance mais augmente le biais. Cependant, bien que biais et variance ne soient pas indépendants l'un de l'autre, leur relation n'est pas linéaire et on n'en connaît pas d'ex-

pression analytique[8]. À la limite, si on dispose d'un classifieur suffisamment complexe pour exprimer les frontières de décision optimales, en augmentant le nombre n d'exemples d'apprentissage, on diminuerait l'erreur jusqu'à atteindre E_b pour $n \to \infty$.

1.4.2 Combiner des classifieurs

Plusieurs travaux ont démontré expérimentalement qu'un ensemble de classifieurs (**EoC**) pouvait être utilisé pour diminuer l'erreur de classification [33, 30, 4] alors que d'autres ont tenté d'expliquer les raisons de ce phénomène [42, 32, 18]. L'idée de combiner plusieurs experts indépendants pour augmenter la probabilité de bonne décision a été formalisée pour la première fois au XVIII[e] s. sous la forme du Théorème du jury de Condorcet, d'après le marquis du même nom. Ce théorème[9] est le suivant :

Théorème 1.4.1 *Soit un jury de n personnes. Chacune a une opinion : vraie ou fausse. On note k le nombre de jurés ayant une opinion vrai. Si p est la probabilité d'erreur de chaque juré (identique pour tous) et si $F(n)$ est la probabilité qu'une majorité de jurés aient la bonne opinion (donc que $k > \frac{n}{2}$), alors le Théorème du jury de Condorcet dit que :*

$$Si\ p < \frac{1}{2},\ alors\ \lim_{n\to\infty} F(n) = 1$$

En d'autres termes, la probabilité que le jury produise une majorité de bonnes opinions tend vers 1 quand le nombre d'experts dans le comité tend vers l'infini si chaque expert a une probabilité de décision supérieure à $\frac{1}{2}$ et qu'ils sont indépendants. Ce théorème n'est pas limité aux problèmes à deux classes (vrai ou faux) puisque tout problème à C classes peut se décomposer en une série de sous-problèmes à deux classes (approche un contre tous). Pour que $F(n)$ continue de tendre vers 1, il faut seulement que les jurés fassent mieux que ne le ferait le hasard.

[8] C'est vrai pour le domaine de la classifi cation car, dans celui de la régression, la relation entre biais et variance est connue et bien défi nie.

[9] Une démonstration est donnée à l'annexe A.

Dans le cadre de la RF, et donc de la classification, ces conditions ne sont pas remplies. Les experts ne sont pas complètement indépendants, ils ne sont pas en nombre infini et ils n'ont pas nécessairement une équiprobabilité d'erreur. Néanmoins, le théorème nous dit que si l'on possède un grand nombre d'experts, s'ils sont plus ou moins indépendants et relativement performants, on doit s'attendre à avoir de meilleurs taux de reconnaissance avec un comité qu'avec un seul expert [57].

Un autre argument en faveur des ensembles vient de la variance dans la décomposition de l'erreur. Cette variance, due aux données d'apprentissage qui ne représentent qu'un échantillon imparfait des objets réels, devrait normalement diminuer si plusieurs experts sont entraînés différemment. On sait du moins que d'un point de vue statistique, la variance tend à diminuer au fur et à mesure que le nombre d'échantillon augmente. Chaque expert, entraîné différemment, peut être vu comme le résultat d'un échantillonnage qui précise l'estimation de la moyenne tout en diminuant la variance autour d'elle.

Si intuitivement l'utilisation d'ensembles de classifieurs semble intéressante, beaucoup de questions restent ouvertes. En effet, comment choisir ces experts ? Quel type de classifieurs choisir et combien puisque en pratique ce nombre devrait être fini et qu'un trop grand nombre d'experts rendrait la technique prohibitive en terme de ressources. Comment intégrer la notion d'indépendance entre les classifieurs ? Est-ce que la règle du vote à majorité simple, utilisée dans le Théorème du jury, est la meilleure façon de fusionner les opinions des différents experts ? Ces questions n'ont toujours pas de réponse et impliquent encore beaucoup de recherche active.

1.5 Problématique

L'un des défis de la reconnaissance de formes est de créer des systèmes qui soient à la fois simples et performants. Par performant on entend un système ayant des taux de reconnaissance élevés alors que simple signifie facile à réaliser et ayant un faible coût en ressources matérielles. Un système ayant beaucoup de classifieurs est considéré complexe

alors qu'un système ayant peu de classifieurs est simple ou peu complexe. Il est démontré que les ensembles de classifieurs peuvent permettre d'obtenir de meilleures performances qu'un classifieur unique [58, 31, 47, 14]. Par contre, la simple agrégation de classifieurs n'est pas optimale et entraîne une complexité qui peut devenir prohibitive.

Le présent mémoire tente d'apporter une réponse à la question de savoir comment construire un système de RF basé sur des **EoC** de *k*-**NN** qui est à la fois simple et performant ?

Le second chapitre traite plus spécifiquement des **EoC** et définit les concepts qui sont utilisés dans nos hypothèses de recherche. Ces dernières sont exposées à la fin du chapitre qui suit. Le troisième chapitre montre les méthodes employées et le protocole des expériences menées pour vérifier nos hypothèses. Les principaux résultats obtenus sont reportés au quatrième chapitre. On trouvera en dernière partie une analyse plus fine des résultats ainsi que la conclusion de ce travail.

CHAPITRE 2

LES ENSEMBLES DE CLASSIFIEURS

L'utilisation d'ensembles de classifieurs permet souvent d'augmenter la performance d'un système de reconnaissance de formes par rapport à un système mono-classifieur. En revanche, les **EoC** forment des systèmes plus complexes qui demandent plus de ressources à mettre en oeuvre. Bien que le principal objectif de tout système basé sur un ensemble de classifieurs soit d'en maximiser la performance, en limiter la complexité fait souvent partie des contraintes importantes.

Le présent chapitre présente les **EoC** et en introduit le formalisme. Par la suite, une revue des méthodes actuelles de création d'ensembles de classifieurs en précède une autre consacrée à leur optimisation. Finalement, le cadre spécifique de notre recherche est exposé. Celui-ci consiste à étudier des façons d'optimiser un ensemble de classifieurs en tenant compte de l'objectif de performance et de la contrainte de complexité mentionnés plus haut.

2.1 Formalisation des EoC

Une manière de formaliser le problème de classification est la suivante : soit la fonction inconnue $y = f(x)$ avec $x_i \mid i \in \{1, ..., n\}$ étant des objets à classer et $y_i | i \in \{1, ..., n\}$ les étiquettes des classes correspondantes. Un classifieur peut alors être vu comme une fonction $h(x)$, appelée hypothèse, qui tente d'approximer au mieux la fonction $f(x)$. Par conséquent, l'apprentissage d'un ensemble de classifieurs consiste à construire un ensemble de N hypothèses et à les combiner de manière à classer des objets inconnus. La figure 7 montre ce principe général.

Bien que les conditions du théorème du Jury de Condorcet ne soient pas réunies, il a été montré expérimentalement que l'utilisation de **EoC** donne généralement de meilleures performances que l'utilisation d'un seul classifieur, et ce indépendamment de la nature

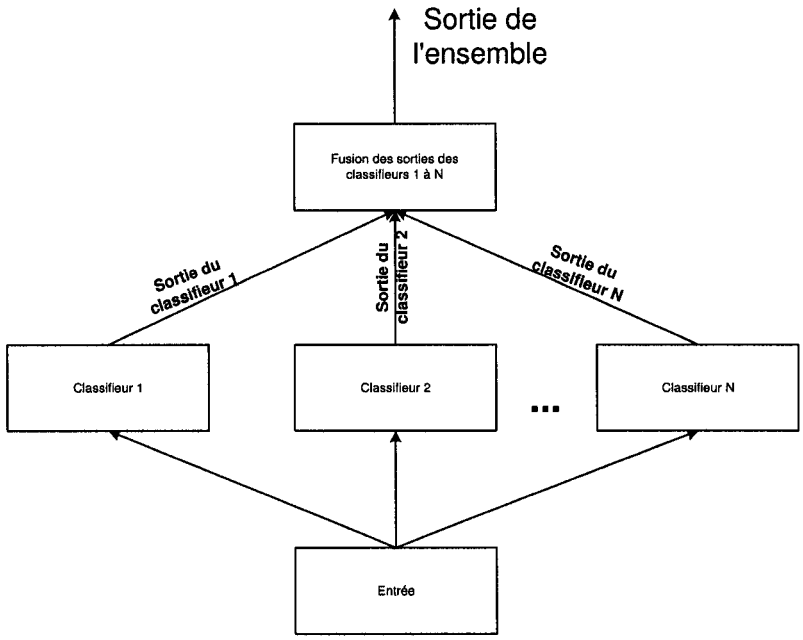


Figure 7 **Architecture d'un ensemble de classifieur.** Architecture d'un ensemble de classifieur utilisant une topologie parallèle.

de ce dernier. Mentionnons à ce sujet les travaux de Ho sur les arbres de décision et les k-**NN** [30, 33], ceux de Kim sur les machines à vecteurs de support [40] et les résultats de Oliveira avec des réseaux neuronaux [58]. Dietterich, quant à lui, a présenté les trois raisons suivantes pour expliquer l'efficacité des **EoC** [18, 19] :

1. **Diminution de la variance statistique** - Un algorithme d'apprentissage a une grande variance lorsqu'il peut produire des hypothèses $h(x)$ qui semblent bien approximer la fonction recherchée $f(x)$ avec les données d'entraînement x mais que ces dernières produisent des résultats très différents sur des données de généralisation. En combinant plusieurs hypothèses, il est possible que cette variance soit diminuée.
2. **Diminution de biais** - La représentation des données ne permet pas toujours d'exprimer $f(x)$, d'où un biais du classifieur qui ne pourra jamais s'approcher de la fonction recherchée plus que ne le permet l'espace des caractéristiques. Mais la

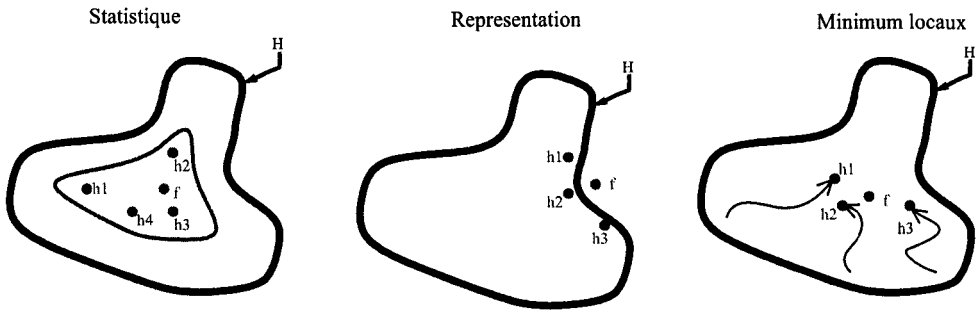


Figure 8 **Représentations de Dietterich pour les EoC [19].** Trois arguments de Dietterich pour expliquer l'efficacité des **EoC** : Diminution de la variance statistique, diminution du biais dû à un problème de représentation et intégration des minimums locaux.

combinaison de plusieurs hypothèses peut permettre d'élargir cet espace. Ainsi, il est possible qu'en combinant plusieurs hypothèses, on parvienne à se rapprocher de $f(x)$ plus qu'aucun classifieur unique ne pourrait le faire.

3. **Intégration des minimums locaux** - Plusieurs algorithmes d'apprentissage sont basés sur des techniques exploitant les minimums locaux et, par conséquent, sont sujets à demeurer «prisonniers» d'un de ces optimums. Un ensemble d'algorithmes, parce que pris dans un ensemble de minimums différents, peut permettre de mieux estimer $f(x)$.

Il est évident que la combinaison d'hypothèses identiques ne saurait donner de meilleurs résultats que l'utilisation d'un seul classifieur. De la même façon, la combinaison d'hypothèses, qui n'est autre que la fusion des opinions des classifieurs, doit être faite de façon à améliorer ces résultats. Un **EoC** sera plus performant qu'un classifieur unique si et seulement si on peut générer des classifieurs différents et combiner leurs sorties de façon judicieuse. Ces deux points feront l'objet des prochaines sections.

2.1.1 Topologie des ensembles de classifieurs

La combinaison des classifieurs ne dépend pas seulement de la fonction de fusion, mais aussi de la topologie de ces derniers. Lam a proposé la classification suivante pour décrire les principales topologies utilisées [51].

- **Conditionnelle** - Dans ce type d'organisation, un premier système de classification prend la décision de classer ou de rejeter un objet. Si ce dernier est rejeté, un second système prend la relève et ainsi de suite. En général le premier étage est constitué d'un classifieur efficace en terme de ressources (mémoire ou temps de calcul). Son rôle est de classer les objets les plus faciles à reconnaître et de laisser aux autres étages, plus complexes, les objets difficiles. C'est une façon de minimiser l'utilisation des ressources ou de réduire le temps moyen de reconnaissance d'un système.
- **Sérielle** - Il s'agit ici de construire un système où les classifieurs sont utilisés les uns à la suite des autres, chacun réduisant l'ensemble de classes possibles pour un objet à classer. Ainsi, un classifieur étant indécis et hésitant entre N classes passera l'information au classifieur qui le suit. Ce dernier produira un ensemble de $M \mid M < N$ classes possibles et ainsi de suite jusqu'au dernier classifieur qui devra n'en retenir qu'une seule. C'est une façon de réduire un problème complexe en une série de problèmes plus simples.
- **Parallèle** - L'organisation parallèle des experts d'un **EoC** implique que chaque classifieur individuel produise une sortie simultanée. L'ensemble de ces sorties est ensuite fusionné de façon à produire une décision unique (voir la figure 7). Si cette dernière organisation est la topologie la plus courante, c'est aussi celle qui a le plus grand coût en ressources.

2.2 Génération d'ensembles de classifieurs

La génération d'un ensemble de classifieurs passe par l'entraînement des experts le constituant. Il n'y aura aucun gain à espérer si ces experts sont identiques ; il faut, par consé-

quent, créer des experts différents. Comment créer cette différence lorsque tous les classifieurs sont basés sur le même algorithme et disposent des mêmes données d'apprentissage ? Breiman explique que pour créer des experts différents, on doit faire varier des paramètres pour lesquels le classifieur est instable [8]. Autrement dit, il faut qu'une faible variation de ces paramètres entraîne la création d'hypothèses éloignées les unes des autres. Ces paramètres dépendent de la nature de l'algorithme d'apprentissage utilisé.

Il existe plusieurs stratégies pour créer des experts différents. On peut, par exemple, modifier les paramètres internes aux classifieurs (en faisant varier le nombre de neurones sur la couche cachée d'un MLP par exemple). Pour créer des classifieurs différents, on peut aussi tout simplement utiliser des ensembles de classifieurs hétérogènes, c'est-à-dire basés sur des algorithmes différents (MLP, *k*-**NN**, classifieurs bayesien, etc.). Par contre, lorsque tous les classifieurs sont basés sur le même algorithme et disposent des mêmes données d'apprentissage, la manipulation de ces dernières est la méthode à privilégier.

La manipulation des données d'entrée peut se faire de différentes façons. L'une d'entre-elle est d'ajouter un bruit aléatoire aux données d'entraînement. Ainsi chaque classifieur est entraîné avec une base de données légèrement différente, puisque bruitée différemment ; il en résulte des classifieurs qui n'ont pas des opinions identiques.

D'autres transformations existent. On peut échantillonner la base de données d'entraînement et faire apprendre les différents classifieurs avec des sous-bases issues de cette manipulation. On peut alors soit utiliser des sous-ensembles différents de prototypes, soit des sous-ensembles différents de caractéristiques. Dans le premier cas, les classifieurs conservent l'intégralité de leurs caractéristiques. Dans le second cas, c'est l'espace de représentation qui est échantillonné : chacun des experts est entraîné avec l'ensemble des exemples disponibles mais n'a accès qu'à une partie des caractéristiques. C'est ainsi qu'on parle de classifieurs ayant une représentation partielle de l'information. Les sous-sections qui suivent détaillent ces deux dernières approches.

Une des difficulté de la création d'experts différents est justement de définir et de mesurer leur différence d'opinion. On désigne plus souvent cette différence par le terme de diversité d'opinion ou simplement de diversité.

Lorsqu'un certain nombre de classifieurs différents ont été générés, on peut les utiliser directement pour créer un **EoC** ou passer par une étape de sélection de classifieurs pour choisir ceux convenant le mieux à un ensemble pour un problème donné, en utilisant éventuellement une mesure de diversité pour guider la sélection. Une fois l'**EoC** construit, on obtient un ensemble pouvant produire au moins autant de sorties qu'il y a de classifieurs le composant. La fusion de ces sorties ainsi que tout le post-traitement qui en découle fait aussi partie des paramètres à prendre en considération lors de la définition d'un **EoC**. La mesure de la diversité, la sélection de classifieurs ainsi que la fusion sont abordées plus bas.

2.2.1 Échantillonner les données d'entraînement

La modification de la base de données d'entraînement consiste essentiellement à diviser la base d'apprentissage en plusieurs sous-bases et à entraîner chaque expert avec son propre ensemble de prototypes. En pratique, les bases de données sont de tailles relativement restreintes. Cela implique que pour avoir un nombre de données acceptable au sein de chacune des sous-bases d'apprentissage, leur intersection ne pourra être nulle. Par conséquent, il faut utiliser des méthodes de sélection des prototypes pour former les sous-bases d'apprentissage. Parmi les plus courantes de ces méthodes, on retrouve le Bagging [8] et le Adaboost [23].

Le Bagging, la méthode la plus simple, est l'équivalent statistique d'un sondage dynamique avec remise des échantillons tirés. On tire un prototype, on le remet dans la base de données, et on réitère jusqu'à ce que toutes les bases aient le nombre de prototypes d'apprentissage voulu. Le Adaboost est un algorithme itératif un peu plus sophistiqué où la probabilité de tirage des échantillons n'est pas la même pour tous (cf algorithme 2). À

chaque itération, on augmente la probabilité d'être tiré des échantillons mal classés. De cette façon, les échantillons difficiles à classer sont mieux représentés et ont plus de poids lors de l'apprentissage.

Algorithme 2 : Algorithme Adaboost [23]

Données :
D : l'ensemble des données et de leur étiquette notés respectivement X^i et Y_i
$W_k(i)$:la k^e distribution pour l'échantillonnage des prototypes d'apprentissage
Z_k : une constante de normalisation calculée de façon à ce que $W_k(i)$ demeure une distribution
$h_k(X^i)$: l'hypothèse, ou la décision prise par le classifieur C_k sachant qu'il a observé l'entrée X^i

début
  $k \longleftarrow 0$
  répéter
    $k \longleftarrow k+1$
    entraîner C_k en utilisant D échantillonné selon $W_k(i)$
    $E_k \longleftarrow$ erreur d'entraînement de C_k mesuré sur D grâce à $W_k(i)$
    $a_k \longleftarrow \frac{1}{2} \cdot \ln(\frac{1-E_k}{E_k})$
    $W_k(i) \longleftarrow \frac{W_k(i)}{Z_k} \cdot \begin{cases} e^{-a_k} & \text{si } h_k(x^i) = y_i \\ e^{a_k} & \text{si } h_k(x^i) \neq y_i \end{cases}$
  jusqu'à $k = k_{max}$
fin
retourner C_k *et* a_k *pour* $k = \{1, ..., k_{max}\}$

Ces techniques de création d'ensembles sont connues pour bien fonctionner avec des algorithmes tels les réseaux neuronaux et les arbres de décision. Les k-**NN** se sont cependant montrés stables à ce type de modifications et ne sont pas de bons candidats pour constituer des ensembles basés sur ces méthodes.

2.2.2 Modifier l'espace de représentation

Certains classifieurs, par exemple le k-**NN**, sont stables aux données d'entraînement. C'est à dire qu'entraîner plusieurs k-**NN** avec des sous-bases d'apprentissage différentes ne crée pas la différence requise à son utilisation en comité d'experts. Par contre, le k-**NN** est instable à la représentation des données. En modifiant la façon de représenter les données, on

pourra créer des *k*-**NN** diversifiés utilisables dans un ensemble de classifieurs. Une donnée à classer est représentée par un vecteur de valeurs numériques où chaque dimension est aussi appelée caractéristique. On imagine alors que ce vecteur représente un point dans un espace de dimension R. En faisant la projection de ce point dans un sous-espace de dimension $D|D < R$, on en obtient une nouvelle représentation. L'algorithme 3 détaille le fonctionnement d'un ensemble de *k*-**NN** projetés dans des sous-espaces aléatoires[1].

Algorithme 3 : Classification à l'aide d'un ensemble de *k*-**NN** projetés dans des sous-espaces aléatoires [33]

Données :
O : l'observation à classer
D : l'ensemble des prototypes et de leur étiquette notés respectivement X^i et Y_i
k : le nombre de plus proches voisins à considérer
N : le nombre de classifieur à utiliser
f : la dimension des sous-espaces
$knn(O, D, k)$: classifieur de type Plus Proche Voisin (voir algorithme 1)
RS : ensemble de N sous-espaces de dimension f ; chacun des sous-espaces est construit en sélectionnant de manière aléatoire f caractéristiques parmi celles disponibles
$Vote(Liste)$: une fonction de vote qui retourne la classe la plus représentée dans une liste et qui départage arbitrairement les cas d'égalité
début
  $i \longleftarrow 0$
  répéter
    $i \longleftarrow i + 1$
    $D_p \longleftarrow$ Projection de D dans RS_i
    $O_p \longleftarrow$ Projection de O dans RS_i
    $L_i \longleftarrow$ Résultat de $knn(O_p, D_p, k)$
  jusqu'à $i = N$
fin
retourner $Vote(L)$

Ho avec ses *Random subspace* [33, 30], Bay avec ses *Multiple Feature Subsets* [4, 3] et Itqon avec les *Feature Combinations* [36] exploitent, sous différents vocables, ce concept qui consiste à créer des ensembles d'experts ayant chacun une vue partielle et différente du même problème en échantillonnant de manière aléatoire des sous-ensembles de caractéristiques pour chaque classifieur de l'ensemble. Des méthodes plus sophistiquées où le

[1] La complexité computationnelle de cet algorithme est exposée à l'annexe J

choix des caractéristiques n'est pas complètement aléatoire mais cherche à maximiser leur utilité ont aussi été proposées, notamment par Cunningham avec un algorithme de type escalade [15], Guerra-Salcedo avec une recherche par algorithmes génétiques [28] et Tumer avec ce qu'il appelle *Input Decimation* où il tient compte de la corrélation entre chacune des caractéristiques [73].

Pour que les ensembles généralisent bien dans les sous-espaces aléatoires (SEA), il faut un grand nombre de caractéristiques d'entrées. Cunningham parle d'un nombre de discriminants supérieur à 35 [14]. En effet, en choisissant de façon aléatoire un sous-ensemble de caractéristiques parmi un vaste ensemble, on obtient une plus grande diversité de combinaisons que si la pige a lieu au sein d'un ensemble restreint. Pour des problèmes ayant un petit nombre de caractéristiques, Ho suggère d'augmenter l'espace de départ en opérant quelques manipulations simples sur les discriminants initiaux (opérations arithmétiques, booléennes, etc.), ce qui a aussi pour effet d'ajouter de la redondance dans les caractéristiques [33].

2.2.3 La diversité

Pour être utiles, les classifieurs d'un **EoC** doivent émettre des opinions différentes les unes des autres, plus particulièrement sur les cas difficiles à classer et susceptibles d'être en erreur. Ils doivent aussi maintenir une certaine performance individuelle. On imagine mal comment un ensemble de classifieurs dont les experts ne seraient pas plus efficace que le hasard pourrait être d'une quelconque utilité. Cette situation crée le paradoxe suivant : plus les experts sont performants, plus ils sont semblables et plus les experts sont semblables moins ils sont utiles à l'ensemble. À la limite, si tous les classifieurs sont des classifieurs omniscients (qui ont un taux de reconnaissance de 100% quelque soit les données d'entrées), ils seront nécessairement identiques ; il serait dès lors inutile d'avoir un comité d'expert : un classifieur unique suffirait.

En pratique, il n'y a pas de classifieurs ayant des taux de reconnaissance de 100% sur des problèmes non triviaux (reconnaissance d'écriture ou de visages, diagnostic médical, etc.). Ce paradoxe met toutefois en lumière la chose suivante, à savoir que la différence recherchée est à la fois nécessaire et difficile à exprimer. Il faut des experts performants mais avec une certaine diversité d'opinion, donc qui se trompent. La clef est qu'ils doivent faire des erreurs sur des exemples différents [8, 23].

Le théorème du Jury de Condorcet parle d'indépendance statistique des experts, ce qui est une condition trop forte dans le cas qui nous occupe. Nous savons néanmoins que le taux de reconnaissance doit augmenter avec le nombre d'experts tant que la diversité au sein de ces derniers croît elle aussi [14, 57]. Le problème réside dans la définition formelle de cette diversité. Si les hypothèses produites ne sont pas statistiquement indépendantes, il faut à tout le moins que les erreurs ne soient pas complètement corrélées. Mais est-ce que la corrélation des erreurs est bien la mesure qui convient ?

Un grand nombre de mesures a été proposé et étudié. Les travaux de Kuncheva et ceux de Ruta en dressent d'ailleurs la liste des plus importantes [49, 65], dont plusieurs sont présentées à l'annexe B. Leur principes de bases sont présentés à la section suivante.

Mesure de la diversité

La difficulté d'exprimer le concept de diversité d'opinion au sein des classifieurs a fait en sorte que de nombreux auteurs en ont donné des définitions différentes. Ces définitions sont basées sur deux couples de stratégies qui s'opposent : mesures *par paire* versus mesures *globale* et mesures *avec oracle* versus mesures *sans oracle*.

- **Mesure par paire versus globale** - Les mesures par paire sont des mesures où sont comparées deux à deux toutes les sorties de tous les classifieurs de l'ensemble. La mesure de $Q_{average}$ [49] présentée en annexe est une mesure par paire. Une mesure

globale est une mesure prise sur l'ensemble des sorties des classifieurs. Les différentes définitions de l'entropie [14, 49] sont des exemples de mesures globales.

- **Mesure avec orale versus sans oracle** - En reconnaissance de formes, l'oracle est un classifieur omniscient que nous pouvons simuler en utilisant une base de données étiquetée. Ce classifieur a, par définition, un taux de reconnaissance de 100%. On se sert d'un oracle dans plusieurs algorithmes d'apprentissage, par exemple pour faire une recherche par minimisation du gradient de l'erreur, cette dernière étant la différence entre la sortie attendue (prédite par l'oracle) et la sortie réelle. Certaines mesures de diversité utilisent un oracle pour quantifier l'erreur des classifieurs. Ces mesures ont donc besoin d'un oracle ; c'est le cas de la mesure de *Fault Majority* [64]. On peut aussi mesurer la dispersion des opinions sans égard à ce qu'elles soient vraies ou fausses. L'*ambiguïté* [14] est ainsi une mesure globale sans oracle.

Il va sans dire que l'oracle ne peut être utilisé qu'au cours de l'entraînement avec des données étiquetées. Les mesures avec oracle ne peuvent donc pas être utilisée sur un **EoC** en production.

Discussion autour de la diversité

Toutes les mesures de diversité proposées cherchent à traduire la différence des classifieurs pour permettre de construire des ensembles performants. Par exemple, Cunningham a tenté, avec un certain succès, de créer des ensembles de *k*-**NN** avec une méthode de type escalade en maximisant, en plus du taux de reconnaissance, une mesure de diversité appelée ambiguïté [15] (cf algorithme 4) et Opitz a fait la même expérience en utilisant un algorithme génétique simple et des ensembles de MLP [60]. Dans les deux cas, la diversité était créée en sélectionnant des sous-ensembles de caractéristiques pour entraîner les différents classifieurs. De son côté, Ho a proposé la méthode des sous-espaces aléatoires pour des *k*-**NN** en avançant, sans le démontrer, que le caractère stochastique du choix des caractéristiques créait une diversité intrinsèque et que, par conséquent, une mesure explicite de

Algorithme 4 : Génération d'ensemble en utilisant une méthode de type escalade et la diversité [15]

Données :
masque : vecteur de valeurs booléennes indiquant si une caractéristique doit être ou non utilisée.

début
  Créer un ensemble de sous-espaces aléatoire
  répéter
    pour *tous les classifieurs i de l'ensemble* **faire**
      Calculer l'erreur initiale E_i et la contribution à l'ambiguïté A_i
      pour *tous les bits j du masque* **faire**
        Inverser le j^e bit du i^e masque
        Calculer les nouvelles E'_i et A'_i ;
        si $(E'_i \leq Seuil \times E_i$ ***ET*** $A'_i > A_i)$ ***OU*** $(E'_i < E_i$ ***ET*** $A'_i \geq Seuil \times A_i)$
        alors
          // modification acceptée
          $E_i \longleftarrow E'_i$
          $A_i \longleftarrow A'_i$
        sinon
          // modification rejetée
          inverser de nouveau le j^e bit du i^e masque
  jusqu'à *ce qu'il n'y ait plus de changement dans les masques* ***OU*** *que le nombre maximal d'itérations ait été atteint*
  Calculer la performance de l'ensemble
fin

diversité n'était pas nécessaire [33]. Les études exhaustives de Kuncheva sur la diversité n'ont pas pu déterminer si une mesure est supérieure à une autre pour créer des ensembles performants, mais semblent indiquer que la diversité est tout de même utile [50, 49, 48]. En reprenant ces expériences et en les poussant plus loin, Ruta trouve plutôt que les mesures de diversité n'aident pas directement à créer des ensembles performants [66, 65]. Oliveira, dans un contexte bien précis d'optimisation d'ensembles à l'aide d'algorithmes génétiques multicritères, semble pour sa part avoir trouvé que la diversité peut avoir un effet bénéfique indirect en guidant mieux la recherche [55].

Bien que la diversité -au sens de la divergence d'opinion- soit une propriété nécessaire des ensembles performants, l'utilité d'une mesure explicite de celle-ci pour la création d'**EoC** demeure un sujet de débat.

2.2.4 La sélection de classifieurs

La construction d'un ensemble de classifieurs demande la création au préalable de plusieurs classifieurs individuels pouvant être obtenus par les différentes stratégies énoncées plus haut. On peut ensuite bâtir l'**EoC** par simple agrégation des experts ou en choisissant un sous-ensemble optimal de ces derniers. La recherche d'un tel sous-ensemble, maximisant par exemple le taux de reconnaissance, s'appelle la sélection de classifieurs. La particularité de cette recherche est qu'elle ne s'intéresse pas nécessairement aux qualités individuelles des éléments de l'ensemble mais plutôt aux qualités globales de l'**EoC** qui découlent des combinaisons de ses classifieurs. La sélection de classifieurs permet de diminuer la taille d'un **EoC** tout en conservant, voire en améliorant, ses taux de reconnaissance [58].

2.2.5 Fusion et post-traitement

Lorsque tous les experts ont pris une décision, il faut rendre un verdict unique, basé sur l'avis de tous les classifieurs. Cette étape du post-traitement est souvent appelée fusion et nous appelons *fonction de fusion* la stratégie utilisée à cette fin. Plusieurs fonctions de fusion sont possibles et ont été étudiées [6, 42, 41] ; en voici une courte liste :

1. **Vote à majorité simple** - C'est la fonction de fusion la plus simple et probablement la plus utilisée. C'est exactement l'équivalent d'un vote uninominal à un tour. La sortie de chaque expert est considérée comme étant un vote pour une classe. Le nombre de votes pour chacune des classes est compté et l'ensemble choisit la classe en ayant remporté le plus.

2. **Sommation** - La sortie de chaque classifieur doit être un vecteur de taux de confiance, ou une probabilité *a posteriori* (à chaque classe correspondant un taux de confiance). La fusion est obtenue en sommant les vecteurs de sortie de tous les experts, puis en sélectionnant la classe ayant la valeur la plus élevée.
3. **Règle du produit** - Comme la règle de sommation, l'utilisation de cette règle nécessite des sorties de classifieurs représentant un vecteur de taux de confiance. La fusion est obtenue en faisant le produit de ces taux pour chacune des classes, puis en sélectionnant la classe ayant la valeur la plus élevée. Cette méthode a le désavantage de produire des valeurs nulles dès qu'un classifieur produit une probabilité *a posteriori* nulle pour une classe donnée.

Ces règles sont à la base d'un grand nombre de fonctions de fusion plus sophistiquées. Cependant, Kittler [42] a montré qu'il n'y avait pas une méthode meilleure que les autres dans l'absolu et que le choix de la fonction de fusion devait dépendre du type de classifieurs du comité ainsi que de l'espace des caractéristiques d'entrée. Il est aussi possible de se servir de toutes les opinions comme entrées d'un nouveau classifieur. Mais on peut, comme Ho [32], se demander quelles informations supplémentaires contiendraient ces nouvelles entrées par rapport au vecteur de caractéristiques original.

On remarquera que le vote à majorité simple ne nécessite pas un vecteur de probabilité *a posteriori* mais seulement l'étiquette d'une classe. C'est là un avantage qui facilite la fusion de classifieurs qui ne produisent pas «naturellement» de telles sorties (le *k*-**NN** en est un exemple). En plus de sa grande simplicité, cette fonction de fusion est connue pour être efficace [64, 42], d'où sa popularité.

2.3 Estimation des probabilités *a posteriori* et rejet

L'utilisation d'un classifieur ou d'un ensemble de classifieurs comme estimateur de probabilité *a posteriori* d'appartenance d'un objet à une classe est une chose courante. Dans ce cas, le système de classification ne fournit pas en sortie l'étiquette d'une classe, contrai-

rement au *k*-**NN**, mais plutôt une probabilité d'appartenance pour chacune des classes. Certain réseaux neuronaux sont un exemple de classifieur pouvant fournir de telles sorties [39]. Ce type d'estimation a l'avantage de situer le problème dans un contexte probabiliste qui a de solides bases théoriques. On peut penser à la théorie bayesienne mais aussi aux techniques de régression pour estimer l'erreur, le biais et la variance [20] et même la diversité [44]. De plus, l'estimation de $P(C_i|x)$ devient souvent incontournable dans le cas où un système doit assurer un mécanisme de rejet. En effet, une grande partie des mécanismes de rejet est basée sur les probabilités *a posteriori*.

2.3.1 Stratégies pour le rejet

Il est souvent utile de prévoir un mécanisme de rejet à un système de reconnaissance de formes pour en assurer un haut taux de fiabilité. En connaissant les probabilités *a posteriori* d'appartenance d'un objet aux classes possibles, on choisira naturellement la classe ayant la probabilité la plus élevée. Mais si cette probabilité maximale n'est pas assez élevée, on pourra décider de rejeter l'objet et de ne pas le classer. C'est la méthode proposée par Chow [12]. Cette stratégie n'utilise qu'un seuil indépendamment de la classe choisie. Dans les faits, on sait que certaines classes sont généralement plus facilement reconnaissables que d'autres. Fumera a ainsi proposé une méthode de rejet où il y a un seuil différent pour chacune des classes du système de façon à exiger une probabilité plus élevée pour les classes «faciles» et ainsi maximiser la fiabilité du classifieur. Pour trouver ces seuils, un algorithme itératif sous-optimal a été proposé par Fumera [24].

2.3.2 Estimation des probabilités *a posteriori*

Dans le cas d'ensembles de classifieurs à vote, comme les ensembles de *k*-**NN**, la difficulté est de transformer un ensemble d'étiquettes en une estimation de la probabilité *a posteriori*. Il y a deux méthodes courantes pour y arriver, une première basée sur les matrices de confusion proposée par Xu [76] et une seconde, utilisée pas Hansen [29], qui se calcule comme une simple moyenne. Les deux auteurs ont rapporté obtenir des résultats

satisfaisants mais en pratique, la méthode de Hansen est beaucoup plus simple à implanter et demande moins d'effort de calcul comme le montrent les équations de l'annexe E.

2.4 Optimisation d'ensembles de classifieurs

La création d'**EoC** nécessite la génération de classifieurs individuels différents les uns des autres. Ces classifieurs ne doivent pas nécessairement être le plus performant possible mais doivent, lorsque utilisés en comité, offrir de meilleures performances que le plus performant des classifieurs de l'ensemble. Souvent, la génération d'un grand nombre de classifieurs suffit à atteindre cet objectif. Ajouter des classifieurs à un ensemble jusqu'à l'obtention d'un certain seuil de performance est donc une façon de créer des **EoC**. Par contre, cette méthode a le désavantage d'être sous-optimale. En effet, rien ne garantit l'obtention d'un seuil donné. De plus, certains classifieurs n'aident pas à améliorer la performance de l'ensemble. Au mieux ils ajoutent à la complexité du système. Au pire, et c'est souvent le cas, ils en diminuent la performance. Aucun classifieur individuel n'est directement responsable de cette situation et c'est l'agencement de ces derniers qui en est la cause. C'est là tout le problème de la création des **EoC** car on ne peut prédire complètement la performance d'un ensemble à partir de la seule étude des individus le composant.

Les sections suivantes expliquent quelques stratégies pour optimiser la performance d'un ensemble. L'objectif est de maximiser le taux de reconnaissance de celui-ci. En pratique, on voudra aussi en limiter la complexité. Cela peut être fait soit en tentant de minimiser le nombre de classifieurs, soit en en fixant *a priori* le nombre de manière arbitraire. Finalement, puisque la diversité doit jouer un certain rôle, la maximisation de cette dernière peut aussi parfois être utilisée comme objectif indirect.

2.4.1 Méthodes de recherche

Le problème de recherche d'ensembles de classifieurs peut être vu selon deux angles. Il s'agit d'une part de combiner des classifieurs qui, à l'aide d'une certaine fonction de fu-

sion, auront le plus haut taux de reconnaissance possible. D'autre part, dans le contexte où les classifieurs sont générés par échantillonnage des caractéristiques, il s'agit de sélectionner des ensembles de caractéristiques maximisant le taux de reconnaissance. Dans les deux cas, l'objectif est le même mais l'angle d'approche diffère. Pour cette raison, nous présentons ici différentes approches visant à faire de la sélection de caractéristiques et/ou de la combinaison de classifieurs.

La méthode de type escalade mentionnée en 2.2.3 est une première approche pour créer des **EoC** qui tient compte de la diversité. L'algorithme mesure le taux de reconnaissance avec et sans l'utilisation d'une caractéristique pour un classifieur. On garde la modification si elle diminue l'erreur sans trop diminuer l'ambiguïté ou si elle augmente l'ambiguïté sans trop augmenter l'erreur. Toutes les caractéristiques de tous les classifieurs sont ainsi testées en boucle jusqu'à ce qu'à ce qu'une condition d'arrêt soit atteinte (voir algorithme 4).

L'utilisation d'algorithmes génétiques est appropriée pour l'optimisation de systèmes de reconnaissance de formes puisque l'espace de recherche est très vaste et mal défini. Par exemple, Guerra-Salcedo a utilisé un AG pour faire de la sélection de caractéristiques [28]. De son côté, Kuncheva s'en est servi pour construire des systèmes de fusion d'**EoC** [45].

Pour la création d'ensembles de classifieurs, Opitz utilise aussi l'erreur et l'ambiguïté comme critère de sélection de caractéristiques [60]. Il utilise comme fonction d'adaptation une combinaison du taux de reconnaissance et de la diversité, avec un facteur ω tel que $0 \leq \omega \leq 1$ pour pondérer l'importance accordée à l'un et à l'autre critère.

$$Adaptation = (1 - \omega) \cdot Reconnaissance + \omega \cdot Ambiguïté \tag{2.1}$$

Plus généralement, la façon d'optimiser plusieurs objectifs contradictoires avec un algorithme génétique simple est de les ramener à un seul objectif. Par exemple, pour minimiser $\{f_1(x), f_2(x), \ldots, f_N(x)\}$, les algorithmes classiques combineront tous les objectifs en une seule fonction $F(x)$ qui sera la somme pondérée de ces derniers (équation 2.2).

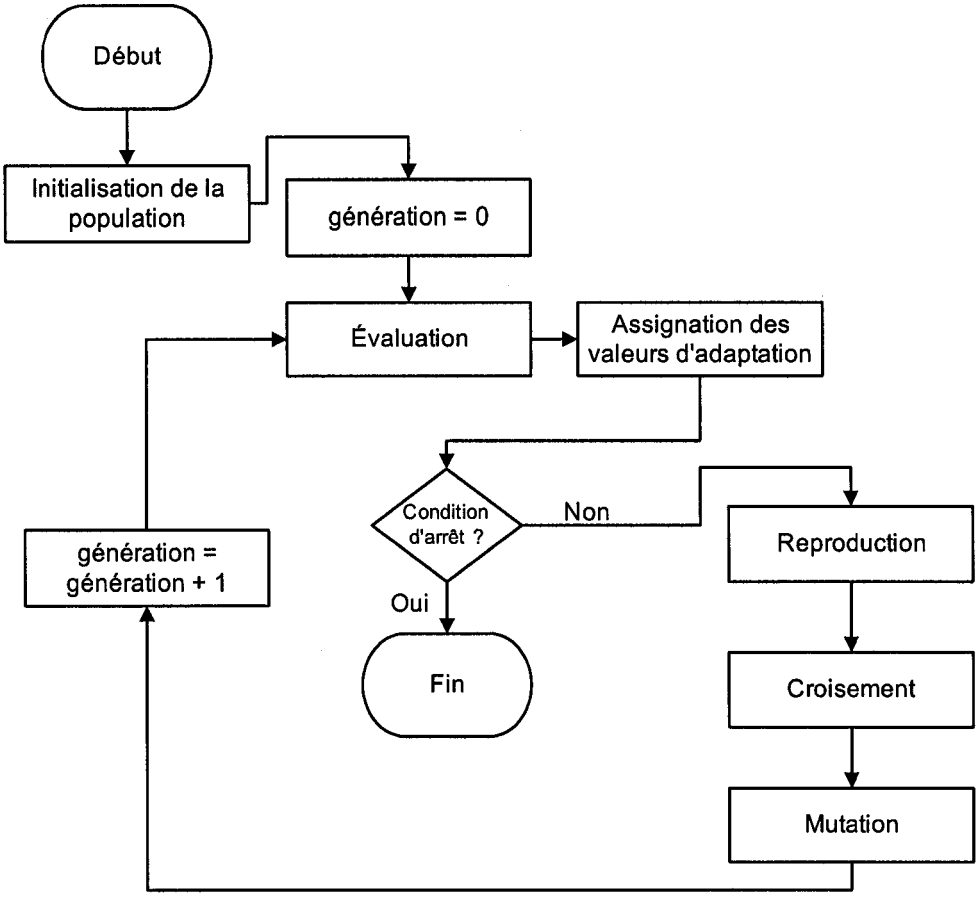


Figure 9 **Organigramme d'un algorithme génétique simple [17].**

$$F(x) = \sum_{i=1}^{N} \omega_i f_i(x) \tag{2.2}$$

où

$x \in X$, X étant l'espace des objectifs

et les poids ω_i sont définis tels que :

$$0 \leq \omega_i \leq 1$$

$$\sum_{i=1}^{N} \omega_i = 1$$

La combinaison d'objectifs dans la fonction d'adaptation d'un AG simple rend difficile la recherche car un objectif a toujours tendance à prendre le dessus sur les autres [17, 58]. Pour cette raison, Oliveira a utilisé des AG multi-objectifs (NSGA [68]) pour créer des ensembles de réseaux de neurones en maximisant conjointement le taux de reconnaissance et l'ambiguïté [58, 55, 59]. Dans un premier temps, un algorithme de type NSGA est utilisé pour créer des classifieurs en minimisant conjointement l'erreur et le nombre de caractéristiques ; la particularité des AG multi-objectifs est de fournir non pas une solution mais un ensemble de solutions (front de Pareto). À partir de l'ensemble de classifieurs formé par le front de Pareto, une deuxième recherche utilisant NSGA est effectuée, cette fois-ci pour trouver des combinaisons de classifieurs qui maximisent le taux de reconnaissance et l'ambiguïté.

2.4.2 Sélection de caractéristiques

Puisque la création d'ensembles de classifieurs fait aussi appel à la sélection de caractéristiques, il est bon de distinguer deux stratégies utilisées pour cette sélection (cf figure 10). L'approche par front de Pareto, utilisée par Oliveira crée des classifieurs en optimisant conjointement deux objectifs : maximiser la performance et minimiser le nombre de caractéristiques [59]. L'ensemble résultant est constitué du front de Pareto obtenu où chaque classifieur utilise un nombre différent de caractéristiques. La seconde approche, celle des sous-espaces aléatoires, fixe *a priori* le nombre de caractéristiques des classifieurs et sélectionne aléatoirement ces dernières pour chacun d'entre-eux. Ces deux méthodes produisent un ensemble qui peut ensuite être optimisé par une méthode de recherche quelconque.

Hypothèses de recherche

Il devrait être possible de diminuer la complexité d'un **EoC** de *k*-**NN** par sélection de classifieurs tout en améliorant sa performance. L'utilisation d'un algorithme d'optimisation pouvant explorer de grands espaces mal définis est nécessaire pour atteindre cet objectif.

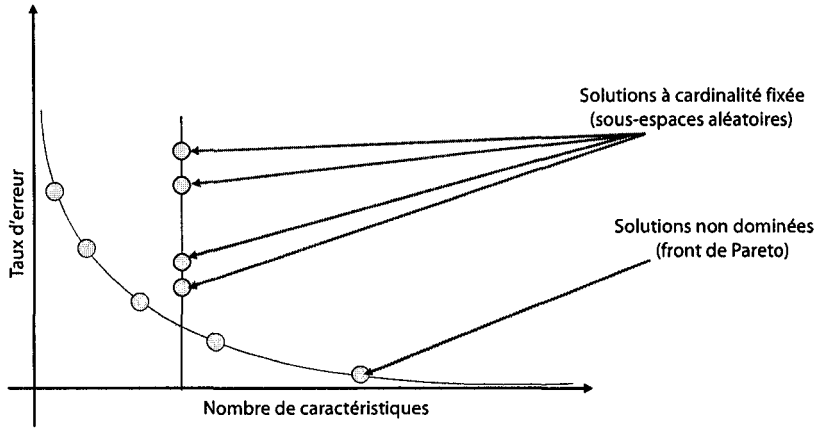


Figure 10 **Deux approches pour la sélection de caractéristiques.** Un front de Pareto fournit un ensemble de solutions dont le taux d'erreur diminue en fonction du nombre de caractéristiques. Avec la méthode des sous-espaces aléatoires, le nombre de caractéristiques est fixé *a priori* et les différentes solutions ont des taux d'erreur aléatoirement distribués.

Une recherche aléatoire ou un algorithme génétique simple devrait y parvenir. Par contre, l'emploi d'un algorithme d'optimisation multi-objectifs devrait être plus intéressante puisqu'il sera possible de minimiser directement la complexité du système en plus de maximiser sa performance. De plus, Oliveira a montré que la maximisation explicite d'une mesure de diversité -l'ambiguïté- en plus de la maximisation de la performance donne d'excellents résultats dans le contexte de sélection de MLP [58]. Cela devrait aussi être le cas dans le contexte de sélection d'ensembles de *k*-**NN** projetés dans des sous-espaces aléatoires. Nous ne savons pas quelle est la meilleure mesure de diversité à utiliser dans notre contexte mais au moins une des celles parmi les plus populaire de la littérature récente devrait aider significativement à la recherche d'ensembles performants.

Le chapitre suivant expose le matériel et les méthodes employées pour vérifier nos hypothèses de recherche. Une première partie est consacrée au matériel en tant que tel (logiciel, équipement informatique et base de données) alors que les parties suivantes développent les stratégies pour tester différentes méthodes de sélection de classifieurs et pour les comparer entre-elles.

CHAPITRE 3

MATÉRIEL ET MÉTHODES

Ce chapitre explique les expériences réalisées pour vérifier les hypothèses de recherche présentées précédemment. Dans une première partie, une description est faite du matériel utilisé (partie logicielle, équipement informatique et description de la base de données). Dans les parties suivantes, ce sont les méthodes employées pour faire les choix des paramètres des expériences ainsi que les techniques de recherche d'ensembles de classifieurs qui sont présentées.

3.1 Matériel

Cette section décrit ce qui a été déployé pour effectuer les expériences, c'est-à-dire la partie logicielle, l'équipement informatique ainsi que la base de données de chiffres manuscrits.

3.1.1 Description de la partie logicielle

L'ensemble des expériences et simulations ont été programmées en *C++* et compilées sur le système d'exploitation *Linux*. Un ensemble de commandes, parfois compilées et parfois faites en *Bourne shell*, a servi à contrôler les programmes (gestion des paramètres, et des itérations) et à extraire l'information des simulations effectuées, la sortie de tous les programmes étant sous forme de fichier texte parfois très volumineux.

Le coeur des programmes consiste en une hiérarchie commune de classes interdépendantes à différents degrés. On y trouve par exemple les différentes mesures de diversités, de distance, les règles de classification par *k*-**NN**, etc.

L'ensemble des classes et des programmes représente environ 10 000 lignes de code en *C++* et environ 1000 lignes de code en différents langages de script (Bourne shell, awk, etc.).

3.1.2 Précalcul des votes

La partie la plus importante du temps de calcul des *k*-**NN** consiste à mesurer les distances entre l'observation à classer et les prototypes de la base d'apprentissage. Puisque, pour l'ensemble du projet, la méthode de fusion des classifieurs est le vote majoritaire simple, les distances ne servent qu'à déterminer le plus proche voisin (on n'utilise pas la distance pour établir un taux de confiance, ou estimer une probabilité *a posteriori*). L'information utile est donc le vote d'un classifieur pour un exemple à classer. Par conséquent, il a été décidé de créer de nouvelles bases de données où les votes des classifieurs seraient déjà calculés. De cette façon, le calcul des performances d'un **EoC** se résume à la combinaison des votes de ses participants. Cette opération est indépendante de la taille de la base d'apprentissage et est extrêmement rapide comparativement aux opérations nécessaires pour le calcul des distances. Les nouvelles bases de données sont donc constituées du vote de chaque classifieur, pour chacune des observations à classer.

Les votes, et par conséquent le contenu de ces bases de données, dépendent à la fois des sous-espaces aléatoires (*SEA*), de la base d'apprentissage et de la base des observations à classer. Les *SEA* ont été générés aléatoirement et sont au nombre de 100 (le même nombre que Ho a choisi dans la présentation originale de la méthode des *SEA* appliquée aux *k*-**NN** [33]). Les différentes bases de données de votes sont donc une fonction de deux bases de données (apprentissage et généralisation). Le tableau XVI de l'annexe C présente les couples utilisés pour créer les bases de données de votes précalculés.

3.1.3 Description de l'équipement informatique

Les expériences ont été menées sur deux types d'architecture en fonction de besoins spécifiques. La majeure partie des simulations ont été faites sur des serveurs de calculs de type *Athlon* 2 GHz équipés de 2 giga-octets de mémoire vive. L'accès à trois serveurs bi-processeurs de ce type ayant permis d'exécuter simultanément plusieurs expériences indépendantes. La manipulation des bases de données, le précalcul des votes, les simu-

lations de caractéristiques absentes et la recherche stochastique d'**EoC** font partie des expériences faites sur cette architecture.

D'un autre côté, la recherche de solutions à l'aide d'algorithmes génétiques (mono et multi-objectifs) aurait été difficilement envisageable sans l'utilisation d'une architecture parallèle. En effet, chaque individu de la population génétique est un ensemble de classifieurs. Pour calculer la valeur d'adaptation d'un individu, il faut entraîner ce dernier (étape rapide dans le cas du *k*-**NN**) et prendre des mesures à l'aide d'une base de données dite, d'optimisation, comprenant 10 000 exemples. Ces mesures sont le taux de reconnaissance et différentes métriques de diversité plus ou moins complexes. Ainsi, toute la recherche à l'aide d'algorithmes génétiques a été effectuée sur une grappe[1] composée de 25 ordinateurs dans une configuration maître-esclaves. Dans cette configuration, une unité maître gère les opérations génétiques de la population et les conditions d'arrêt de l'algorithme. La valeur d'adaptation des nouveaux individus est évaluée simultanément par tous les esclaves qui transmettent le résultat au maître.

3.1.4 Bases de données

Les expériences ont été conduites avec une base de données composée de chiffres manuscrits isolés stockés sous forme d'images binaires (voir la figure 11 pour en avoir un aperçu). Cette base, appelée NIST SD19, est rendue disponible par l'organisme américain National Institute of Standards and Technology (NIST). Les chiffres manuscrits isolés présentent l'avantage d'être abondamment disponibles sous la forme de base de données étiquetées. Cette grande disponibilité permet d'ajuster le nombre d'exemples d'apprentissage selon les besoins tout en conservant de larges bases pour la validation et le test. De plus, le problème de la reconnaissance de chiffres manuscrits isolés est réputé comme étant difficile et a fait l'objet de nombreuses études, il est donc facile de comparer de nouveaux résultats à d'autres déjà publiés dans la littérature [58, 10, 35, 78].

[1] Le terme anglophone *cluster* est peut-être plus familier.

Figure 11 **Exemples de la base de données NIST.** Reproduction de 10 échantillons de chaque classe de la base NIST utilisée dans le projet.

Les chiffres manuscrits isolés et les caractéristiques extraites

Au moment de faire le projet, les caractéristiques ayant permis l'obtention des plus hauts taux de reconnaissance avec la base NIST étaient celles publiées par Oliveira appelées «Concavities and Contours» [59]. Ce sont elles que nous avons utilisées et elles sont brièvement présentées ici. Pour plus de détails, on pourra se référer à la référence originale.

Ces caractéristiques se divisent principalement en deux types : la direction des contours et les concavités. Chaque caractère est d'abord séparé en six zones. Un histogramme des 8 directions de Freeman du contour est ensuite construit pour chaque zone (voir la fi-

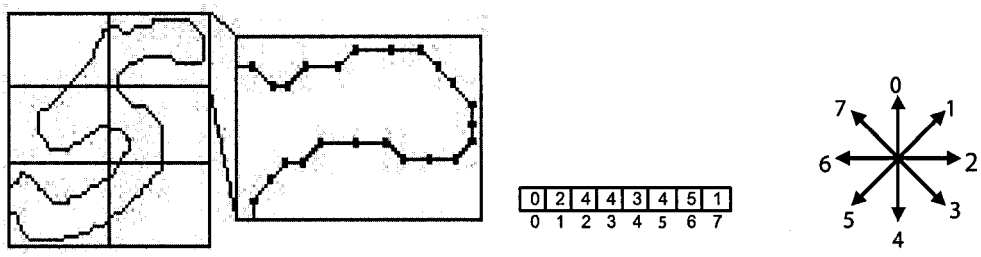


Figure 12 **Histogramme des directions du contour [58].** a) Division en six zones b) Histogramme des directions de Freeman c) Codage des 8 directions de Freeman ©*L. S. Oliveira. Reproduit avec l'autorisation de l'auteur.*

gure 12). De ces mêmes zones sont extraites les caractéristiques de concavité. Pour chaque pixel blanc de la zone, une recherche est effectuée dans les 4 directions de Freeman pour trouver un pixel noir (cf figure 13 a et d) et un premier histogramme est compilé avec le nombre de pixel noirs rencontrés (cf figure 13 b). S'il y en a 4, une nouvelle recherche est effectuée à partir de ceux-ci pour trouver une direction de sortie (cf figure 13 a et c) et les résultats sont reportés dans un nouvel histogramme. Seules 13 dimensions sont retenues dans le vecteur ainsi formé. La concaténation de ces 13 valeurs avec les 8 caractéristiques de contours, ajoutées à une dernière caractéristique de surface (nombre de pixels noirs), donne un vecteur de $13 + 8 + 1 = 22$ caractéristiques par zone, un caractère divisé en six zones donnant un vecteur de 132 caractéristiques.

Divisions de la base NIST SD19

Cette partie apporte des précisions sur l'utilisation de la base NIST. La division de cette denière est décrite dans le tableau I.

Toutes les classes sont parfaitement équiréparties au sein de la base ***HSF_0123***. Cette dernière sert à l'apprentissage des *k*-**NN** et a été sous-divisée en sous-bases tel qu'illustré dans le tableau II [72]. La base d'entraînement ***TRAIN50K*** est elle aussi divisée en sous-bases d'apprentissage pour former les bases décrites au tableau III. Ainsi ***TRAIN500*** est un sous-ensemble de ***TRAIN1K***, elle même un sous-ensemble de ***TRAIN5K***, etc. Nous

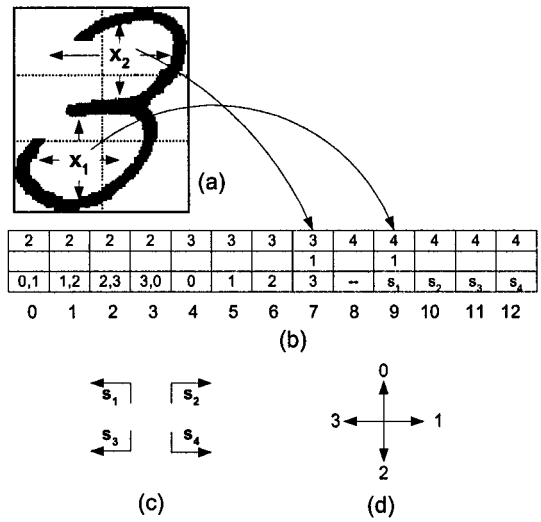


Figure 13 **Histogrammes des concavités [58].** a) Division en six zones b) Histogramme des contours et des directions de sortie c) Codage des directions de sortie d) Codage des 4 directions de Freeman ©*L. S. Oliveira. Reproduit avec l'autorisation de l'auteur.*

Tableau I

Répartition de la base NIST

HSF_0123	195 000 exemples
HSF_3	28 132 exemples
HSF_7	60 089 exemples
HSF_4	58 646 exemples

nous sommes aperçu expérimentalement que les taux de reconnaissances approchaient leur valeur maximale à partir du moment où les classifieurs étaient entraînés avec une base de 50 000 exemples et plus (cf figures 18 et17). Pour limiter le temps de calcul des expériences et en raison du peu de variation attendu dans les résultats, les bases de données ont été découpées en sous-bases de 50 000 exemples et moins. Le découpage a été conçu de manière à pouvoir mesurer l'effet de la taille de la base de données sur

différents paramètres en assurant un nombre de points raisonnable (petit pour limiter les expériences mais suffisamment grand pour tracer une courbe).

Tableau II

Division de la base ***HSF_0123***

Nom	Taille	Position
TRAIN50k	50 000	exemples 1 à 50 000
VAL1	10 000	exemples 50 001 à 60 000
VAL2	10 000	exemples 60 001 à 70 000
VAL3	10 000	exemples 70 001 à 80 000

Tableau III

Division de la base ***TRAIN50K***

Nom	Taille	Position
TRAIN50k	50 000	exemples 1 à 50 000
TRAIN25k	25 000	exemples 1 à 25 000
TRAIN10k	10 000	exemples 1 à 10 000
TRAIN5k	5 000	exemples 1 à 5 000
TRAIN1k	1000	exemples 1 à 1000
TRAIN500	500	exemples 1 à 500

Dans les expériences qui suivent, ***HSF_0123*** et ses sous-ensembles servent à l'apprentissage des *k*-**NN**. La base ***VAL1*** est utilisée pour faire la recherche dans les ensembles de *k*-**NN** et est aussi appelée base d'*optimisation*. Ce sont les résultats en reconnaissance mesurés sur cette base de données qui guident la recherche d'ensembles performants (quel que soit l'algorithme de recherche). ***VAL2*** est appelée base de *validation*. Les résultats mesurés sur cette base servent à choisir les meilleurs **EoC** parmi les solutions trouvées en apprentissage avec la base d'optimisation. Enfin, ***HSF_7*** sert de base de *test* pour l'ensemble des manipulations. ***VAL3***, ***HSF_3*** et ***HSF_4*** ne sont pas utilisées. Le tableau IV résume cette nomenclature.

Tableau IV

Nomenclature des bases de données utilisées pour faire la recherche d'**EoC**

Nom	Base correspondante
Apprentissage	***TRAIN5k***
Optimisation	***VAL1***
Validation	***VAL2***
Test	***HSF_7***

3.2 Choix des paramètres

Afin de fixer le protocole expérimental, un certain nombre de choix doit être fait. Ainsi les différentes métriques servant à mesurer les concepts à optimiser doivent être choisies, de même que les paramètres des *k*-**NN**. Cette section détaille et explique ces choix.

3.2.1 Mesures d'optimisation

Trois concepts sont utilisés pour optimiser les ensembles : la performance, la complexité et la diversité. La notion de performance fait appel au taux de reconnaissance ou au taux d'erreur. Maximiser la performance veut donc dire maximiser le taux de reconnaissance tel que défini à l'équation 1.1 (ou minimiser le taux d'erreur, ce qui est équivalent). La complexité à minimiser qui nous intéresse est le nombre de classifieurs faisant parti d'un ensemble, c'est-à-dire le cardinal de ce dernier (également appelée cardinalité).

Finalement, la diversité doit être maximisée mais elle est plus difficile à définir. Puisqu'il n'y a pas de consensus sur la définition de la meilleure mesure de diversité (ni même sur sa réelle utilité pour optimiser des **EoC**), plusieurs mesures, présentées ci-dessous, sont utilisées. La façon de les calculer ainsi que leurs références sont présentées à l'annexe B.

- **Ambiguïté** - Mesure de type *globale* sans oracle utilisée par Oliveira [58] dans un contexte d'optimisation à l'aide d'algorithmes génétiques multi-objectifs et par Cunnigham [14] dans une recherche de type escalade. Cette mesure a déjà montré

des résultats intéressants dans des contextes proches du nôtre (sélection de classifieurs, algorithmes génétiques multi-objectifs) et constitue donc de ce fait la mesure de diversité de base de notre expérimentation.

Entropie (TI) - Mesure de type *globale* sans oracle. Il s'agit de la définition classique de la mesure du désordre en théorie de l'information. Elle a été utilisée notamment par Tibshirani [69] et par Cunningham [14] comme mesure de diversité dans le contexte d'ensembles de classifieurs.

- **Entropie (EK)** - Mesure de type *globale* sans oracle, il ne s'agit pas de l'entropie au sens généralement admis en théorie de l'information mais d'une définition donnée par Kuncheva [49]. Cette mesure se retrouve très souvent parmi celles utilisées comme référence lors de la comparaison des différentes façons de mesurer la diversité. Cette mesure a l'avantage d'être sans oracle et peut donc être utilisée en phase de généralisation.
- ***Fault Majority* (FM)** - Cette mesure se calcule de manière complètement différente de celles présentées plus haut puisque c'est une mesure *par paire* et qu'elle nécessite un oracle. Cette différence permet de tester une métrique d'une autre famille et d'élargir la définition de diversité. Proposée récemment par Ruta [64], cette mesure a été mise au point pour le contexte spécifique des ensembles utilisant le vote à majorité simple comme fonction de fusion et constitue de ce fait une mesure importante de notre étude. On désigne par NFM (*Normalized Fault Majority*), la mesure FM normalisée par le nombre de classifieurs.
- $\mathbf{Q_{average}}$ - Mesure de type *par paire* avec oracle. Basée sur la statistique Q proposée au début du XX^e s. par Yule [77], cette mesure a été largement utilisée par Kuncheva [49] dans le contexte d'**EoC**. La valeur Q est comprise entre -1 et 1. Pour des classifieurs statistiquement indépendants, l'espérance de Q est 0. Les classifieurs qui ont tendance à reconnaître correctement les mêmes objets auront des valeurs Q positives et les classifieurs qui tendent à faire des erreurs sur des objets différents auront

des valeurs Q négatives. C'est justement sur les erreurs que la diversité d'opinion est nécessaire, aussi cette mesure est-elle à minimiser.

L'ambiguïté mesure le désaccord moyen entre les classifieurs, c'est donc une mesure de de conflit. Nous avons remarqué que celui-ci est souvent très faible, voire nul, lorsque l'**EoC** identifie correctement une classe et qu'au contraire, il est généralement maximal en cas d'erreur. On remarquera, que l'ambiguïté se calcule comme le complément d'une définition de $P(C_i|x)$ (voir annexes B et E). Maximiser l'ambiguïté correspond donc à minimiser la moyenne, de cette définition de $P(C_i|x)$. L'augmentation de l'ambiguïté devrait maximiser la diversité, effet recherché dans la création d'**EoC**. D'un autre côté, diminuer le désaccord entre les classifieurs pourrait maximiser le taux de reconnaissance puisque ça augmente la probabilité $P(C_i|x)$. Ces deux argumentations sont paradoxales, l'une favorisant la maximisation et l'autre la minimisation d'un même paramètre. Pour cette raison, les deux cas de figures ont été testés afin de les comparer.

Pour mieux comprendre les relations entre ces différentes mesures de diversité, de la complexité et de la performance, un ensemble de **EoC** a été étudié. Cet ensemble provient d'une sélection de classifieurs faite par algorithme génétique simple ou le seul objectif est de minimiser l'erreur (cf section 3.3.3). Tous les individus de chacune des 1000 générations ont été enregistrés et, après élimination des doublons (individus identiques au sein d'une même génération ou se répliquant sur plusieurs générations), nous avons obtenu 2992 **EoC** ayant des cardinalités allant de 33 à 100 classifieurs. Ni la cardinalité, ni aucune des mesures de diversité n'ayant servi de critère d'optimisation, une étude *a posteriori* de ces valeurs permet de déceler des relations naturelles entre celles-ci. Pour ce faire, une matrice de corrélation de toutes ces mesures est construite et décomposée avec une analyse en composantes principales (ACP). Les deux premiers facteurs sont utilisés pour visualiser les relations (cf figure 14).

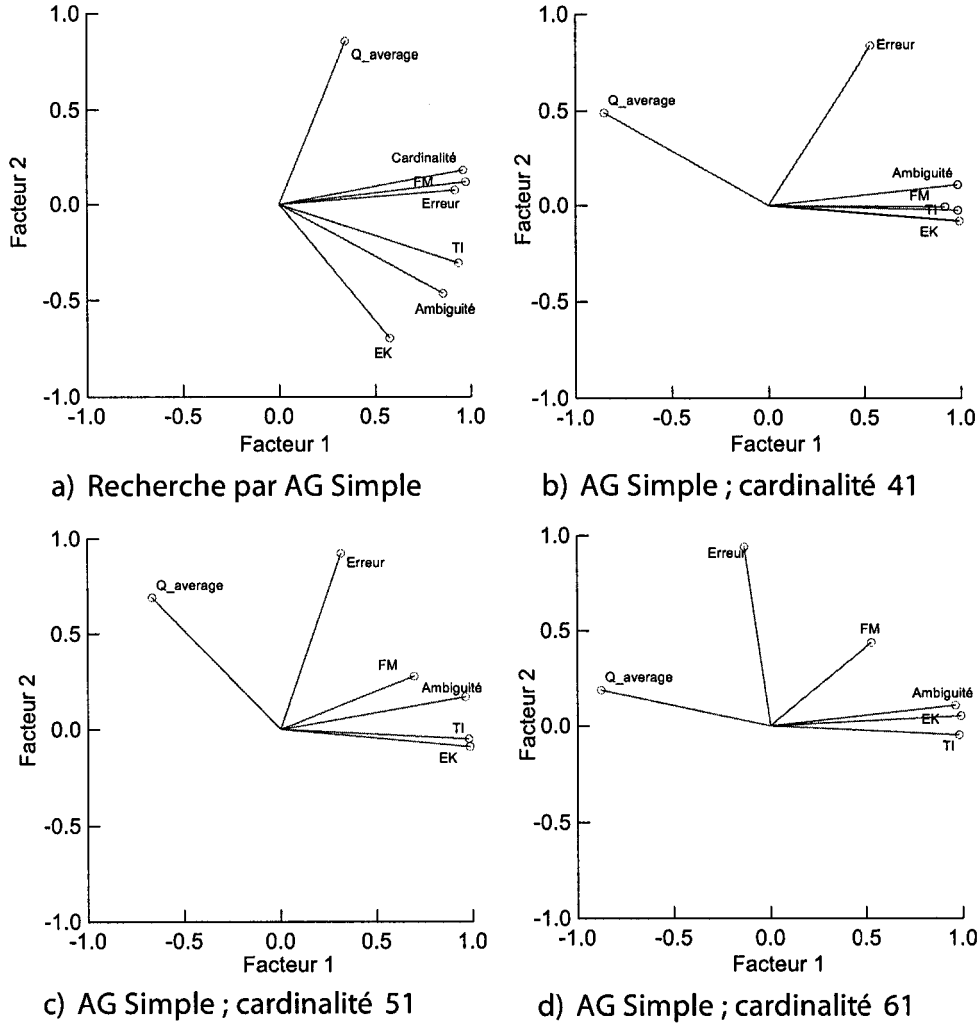
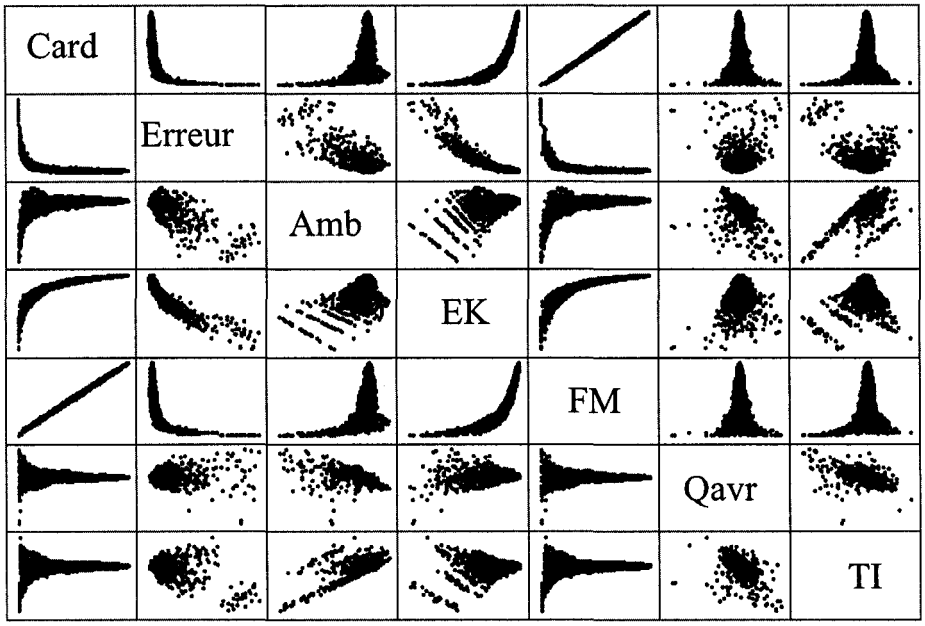


Figure 14 **Corrélation entre les mesures de diversité.** Sur ces figures sont projetées les décompositions (deux premiers facteurs) de la matrice de corrélation entre les différentes mesures de diversité. On retrouve en a) les résultats obtenus pour tous les ensembles obtenus avec AG Simple, alors qu'en b), c) et d), un échantillon de cardinalité fixe a été utilisé (cardinalité moyenne = 51 ; moyenne ± un écart-type = 41 et 61).

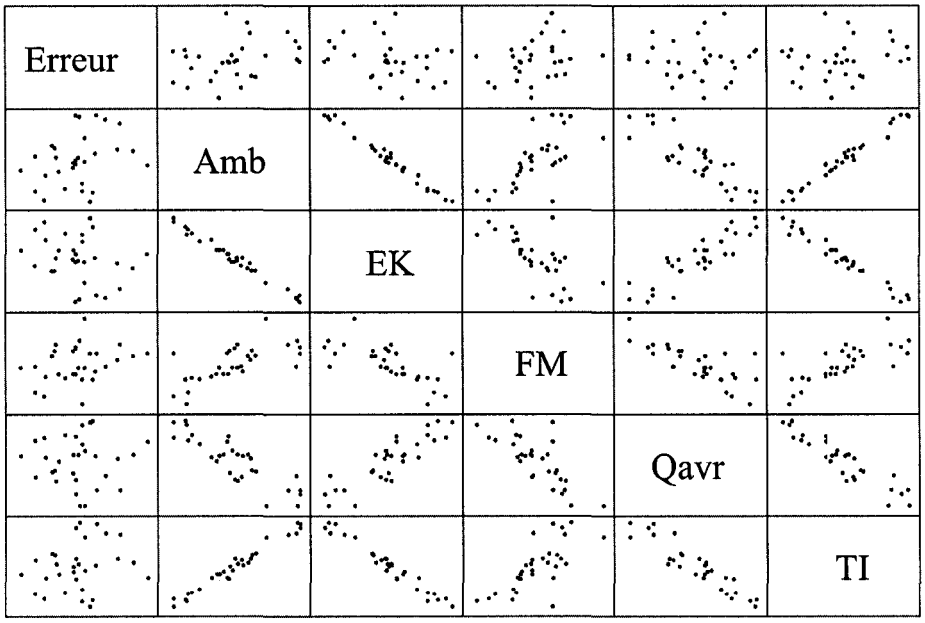
Sur la figure 14-a) on peut voir les relations globales entre la performance, la cardinalité et les mesures de diversité. Sur les autres figures (14-b) à 14-d)), la cardinalité a été fixée de sorte que ce paramètre n'influe pas sur les autres. Lorsqu'on considère les ensembles

de cardinalité moyenne (51), TI et EK sont très fortement corrélés ce qui est aussi vrai pour les deux autres cardinalités testées (41 et 61). Pour cette raison, il ne semble pas pertinent d'utiliser ces deux mesures aussi nous n'avons conservé que la définition de Kuncheva de l'entropie. On remarque aussi que FM est fortement lié à la cardinalité de l'ensemble (ce qui est tout à fait conforme à la façon de le calculer, cf annexe B). Il est donc plus intéressant d'utiliser une version normalisée par le nombre de classifieurs, soit NFM. De façon générale, $Q_{average}$ est faiblement corrélée (plutôt négativement) avec les autres mesures lorsque la cardinalité est fixée. C'est la mesure qui se démarque le plus du lot, formant même un angle de près 90° avec FM et TI selon les cas de figures (l'orthogonalité signifiant l'absence de corrélation). Aucune mesure ne semble être fortement corrélée à la performance.

Les solutions obtenues par algorithme génétique simple ont en commun le biais d'avoir été générées en tentant de maximiser la performance. Ce biais est acceptable dans notre contexte puisque dans toutes nos expériences, la performance est maximisée d'une manière ou d'une autre. Cependant, pour observer les relations entres les différentes mesures de diversité, la cardinalité et la performance sans aucun biais, un autre ensemble de solutions a été étudié. Cet ensemble de 3000 **EoC** a été construit avec les 30 premières itérations d'une recherche stochastique (qui en comporte 1280 comme il est expliqué à la section 3.3.2). La taille de cet ensemble (3000 solutions) a été choisie de façon à être du même ordre de grandeur que la taille de l'ensemble obtenu par AG simple présenté précédemment. Les solutions de cet ensemble sont projetées sur toutes les combinaisons de deux axes possibles, formant ainsi une matrice de graphiques appelée *diagramme de dispersion* (cf figure 15-a). On voit encore une fois la nécessité d'utiliser une version normalisée de FM qui est très fortement corrélée avec la cardinalité. Toutes les mesures de diversité de même que l'erreur ont une relation avec la cardinalité même si cette dernière n'est linéaire que dans le cas de FM. Pour cette raison, la cardinalité a été fixée sur la figure 15-b à 50 classifieurs (qui correspond à la moyenne de l'ensemble de 3000 **EoC**).



a) 3000 solutions de cardinalité différente



b) 300 solution de même cardinalité (50)

Figure 15 **Diagramme de dispersion de solutions obtenues** par recherche stochastique. Sur la figure a), on peut voir 3000 solutions obtenues par la recherche stochastique projetées sur divers axes. Sur la figure b), seules les solutions de cardinalité 50 (moyenne) sont projetées. Card est la cardinalité et Amb est l'ambiguïté.

Le fait de fixer la cardinalité dévoile des relations entre les différentes mesures qui n'étaient pas visibles autrement. On peut ainsi voir que toutes les mesures de diversité sont corrélées entre-elles à certains degrés mais qu'aucune ne l'est avec l'erreur. L'ambiguïté, TI et EK semblent particulièrement liés par une relation linéaire. Puisque les **EoC** ainsi projetés ont été générés de manière aléatoire, on peut présumer que les relations révélées sont intrinsèques aux mesures elles-mêmes et ne dépendent pas d'autre facteur.

Le tableau V résume la nomenclature utilisée pour se référer aux différentes méthodes d'optimisation (décrites dans une section suivante) et fait la synthèse des mesures de diversité retenues.

Tableau V

Nomenclature pour la désignation des différentes méthodes d'optimisation

Nom	Description de la méthode
Ordonnancement	Utilisation des N-meilleurs classifi eurs (méthode déterministe)
Stochastique	Recherche stochastique guidée par le taux de reconnaissance maximal
AG Simple	Algorithme génétique simple : maximiser le taux de reconnaissance
NSGA-a	Maximiser conjointement le taux de reconnaissance et l'ambiguïté
NSGA-b	Maximiser conjointement le taux de reconnaissance et l'entropie (Kuncheva)
NSGA-c	Maximiser conjointement le taux de reconnaissance et NFM
NSGA-d	Minimiser conjointement le taux d'erreur et l'ambiguïté
NSGA-e	Minimiser conjointement le taux d'erreur et le nombre de classifi eurs
NSGA-f	Minimiser conjointement le taux d'erreur et la mesure $Q_{average}$

On s'attend à ce que les ensembles formés avec *NSGA*-a possèdent un grand nombre de *k*-**NN** plus ou moins performants puisqu'ils sont construits en maximisant à la fois la performance et l'ambiguïté. Puisque cette dernière est une mesure exprimant le degré de désaccord entre les classifieurs et que les classifieurs plus performants ont forcément moins de désaccord entre-eux que les moins performants, une pression est faite pour sélectionner des *k*-**NN** ayant des taux de reconnaissance différents. À l'inverse, *NSGA*-d devrait produire des ensembles de petites tailles avec des *k*-**NN** performants, puisque c'est ainsi

que le désaccord est le plus faible. Si l'AG Simple fonctionne, il devrait donner des **EoC** ayant de hauts taux de performance et *NSGA*-e devrait donner des **EoC** ayant de petites cardinalités. En revanche, les différences dans les résultats basés sur les autres mesures de diversité sont plus difficiles à prévoir. En effet, il semble que ces mesures soient toutes plus ou moins corrélées entre-elles. Dans ce cas, les résultats obtenus par l'une ou l'autre de ces métriques devraient être relativement semblables. Selon Kuncheva, il est très difficile, voire impossible, d'obtenir des valeurs négatives de $Q_{average}$ lorsque les ensembles sont constitués d'un grand nombre de classifieurs [49] (ce qui est notre cas). Il est donc probable que malgré l'effort de minimisation de la mesure, celle-ci demeure positive.

3.2.2 Caractérisation de la méthode des sous-espaces aléatoires appliquée au *k*-NN

Nous avons repris le protocole de Ho [33] sur la base de données NIST SD19 pour comprendre l'effet de la cardinalité des sous-espaces, de la valeur de k dans les k-**NN** et de la taille de la base d'apprentissage. Les résultats découlant de ces expériences ont permis de fixer les paramètres de l'ensemble à optimiser. La figure 16 illustre la façon dont ces expériences ont été conduites. Pour chaque taille de base d'apprentissage testée, un classifieur de type perceptron multi-couches (MLP) a été entraîné et sa performance a été mesurée. Les MLP sont constitués de 132 neurones d'entrée et de dix neurones de sortie (une sortie par classe). Le nombre de neurones sur la couche cachée a été déterminée empiriquement en le faisant varier et en mesurant les performances sur la base d'optimisation. L'architecture donnant les meilleures résultats sur cette dernière base a été retenue (une architecture différente pour chaque base d'apprentissage). Finalement, la performance d'un k-**NN** unique, utilisant la totalité des 132 caractéristiques disponibles, a aussi été mesurée pour être comparée à celles des ensembles. Toutes les mesures de performance ont été mesurées sur la base commune d'optimisation. Les valeurs des paramètres variés sont montrées dans le tableau VI.

Les résultats obtenus ont indiqués que l'ensemble de 100 k-**NN** offrait les meilleures performances pour une valeur de $k = 1$ (cf figure 19) et des sous-espaces ayant une cardinalité

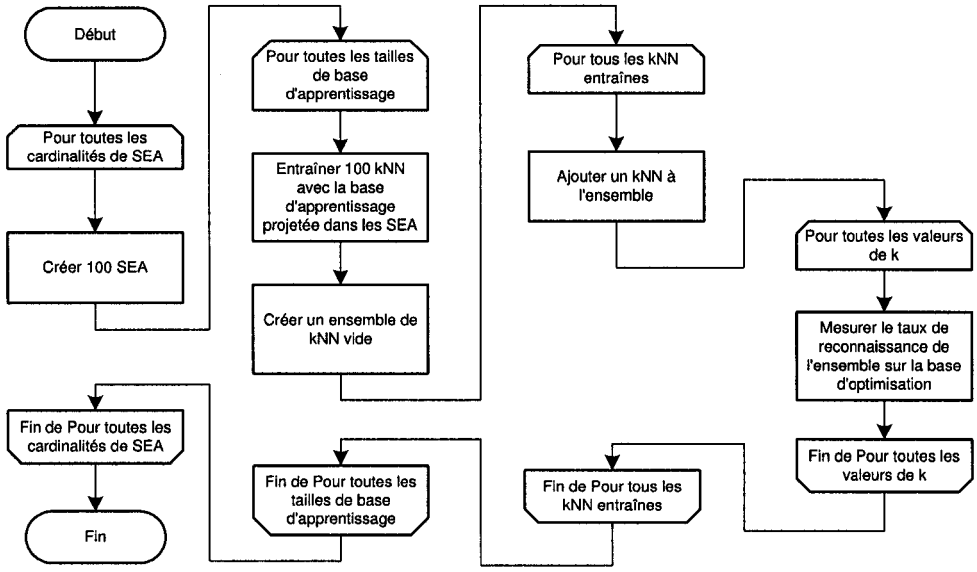


Figure 16 **Organigramme du déroulement des expériences préliminaires.**

Tableau VI

Valeurs des paramètres variés lors des expériences préliminaires

Paramètre	Valeur
Bases d'apprentissage utilisées	***TRAIN50k***, ***TRAIN25k***,***TRAIN10k***, ***TRAIN5k***, ***TRAIN1k***, ***TRAIN500***
Nombre de plus proches voisins (k)	1, 3, 5, 20
Cardinalité des sous-espaces aléatoires	8, 16, 32, 64
Nombre de classifi eurs	1, 2, 3, . . ., 100

de $f = 32$ caractéristiques (cf figure 20). La performance de l'**EoC** est supérieure ou égale à celle du réseau de neurones lorsqu'entraîné avec une base de données ne dépassant pas 5000 exemples (cf figure 17). Le reste de notre étude a donc porté sur ce cas de figure où le nombre de données d'apprentissage est restreint et la base d'entraînement a été fixée (***TRAIN5k***). Les valeurs de k et la cardinalité des sous-espaces ont aussi été fixées à la lumière de ces résultats préliminaires pour maximiser le taux de reconnaissance ($k = 1$ et $f = 32$).

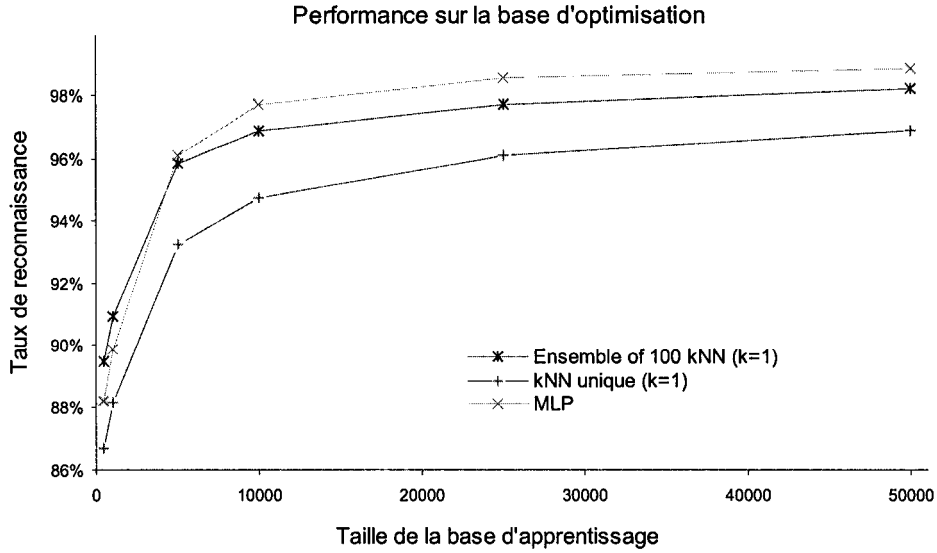


Figure 17 **Performances obtenues en fonction de la taille de la base de données.** Sur la base d'optimisation, les performances du MLP et celles d'un ensemble de 100 *k*-**NN** sont à peu près équivalentes lorsqu'entraînés avec 5000 prototypes. Le MLP a des performances supérieures avec les bases plus grandes, tandis que l'ensemble de *k*-**NN** est meilleur avec les bases plus petites.

On notera que les performances avec $f = 32$ et $f = 16$ sont très semblables. Le choix de $f = 32$ se justifie par le fait qu'une plus grande cardinalité offre plus de marge de manoeuvre en cas de caractéristiques absentes ou manquantes lors de la mise en service du classifieur. En effet, si la moitié des caractéristiques ne sont pas disponibles, un classifieur à 32 caractéristiques aura en moyenne 16 valeurs disponibles pour prendre une décision alors qu'un classifieur de cardinalité 16 n'en aurait plus que 8.

Un désavantage de l'utilisation d'un MLP est sa tendance à mémoriser les petites bases d'apprentissage. Comme le montre la figure 18, lorsqu'il est entraîné avec des bases d'apprentissage inférieures à 5000 exemples, ses performances en généralisation tombent très rapidement alors que rien ne laisse présager ce phénomène en ne regardant que les taux de reconnaissance sur la base d'apprentissage. Ceci explique pourquoi un ensemble de *k*-**NN**

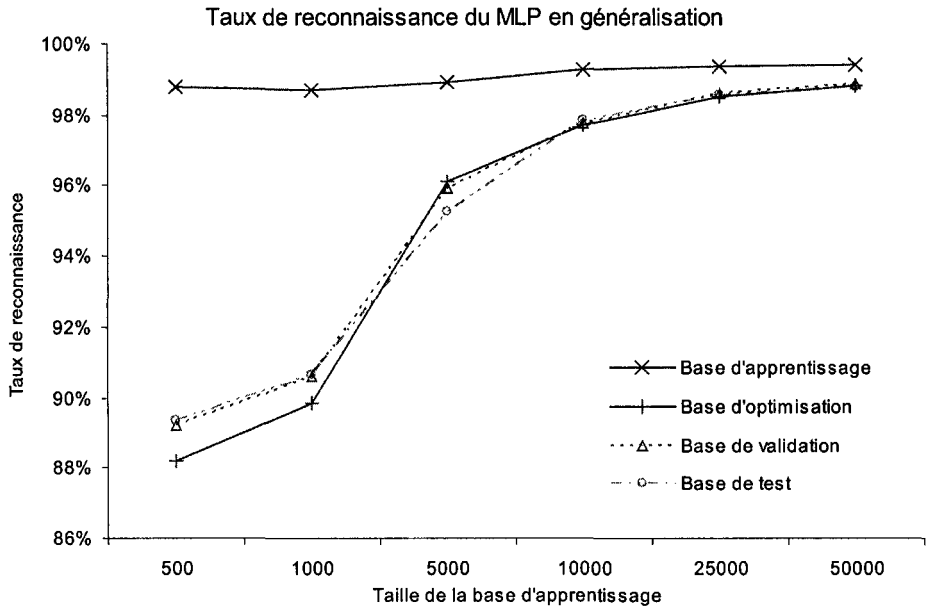


Figure 18 **Effet de la généralisation sur le MLP.** Le MLP a tendance à mémoriser les petites bases d'apprentissage. Ainsi, lorsqu'il est entraîné avec des bases d'apprentissage inférieures à 5000 exemples, ses performances en généralisation tombent très rapidement alors que rien ne laisse présager ce phénomène en ne regardant que les taux de reconnaissance sur la base d'apprentissage.

peut avoir des taux de reconnaissances supérieurs au MLP dans un contexte où il y a peu de prototypes d'entraînement.

Le MLP utilisé en référence a été entraîné avec les paramètres suivants :

- **Taux d'apprentissage initial :** 0,5
- **Momentum :** 0,93
- **Gradient :** 0,5

Le nombre de neurones sur la couche cachée ainsi que le nombre d'époques nécessaires à l'entraînement du MLP utilisé en référence ont été ajustés pour chaque taille de base d'entraînement de façon à en maximiser de taux de reconnaissance sur la base d'optimisation.

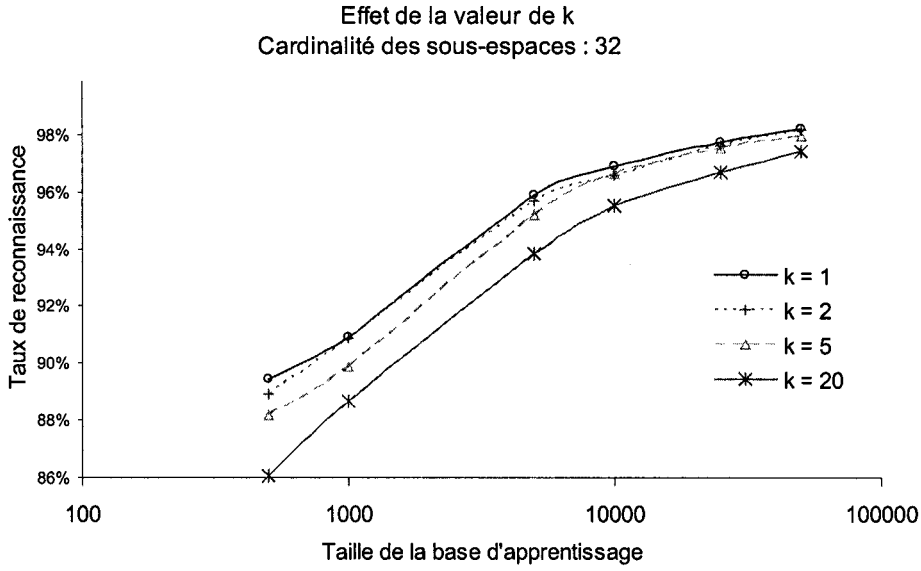


Figure 19 **Performances obtenues pour différentes valeurs de k.** Bien que la différence semble diminuer avec la taille de la base d'apprentissage, le taux de reconnaissance de l'ensemble de 100 k-**NN** est d'autant meilleur que la valeur de k est petite. $k = 1$ donne les meilleurs résultats.

3.3 Recherche des ensembles

La recherche, ou l'optimisation, des ensembles de classifieurs constitue le coeur de la partie expérimentale. La procédure utilisée est illustrée à la figure 21 sous forme d'organigramme. Un ensemble de sous-espaces aléatoires a été fixé pour toute la durée des expériences et 100 k-**NN** ont été entraînés avec la base d'apprentissage projetée dans ces derniers. Cet ensemble de k-**NN** constitue l'**EoC** à optimiser, la recherche consistant à trouver le meilleur sous-ensemble au sein de l'ensemble de départ. Le principe de recherche d'un **EoC** est le suivant :

- **Recherche avec la base d'optimisation** - À cette étape, on utilise la base d'optimisation pour mesurer le taux de reconnaissance et les mesures de diversité. L'al-

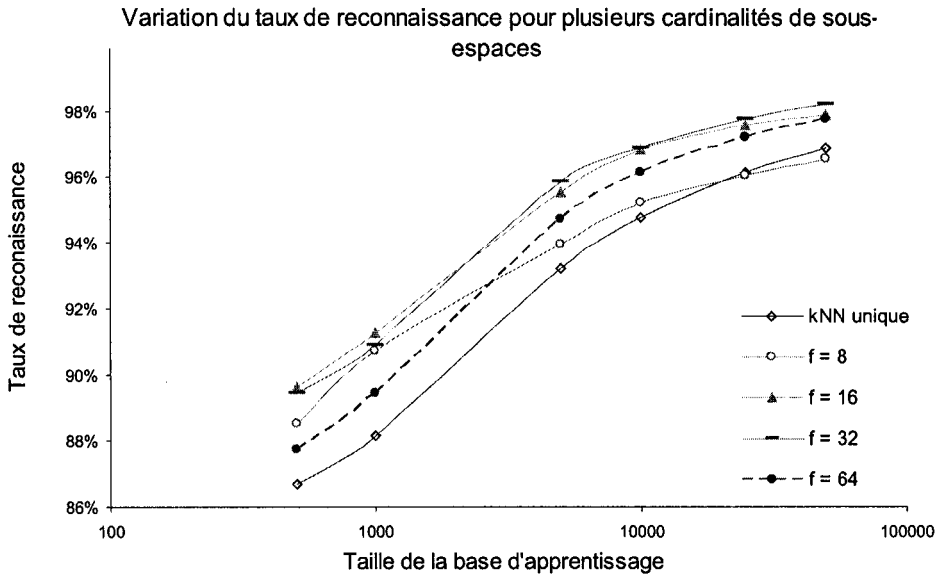


Figure 20 **Performances obtenues pour différentes valeurs de cardinalité f.** Il y a peu de différence de performance entre les ensembles projetés dans un espace à 16 ou à 32 caractéristiques (alors qu'il y a un rapport du simple au double entre ces deux cardinalités). Une cardinalité aussi faible que 8 caractéristiques fonctionne bien en généralisation pour les petites base d'apprentissage mais est rapidement surpassée quand la taille de cette dernière augmente.

gorithme de recherche est guidé par ces résultats pour produire un ensemble de solutions potentielles.

- **Choix avec la base de validation** - La meilleure solution de l'ensemble des solutions obtenu à l'étape précédente est conservée. Celle-ci est définie comme étant le **EoC** ayant le meilleur taux de reconnaissance mesuré sur la base de validation. Dans les rares cas d'égalité, l'ensemble de classifieurs ayant la cardinalité minimale est considérer comme étant le meilleur.
- **Mesures sur la base de test** - Finalement le taux de reconnaissance, la cardinalité et la diversité de la solution sont mesurées avec la base de test.

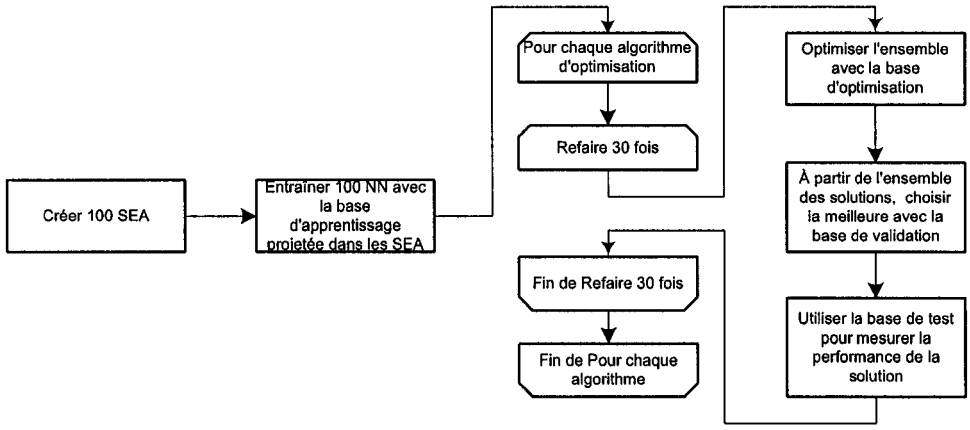


Figure 21 **Méthodologie de recherche employée.** Après avoir entraînés 100 *k*-**NN** projetés dans des SEA, on répète 30 fois le processus suivant pour chacune des méthodes étudiée : 1) Faire la sélection de classifieurs à l'aide de la base d'optimisaition. 2) Utiliser la base de validation pour déterminer quelle est la meilleure solution. 3) Se servir de la base de test pour mesurer les performances de la solution retenue

3.3.1 Recherche par ordonnancement

La recherche par ordonnancement, aussi appelée méthode des N-meilleurs classifieurs, se base sur une heuristique simple et déterministe pour former un ensemble de classifieurs [65]. Il s'agit, en premier lieu, de trier par ordre décroissant de performance les classifieurs de l'ensemble à optimiser (mesurée sur la base d'optimisation). En second lieu, on prend les n meilleurs classifieurs pour former un **EoC**. On peut déterminer n *a priori* selon la complexité désirée du système ou choisir la valeur de n pour laquelle le taux de reconnaissance de l'**EoC** est maximal. Dans ce dernier cas, on utilisera la base de validation pour déterminer la valeur de n après avoir trié les classifieurs en fonction de leur performances sur la base d'optimisation. Ce processus étant entièrement déterministe, il n'est pas nécessaire de refaire l'expérience plusieurs fois.

3.3.2 Recherche stochastique

La recherche stochastique est nécessaire pour contrôler l'efficacité des AG. En effet, chaque recherche effectuée par ces derniers fait l'évaluation de 128 individus $\times$ 1000 générations, pour un total de $1,28 \times 10^5$ solutions[2]. Pour contrôler la pertinence d'utiliser un algorithme de recherche aussi complexe que les AG, il est essentiel d'évaluer la qualité d'autant de solutions obtenues de manière aléatoires. La technique retenue pour ce faire est présentée dans l'algorithme 5.

Algorithme 5 : Recherche stochastique

Données :
S : l'ensemble des classifieurs à optimiser
P : ensemble des solutions
$PERF(E, DB)$: fonction qui retourne le taux de reconnaissance de l'ensemble E sur la base de données DB

```
début
    tant que Nombre d'itération maximal n'est pas atteint faire
        Créer un ensemble vide de classifieurs : S' ⟵ {φ}
        i ⟵ 1
        tant que S ≠ S' faire
            x ⟵ un élément tiré aléatoirement de S tel que x ∉ S'
            S' ⟵ x
            perf ⟵ PERF(S', base d'optimisation)
            si perf > PERF(P_i, base d'optimisation) alors
                P_i ⟵ S'
            i ⟵ i + 1
fin
```

Cette recherche stochastique évalue 100 solutions par itération de l'algorithme (c'est la cardinalité de l'ensemble de classifieurs à optimiser). Pour évaluer le même nombre de solutions qu'avec les AG, la recherche stochastique doit donc faire 1280 itérations (1280 $\times$

[2] Bien que le nombre de solutions évaluées ($1,28 \times 10^5$) puisse sembler important, ce dernier demeure très petit en comparaison de la taille de l'espace de recherche. En effet, l'espace de toutes les combinaisons possibles de 100 classifi eurs a une taille N qui se calcule comme suit : $N = 2^{100} \approx 1,3 \times 10^{30}$. Même en évaluant un milliard de solutions par secondes et en ayant commencé les calculs en même temps que la naissance présumée de l'Univers (évaluée à plus ou moins il y a 5×10^{17} secondes), nous n'aurions toujours pas terminé de parcourir l'espace de recherche aujourd'hui.

$100 = 128 \times 1000$). Il faut noter que ce type de recherche n'est pas purement stochastique puisqu'à chaque itération, une et seulement une solution est systématiquement évaluée pour toutes les cardinalités possibles. Il y a donc 1280 solutions évaluées pour toutes les cardinalités de 1 à 100. En sortie l'algorithme fournit un ensemble de 100 solutions, soit la meilleure pour chaque cardinalité.

Toutes les solutions n'ont donc pas la même chance d'être sélectionnées. Le nombre de solutions pour une cardinalité donnée correspond au coefficient binomial $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ avec $n = 100$ et la cardinalité k variant entre 1 et 100. Une solution de 50 k-**NN** a une probabilité de sélection presque nulle, de l'ordre de $1/\binom{100}{50} \approx 10^{-29}$ alors que les solutions composées d'un seul classifieur ont 1 chance sur 100 d'être sélectionnées. En contre partie, si la recherche était purement aléatoire, certaines cardinalités seraient injustement désavantagés par rapport à celles autour de la moyenne de 50 solutions (cf figure 22). Puisque la cardinalité est un paramètre important de notre étude, une recherche stochastique légèrement guidée pour assurer la représentation de toutes les tailles d'**EoC** est préférable à une recherche purement aléatoire.

3.3.3 Recherche dirigée à l'aide d'algorithmes génétiques

Deux types d'algorithmes génétiques ont été appliqués dans notre recherche d'ensembles de classifieurs. Un AG simple a été utilisé comme algorithme de recherche à un seul objectif (maximiser le taux de reconnaissance de l'ensemble). Pour les recherches à plusieurs objectifs, l'algorithme génétique multi-objectifs (AGMO) NSGA a été utilisé.

3.3.3.1 Algorithmes génétiques multicritères et front de Pareto

Vilfredo Pareto a montré qu'au début du 20^e siècle, 20% de la population italienne détenait 80% de la richesse du pays. Il en a tiré une loi empirique qui exprime que dans une société donnée, à un instant donné, le nombre N de contribuables de revenu r est une fonction puissance décroissante de ce revenu. C'est dire qu'il existe deux réels strictement positifs

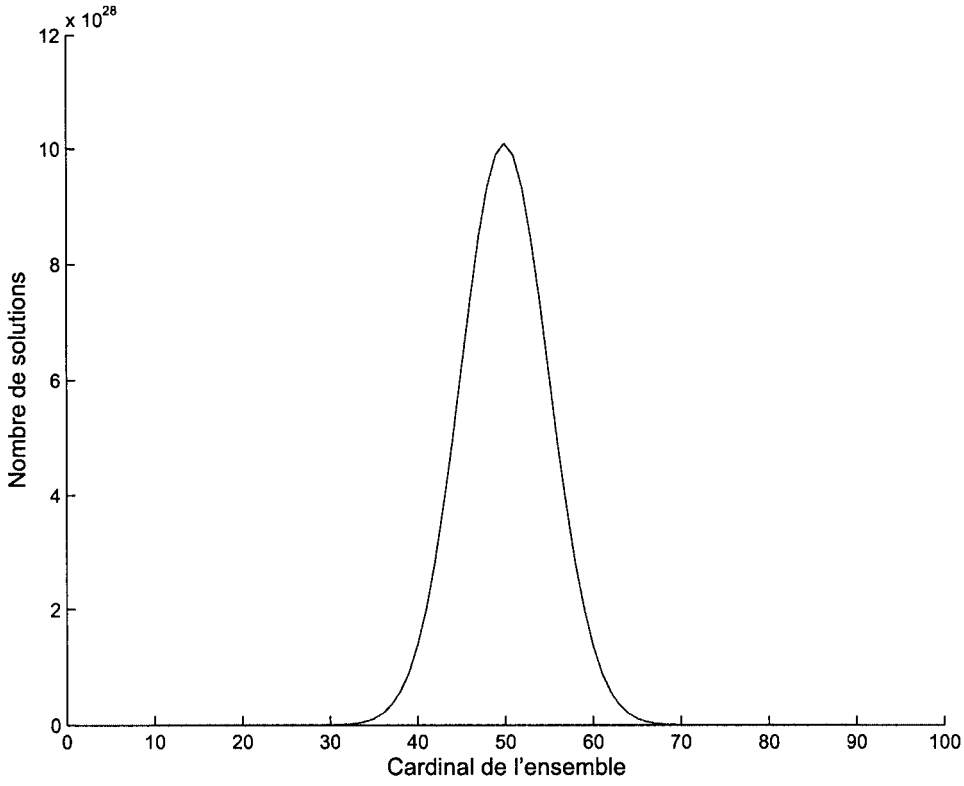


Figure 22 **Distribution du nombre de solutions en fonction du cardinal.** Il y a 100 **EoC** composé de 1 *k*-**NN**, 1 **EoC** composé de 100 *k*-**NN** et environ 10^{29} **EoC** composé de 50 *k*-**NN**.

a et k tels que :

$$N = \frac{k}{r^a} \tag{3.1}$$

Les algorithmes génétiques multi-objectifs utilisent un principe inspiré de cette loi de Pareto pour comparer des solutions ayant plusieurs objectifs contradictoires. Les solutions dans l'espace de recherche sont triées par front de Pareto et les valeurs d'adaptation sont calculées en fonction du rang occupé (voir à la figure 23). Le front de Pareto est défini, dans ce contexte, comme étant l'ensemble des solutions non dominées. Dans le cas d'un problème où il faut minimiser $\{f_1(x), f_2(x), \ldots, f_N(x)\}$, on définit que la solution x_1

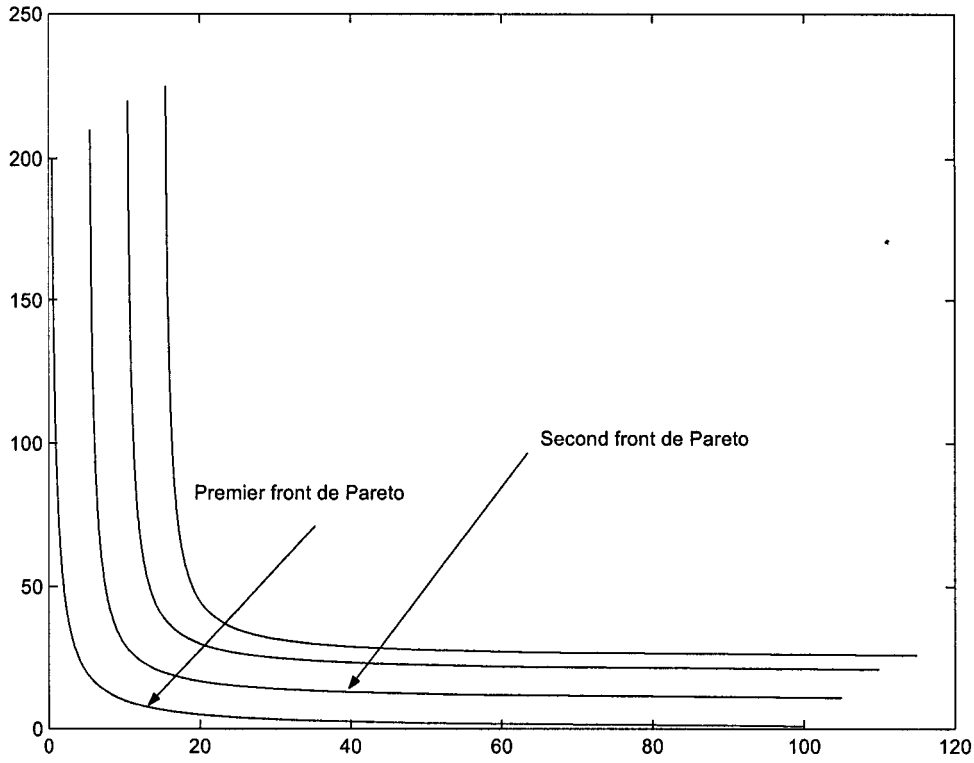


Figure 23 **Tri par front de Pareto [17].** Les individus non dominés forment le premier front de Pareto. Le second front de Pareto est constitué des solutions uniquement dominées par des solutions du premier front de Pareto et ainsi de suite. À chaque solution correspond donc un rang qui dépend de leur front d'appartenance. Les solutions de même rang sont considérées équivalentes.

domine la solution x_2 si et seulement si :

$$\begin{aligned} &\forall i \in \{1, \ldots, N\} : f_i(x_1) \leq f_i(x_2) \wedge \\ &\exists j \in \{1, \ldots, N\} : f_i(x_1) < f_i(x_2) \end{aligned} \tag{3.2}$$

Les solutions appartenant au premier front de Pareto ne sont dominées par aucune autre solution. Celles appartenant au second front de Pareto ne sont dominées que par celles appartenant au premier front et ainsi de suite jusqu'au dernier front de Pareto qui est constitué des solutions ne dominant aucune autre solution.

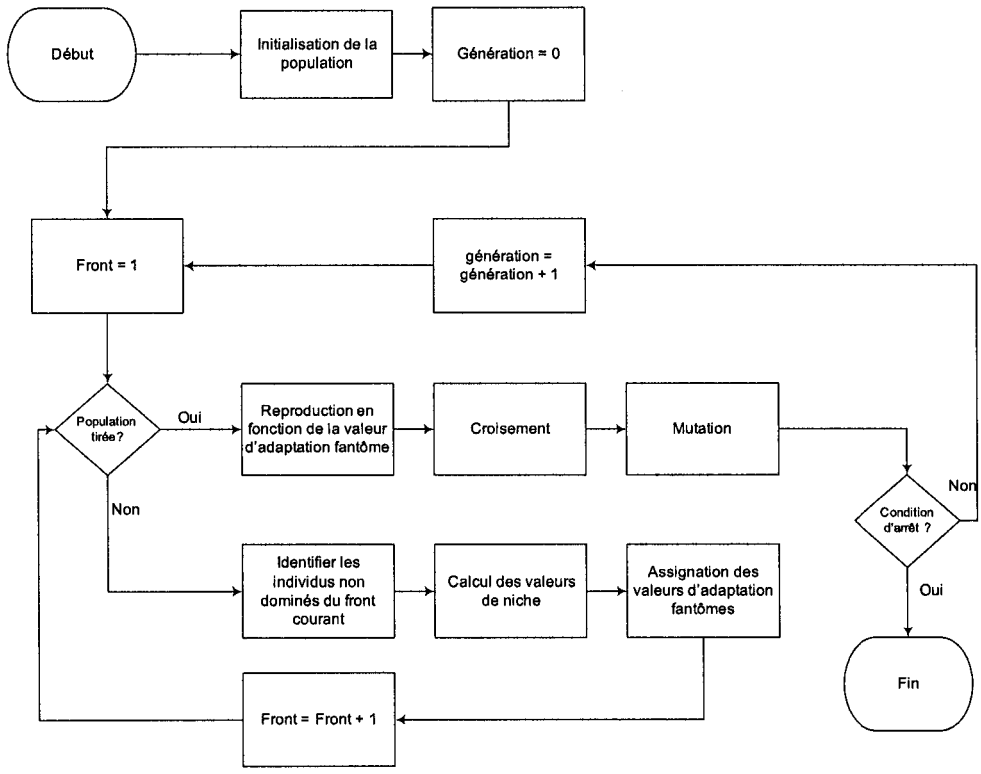


Figure 24 **Algorithme NSGA [17].** La différence la plus notable entre un AG simple et NSGA est le tri de la population par front de Pareto. Les solutions d'un même front voient ensuite leur VA dégradée par le partage des ressources.

Mis à part quelques particularités -dont le tri par front de Pareto- pour calculer la valeur d'adaptation, les AG multi-objectifs ne sont pas très différents des AG simples dont l'algorithme est présenté sous forme d'organigramme à la figure 9. Leur plus grande particularité est de fournir un ensemble de solutions optimales (le premier front de Pareto) parmi lequel on peut choisir une ultime solution en fonction de l'importance accordée à l'un ou l'autre critère. L'organigramme de NSGA est présenté à la figure 24.

3.3.4 Niche et diversité génétique

Il ne faut pas confondre la diversité génétique et la diversité des **EoC**. La diversité génétique est un concept qui renvoie à la variété des solutions. Cette variété peut être mesurée

dans l'espace des génotypes (distance entre les chromosomes) ou dans l'espace des phénotypes (distance entre les valeurs obtenues par les solutions pour les différents objectifs)[3]. La diversité génétique aide à éviter les minimums locaux et permet une plus large exploration de l'espace ; son importance est illustrée dans une étude de Burke [9]. Dans le cadre d'optimisation d'**EoC**, les solutions sont elles aussi des populations et la diversité de ces dernières en est une caractéristique importante. Cependant, sous la même appellation se cache deux concepts différents.

Parmi les façons de créer ou maintenir la diversité génétique, il y a le concept de niche écologique emprunté à l'écologie et proposé par Goldberg et Richardson [27]. C'est la façon dont l'algorithme NSGA s'y prend pour assurer un maximum de diversité génétique au sein des solutions. L'une des définitions de niche écologique donnée par l'Office québécois de la langue française (OQLF) est la suivante :

> «Milieu dans lequel les facteurs affectant la survie, la reproduction et la croissance sont favorables à un biotype particulier. Un tel habitat peut être discontinu ou encore constituer un gradient écologique. Les niches favorisent des hybrides ou des mutants défavorisés dans un autre milieu. »

L'OQLF définit également la niche écologique ainsi : « La plus petite unité d'habitat convenable, c'est-à-dire l'emplacement occupé par un organisme individuel.» Ensembles, ces deux définitions sont fidèles à la façon dont NSGA utilise la métaphore écologique. Cette dernière veut que si plusieurs individus évoluent à proximité les uns des autres, ils auront à partager les ressources disponibles. Les ressources étant limitées, ce partage les rendra plus faibles et diminuera leur potentiel reproductif. Cette pression favorisera l'éloignement des individus et, par conséquent, la diversité génétique.

Concrètement, l'application du concept de niche se fait en dégradant la valeur d'adaptation des solutions en fonction de leurs distances aux solutions appartenant au même front de

[3] La distinction de ces espaces tient au fait que deux chromosomes complètement différents peuvent produire des solutions très rapprochées en terme d'objectif et qu'au contraire, des solutions aux chromosomes similaires peuvent donner des résultats très éloignés.

Pareto tel que détaillé à l'a. Il est à noter que l'algorithme s'assure que, malgré cette dégradation, toutes les solutions sur un front de Pareto donné aient une valeur d'adaptation supérieure aux solutions qu'elles dominent. Ainsi, il est nécessaire de calculer les valeurs d'adaptation d'un front après partage des ressources avant de pouvoir assigner une valeur aux fronts suivants.

La mise à jour des valeurs d'adaptation se fait premièrement en calculant la fonction de partage suivante [17] :

$$Sh\left(d(i,j)\right) = \begin{cases} 1 - \left(\frac{d(i,j)}{\sigma_{share}}\right) & \text{si } d(i,j) < \sigma_{share} \\ 0 & \text{sinon} \end{cases} \tag{3.3}$$

où σ_{share} est un paramètre à fixer (rayon de la niche) et $d(i,j)$ est la distance euclidienne entre les solutions i et j (dans l'espace des phénotypes)

On appelle valeur d'adaptation fantôme (VAF) la valeur d'adaptation dégradée. Elle se calcule comme suit :

$$VAF_i = \frac{VA_i}{nc_i} \tag{3.4}$$

où VAF_i est la valeur d'adaptation fantôme de la solution i, VA_i est la valeur d'adaptation de la solution i et nc_i est la valeur de niche de la solution i définie comme suit :

$$nc_i = \sum_{j=1}^{N} Sh\left(d(i,j)\right) \tag{3.5}$$

Pour NSGA la valeur du paramètre de niche σ_{share} à été déterminée expérimentalement en faisant varier sa valeur et en répliquant 10 fois l'expérience pour chaque valeur testée. Le choix de la valeur à utiliser s'est fait en prenant celle pour laquelle le nombre moyen de solutions sur le front de Pareto était maximal. Cette mesure est imparfaite puisqu'elle ne tient pas compte directement de la répartition des solutions sur le front de Pareto. Par contre, nous avons constaté qu'en général, le nombre d'individus sur le front de Pareto

Algorithme 6 : Assignation de la VAF

Assignation de la VAF [17]
Données :
σ_{share} : un paramètre fixé
ϵ : un petit nombre positif
début
  $F_{min} \longleftarrow N + \varepsilon$
  $j \longleftarrow 1$
  Trier la population P en fonction des fronts de Pareto
  tant que $j \leq$ *nombre de fronts de Pareto* **faire**
    pour *tous les* $q \in P_j$ **faire**
      $F_j(q) \longleftarrow F_{min} - \epsilon$
      $nc_q \longleftarrow$ calculer la valeur de partage avec les équations 3.3 et 3.5
      $VAF_j(q) \longleftarrow$ calculer la valeur d'adaptation fantôme selon l'équation 3.4
    $F_{min} \longleftarrow min(VAF_j(q)$ tel que $q \in P_j)$
    $j \longleftarrow j + 1$
fin

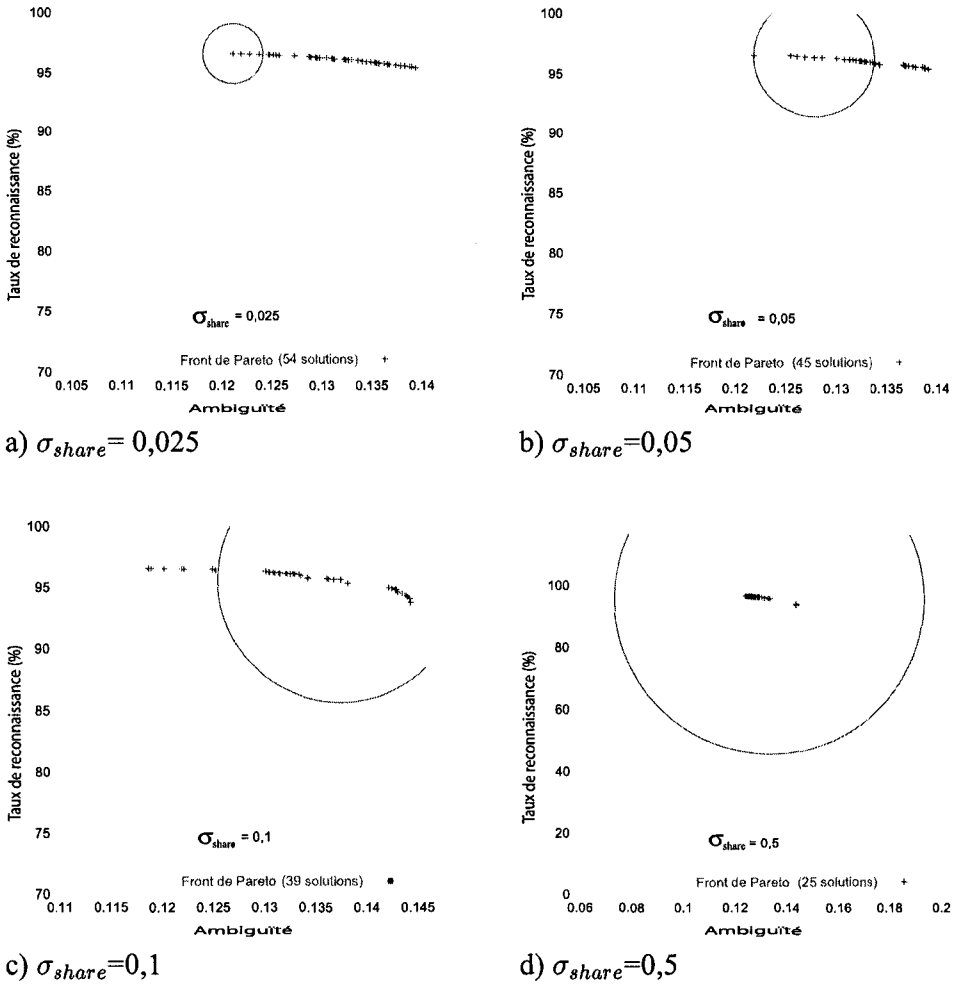
était fortement relié à la qualité de son échantillonnage. En effet, de grands trous sur le front de Pareto diminuent inévitablement le nombre total de solutions lui appartenant (cf figure 25). On trouvera dans le tableau VII les paramètres utilisés pour les algorithmes génétiques.

Tableau VII

Paramètres des expériences avec AG simple et NSGA

Nombre de générations	1000
Nombre d'individus	128
Élitisme	100%
Valeurs de niche testées NSGA	{0,025, 0,05, 0,1, 0,2, 0,3, 0,4, 0,5, 0,6, 0,7, 0.8}
Valeur de niche conservées	*NSGA*-{a,b,c,d,f} : 0,025
	NSGA-e : 0,05

L'élitisme est utilisé pour préserver les meilleures solutions. Cette méthode consiste à conserver les solutions d'une génération et à les comparer avec les solutions produites après les opérations génétiques (i.e. croisement, mutation). La génération suivante est ensuite composée des meilleures solutions parmi ces deux ensembles (cf figure 26).



a) σ_{share}= 0,025 b) σ_{share}=0,05

c) σ_{share}=0,1 d) σ_{share}=0,5

Figure 25 **Influence de la niche sur le front de Pareto pour différentes valeurs de σ_{share}.** Fronts de Pareto obtenus avec *NSGA*-a et différents rayons de niche (paramètre σ_{share}). Centré sur une solution du front de Pareto, le cercle montre la limite de la zone d'influence de la niche. Plus le nombre de solutions est élevé, moins il y a de «trous» dans le front de Pareto. L'échelle en d) a été changée en raison de la taille du cercle qui ne serait pas visible autrement.

3.4 La création d'un ensemble de 100 *k*-NN

La création d'un ensemble de 100 *k*-**NN** projetés dans des sous-espaces de 32 caractéristiques se fait en effectuant 100 fois une pige sans remise de 32 étiquettes de caractéris-

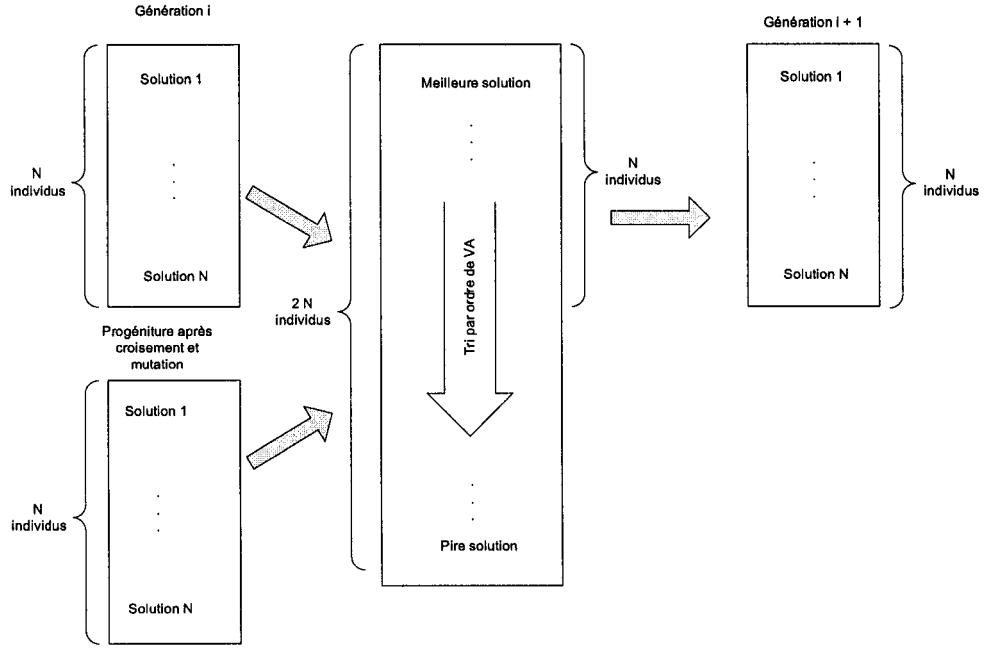


Figure 26 **Élitisme générationnel [17].** Les solutions issues d'une population ne feront partie de la nouvelle génération que si elles sont meilleures que leurs parents.

tiques parmi les 132 disponibles. Chacun des 100 ensembles de 32 étiquettes constitue un masque qui sera appliqué aux k-**NN** lors de l'étape de classification. La figure 27 montre la distribution des caractéristiques pigées.

Pour permettre la comparaison des différents **EoC** générés, trois classifieurs servent de référence. Le premier est un k-**NN** classique ($k = 1$) utilisant la totalité des 132 caractéristiques. Le second est l'ensemble initial composé de 100 k-**NN**. C'est la référence la plus significative puisque c'est à partir de cet ensemble que se fera la sélection de classifieurs. Finalement, les performances d'un réseau neuronal de type MLP ont aussi été mesurées. Tous ces systèmes ont été entraînés sur la même base de données, soit ***TRAIN5K***. Les résultats de références sont reportés au tableau VIII.

On remarque que le k-**NN** simple est moins performant que l'**EoC** ou que le MLP, avec une différence avoisinant 3%. L'**EoC** et le MLP ont des performances similaires en géné-

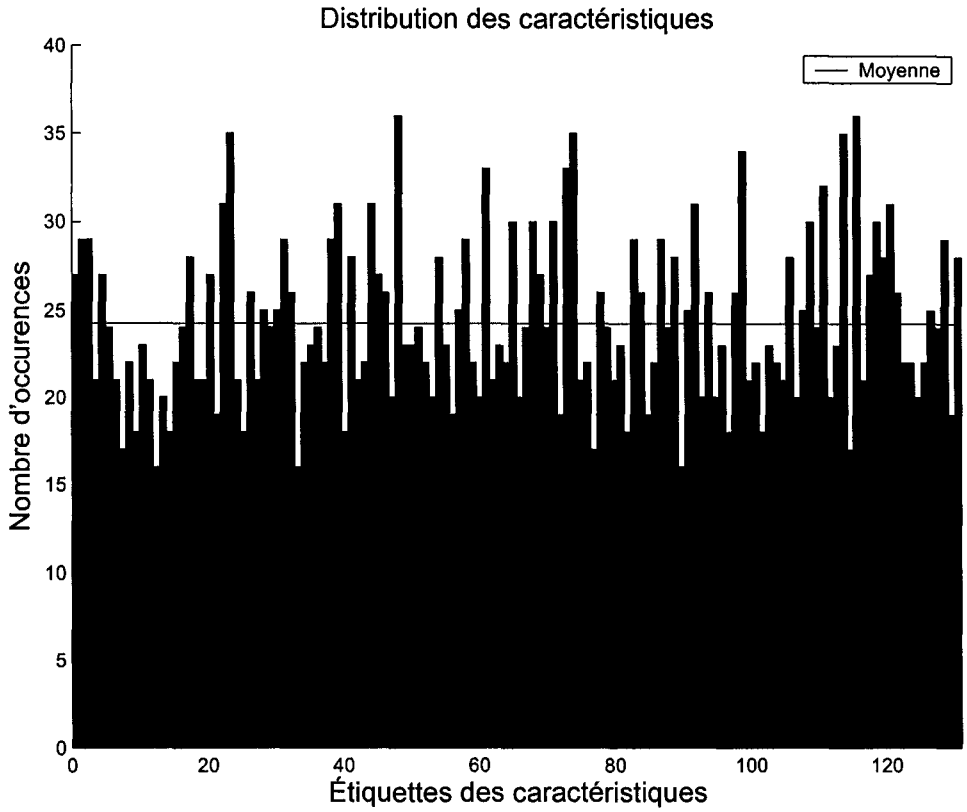


Figure 27 **Histogramme des caractéristiques dans l'ensemble de 100 *k*-NN.** La ligne horizontale représente le nombre moyen d'occurrences et correspond à $\frac{32}{132} \times 100 \approx 24,24$, c'est-à-dire le rapport entre le nombre de caractéristiques par classifieur et le nombre de caractéristiques total disponibles multiplié par le nombre de classifieurs.

ralisation (bases de validation et de test). Ceci est une conséquence normale du choix de la taille de base d'apprentissage. En effet, il a été déterminé que l'ensemble de 100 *k*-**NN** performait mieux que le MLP pour de petites bases d'apprentissage et qu'avec notre base de données, les performances sont à peu près équivalentes pour 5000 prototypes (voir figure 17).

Tableau VIII

Taux de reconnaissance de référence sur différentes bases de données

	Optimization	Validation	Test
k-NN simple (132 caract.)	93.23%	93.33%	93.34%
Ensemble de 100 *k*-NN	95.79%	96.09%	96.28%
MLP (132 caract.)	96.11%	95.91%	95.27%

Résumé du cadre de recherche

La majeure partie des techniques de sélection de classifieurs étudiées ne sont pas déterministes. Pour cette raison, elles sont répétées 30 fois afin de permettre ultérieurement le calcul de tests statistiques sur les moyennes et les variances obtenues. Le nombre de 30 échantillons constitue la limite à partir de laquelle on considère que le théorème central limite s'applique, ce qui permet de comparer moyenne et variance à l'aide de tests paramétriques (test t pour la moyenne et test de Fisher pour la variance) [63]. On pourra ainsi vérifier quels types de recherche produisent en moyenne les meilleurs ensembles, c'est-à-dire les **EoC** ayant les plus hauts taux de reconnaissance sur la base de test. La moyenne du nombre de classifieurs par **EoC** pourra aussi être testée de la même façon. Cette approche autorise à juger de la supériorité d'une méthode par rapport à une autre selon le critère vérifié. Sans cette précaution, il serait hasardeux de juger si la différence entre deux résultats n'est pas uniquement due aux variations aléatoires intrinsèques à tout processus non déterministe.

Le cadre expérimental du présent travail est défini comme suit :

1. Créer un ensemble de 100 *k*-**NN** projetés dans des sous-espaces de 32 dimensions. Chacun des *k*-**NN** a une valeur de $k = 1$ et est entraîné sur la base ***TRAIN5K***.
2. Utiliser les différentes méthodes de sélection de classifieurs présentés en 3.3 et comparer les ensembles ainsi produits.

3. Mesurer la variabilité et la reproductibilité de ces méthodes.
4. Émettre un avis sur les mesures de diversité employées dans le cadre de la sélection de classifieurs.
5. Émettre des recommandations quant à la meilleure façon de créer un **EoC** de *k*-**NN** par sélection de classifieurs.

Le chapitre suivant présente les résultats de la mise en oeuvre de ce protocole. Dans un premier temps, sont présentés les meilleurs **EoC** obtenus par chacune des méthodes. Dans un second temps, ces résultats sont étudiés d'un point de vu statistique et d'un point de vu de reproductibilité. Le chapitre se clos en proposant une interprétation des résultats et en résumant les avantages et inconvénients de différentes approches de sélection de classifieurs étudiées.

CHAPITRE 4

ANALYSE DES RÉSULTATS

Cette partie du mémoire présente les résultats des expériences réalisées. Différentes méthodes de sélection de classifieurs dans le but de former un **EoC** performant et peu complexe ont été testées. Outre une méthode heuristique déterministe, des techniques de recherche non-déterministes ont été appliquées et répliquées 30 fois afin d'en vérifier le comportement statistique. Une partie des résultats qui suivent a déjà fait l'objet d'une communication [70] et de deux rapports techniques RDDC-Valcartier [71, 72].

4.1 Les méthodes de sélection de classifieurs

Les différentes façons de sélectionner des classifieurs étudiées ont toutes permis de diminuer la complexité de l'**EoC** initial tout en conservant ou en améliorant la performance. Les résultats reportés au tableau IX sont ceux des meilleurs ensembles (plus haut taux de reconnaissance mesuré en validation) trouvés par chacune des méthodes. Pour les méthodes non-déterministes, le processus de recherche a été répété 30 fois, chaque itération donnant un meilleur **EoC** en sortie. Les résultats du tableau sont ceux du meilleur **EoC** parmi l'ensemble des 30 «meilleurs».

Tel qu'expliqué au chapitre 3, chacune des méthodes de recherche génère un ensemble de solutions à partir duquel est choisi l'**EoC** ayant la meilleure performance sur la base de validation. Les figures de l'annexe F montrent les ensembles de solutions dans lesquels se trouvent les **EoC** du tableau IX, c'est-à-dire les **EoC** les plus performants trouvés par chacune des méthodes. Notons que les solutions de l'AG Simple sont projetés artificiellement sur deux axes (performance et complexité) puisque seul le taux de reconnaissance est utilisé dans la recherche. Les autres ensembles de solutions sont projetés dans leur espace naturel. Les ensembles de solutions obtenus avec algorithmes génétiques multi-objectifs sont représentés par leur front de Pareto pour les mesures faites avec la base d'optimisa-

Tableau IX

Résultats obtenus sur les trois bases de données par le meilleur **EoC** (en validation) produit par chaque méthode de recherche

	#Classifi eurs	Optimisation	Validation	Test
k-NN simple (132 caract.)	1	93.23%	93.33%	93.34%
Ensemble de 100 *k*-NN	100	95.79%	96.09%	96.28%
MLP (132 caract.)	1	96.11%	95.91%	95.27%
Ordonnancement	76	95.93%	96.15%	96.26%
Stochastique	46	96.04%	96.35%	96.28%
AG Simple	31	96.82%	96.43%	96.41%
NSGA-a	44	96.57%	96.33%	96.40%
NSGA-b	31	96.69%	96.38%	96.36%
NSGA-c	25	96.62%	96.38%	96.44%
NSGA-d	28	96.73%	96.36%	96.41%
NSGA-e	24	96.84%	96.29%	96.40%
NSGA-f	28	96.47%	96.36%	96.34%

tion. Ces mêmes points sont ensuite évalués sur la base de validation et leur performance actualisée sur le graphique (mais pas leur second objectif dont on conserve la valeur mesurée en optimisation).

Les ensembles de solutions sont différents les uns des autres selon la méthode utilisée pour les obtenir. Ainsi, l'AG Simple ne génère que très peu de solutions qui sont toutes semblables alors que les recherches stochastique et par ordonnancement génèrent systématiquement 100 solutions selon une courbe caractéristique. Bien que les recherches avec NSGA génèrent toutes des fronts de Pareto, ils n'ont pas tous la même nature ni la même qualité. Ainsi on voit qu'avec *NSGA*-d, le front de Pareto est moins bien échantillonné qu'avec les autres méthodes. Aussi, la nature discrète du second objectif de *NSGA*-e crée un ensemble de solutions différents des autres fronts de Pareto, le rapprochant des ensembles de solutions obtenus par les méthodes stochastique et d'ordonnancement.

4.1.1 Recherche par ordonnancement

La méthode d'ordonnancement des classifieurs a été appliquée sur l'ensemble des 100 *k*-**NN** de départ en triant ces derniers par ordre décroissant de performance (obtenue sur la base d'optimisation). Quelques résultats sont reportés au tableau X. La base de validation a servi à déterminer le nombre de *k*-**NN** donnant le plus haut taux de reconnaissance (76). On remarque que les taux de reconnaissance sont inférieurs à ceux de l'ensemble de départ (100 *k*-**NN**), tant en validation qu'en test même si la différence n'est pas très grande. Par exemple, seulement 0,12% séparent le taux de reconnaissance en test de l'ensemble à 25 classifieurs de celui de l'ensemble non optimisé à 100 classifieurs alors que la complexité a été divisée par quatre.

Tableau X

Taux de reconnaissance obtenus par ordonnancement des N meilleurs classifieurs

Nombre de *k*-**NN**	15	20	25	76	100
Validation	95.76%	95.87%	96.00%	96.15%	96.09%
Test	96.00%	96.12%	96.16%	96.26%	96.28%

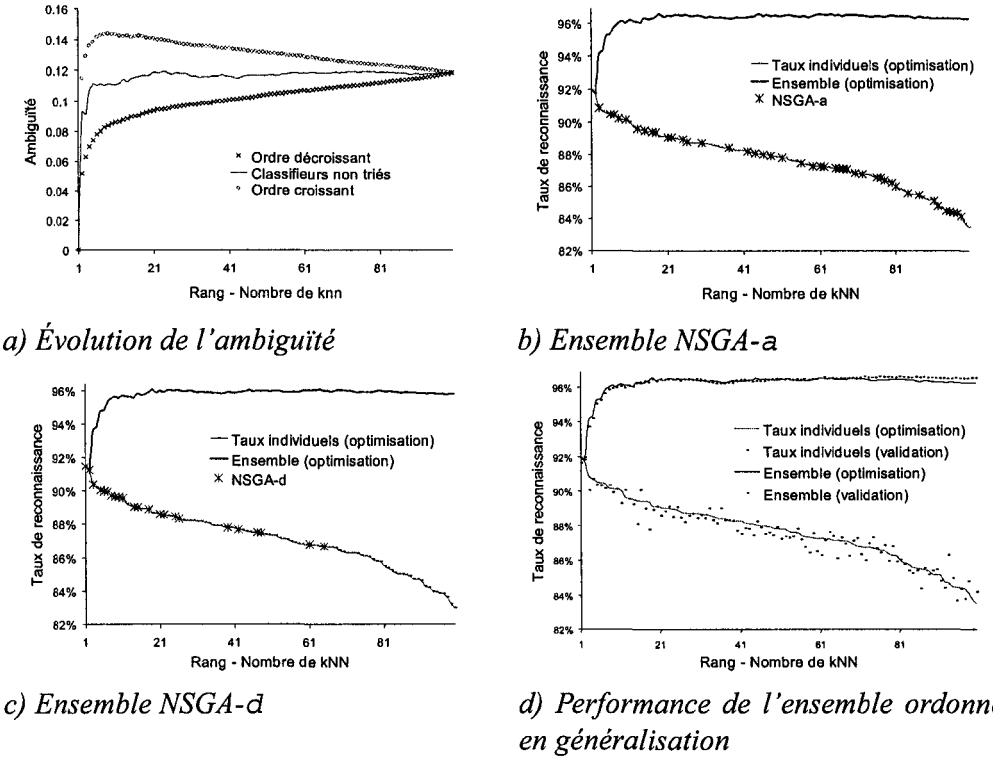
Sur la figure 28-{b,c}, on peut voir que la méthode d'ordonnancement permet de rapidement trouver un ensemble performant puisque le plateau est à peu près atteint dès l'agrégation d'un dizaine de classifieurs. En effet, la courbe du haut représente le taux de reconnaissance de l'ensemble formé par les N meilleurs *k*-**NN** alors que la courbe du bas indique la performance individuelle du *k*-**NN** de rang N. Les astérisques indiquent les classifieurs qui ont été sélectionnés pour former le meilleur ensemble avec *NSGA*-a (cf figure 28-b) et avec *NSGA*-d (cf figure 28-c). On voit que les meilleurs classifieurs ne sont pas systématiquement sélectionnés par ces méthodes et que des *k*-**NN** parmi les moins performants font partie des ensembles ainsi créés. Ces **EoC** sont d'ailleurs plus performants que ceux formés par la méthode d'ordonnancement (voir tableau IX).

La figure 28-a montre certaines relations entre l'ambiguïté et la qualité des classifieurs d'un ensemble. La courbe du haut est l'évolution de la valeur de l'ambiguïté lorsqu'on tri les classifieurs par ordre croissant de performances. La courbe du bas montre la même valeur quand on tri les classifieurs par ordre décroissant de performance (i.e. par la méthode d'ordonnancement). Ainsi l'ambiguïté d'un classifieur est d'autant plus élevée que la concentration de *k*-**NN** performants est faible. La courbe du centre montre l'ambiguïté lorsque les *k*-**NN** ne sont pas triés, c'est-à-dire en conservant l'ordre d'arrivée aléatoire dans lequel ils ont été initialement créés. Comme on s'y attendait, l'ensemble formé avec *NSGA*-a (maximisation de l'ambiguïté) possède un grand nombre de *k*-**NN** pas tous performants (cf figure 28-b) alors que *NSGA*-d (minimisation de l'ambiguïté) génère un **EoC** ayant moins de classifieurs et avec un plus grande proportion de *k*-**NN** performants (cf figure 28-c).

Finalement, la figure 28-d montre qu'il y a peu de différence entre les performances mesurées sur la base d'optimisation et celles mesurées sur la base de validation, tant au niveau des ensembles (courbes du haut) qu'au niveau des classifieurs individuels (courbes du bas). Bien que le rang des classifieurs dépende de la base de données, les performances de ceux-ci sont plus ou moins équivalentes d'une base à l'autre.

4.1.2 Les méthodes de recherche non-déterministes

Les méthodes de recherche non-déterministes sont appelées ainsi en raison de leur caractère non reproductible. En effet, ces techniques utilisent à divers degrés des événements aléatoires dans leur fonctionnement et deux itérations de la même expérience ne donneront pas des résultats identiques (bien que l'on s'attende à ce qu'ils soient similaires). Pour cette raison, il est plus intéressant de comparer non pas les meilleurs résultats parmi 30 itérations mais plutôt les résultats moyens de celles-ci. Les moyennes du nombre de classifieurs et des taux de reconnaissance mesurés sur les 3 bases de données sont représentées à la figure 29.



a) Évolution de l'ambiguïté

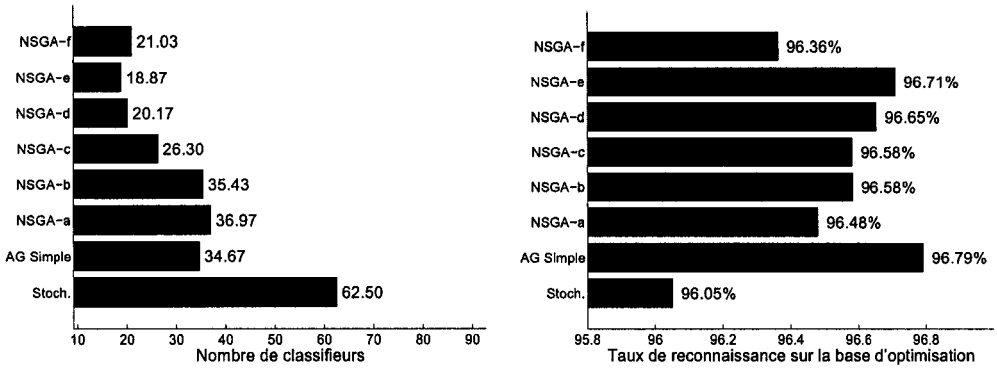
b) Ensemble NSGA-a

c) Ensemble NSGA-d

d) Performance de l'ensemble ordonné en généralisation

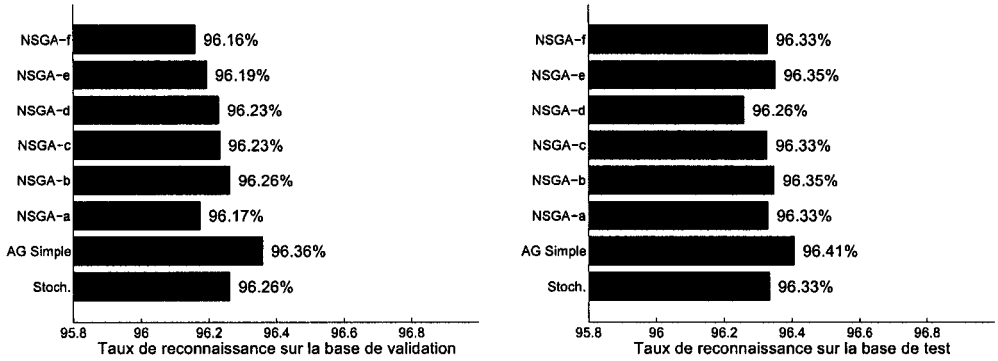
Figure 28 **Progression du taux de reconnaissance en triant les classifieurs par ordre décroissant de performance.** *NSGA*-a produit des ensembles avec beaucoup de classifieurs dont une grande partie n'est pas des plus performants. *NSGA*-d génère des ensembles plus petits avec une plus forte concentration de «bons» classifieurs.

On constate que la méthode qui donne en moyenne les meilleures performances est l'AG Simple, quelque soit la base de données utilisée pour les mesurer. Cette méthode, qui ne met de pression que sur la maximisation du taux de reconnaissance, est aussi celle qui a donné l'**EoC** le plus performant sur la base de validation et parmi les meilleur en test (cf tableau IX). Cette performance est atteinte en divisant plus ou moins par 3 le nombre de classifieurs par rapport à l'ensemble de départ (moyenne de 34,6 classifieurs). Bien que substantielle, cette diminution de la complexité est surpassée par plusieurs autres méthodes.



a) Nombre de classifieurs

b) Taux de reconnaissance sur la base d'optimisation



c) Taux de reconnaissance sur la base de validation

d) Taux de reconnaissance sur la base de test

Figure 29 **Caractéristiques moyennes des ensembles obtenus.**

La méthode qui domine en terme de diminution de la complexité est *NSGA*-e puisqu'elle crée en moyenne des **EoC** plus de 5 fois plus petits que l'ensemble original (18,9 classifieurs). Malgré cette réduction du nombre de classifieurs, cette méthode demeure excellente pour construire des **EoC** performants puisque ses ensembles arrivent second après ceux de l'AG Simple pour le meilleur taux de reconnaissance en optimisation et en test.

Les pires résultats en terme de complexité sont obtenus par la recherche stochastique alors qu'ils dépendent de la base de données utilisée pour mesurer lorsque l'on compare les taux de reconnaissance.

La figure 50 de l'annexe F montre que malgré l'effort de minimisation de la valeur de Q_{avr}, les valeurs de cette mesure sont encore largement positives (la plus petite étant 0,73). Étant donné le grand nombre de classifieurs par rapport au nombre de classes (~20 classifieurs et 10 classes), la tendance a avoir une corrélation positive dans les bonnes décisions l'emporte sur la corrélation négative des erreurs. Dans ce cas, la minimisation de Q_{avr} entraîne une diversité d'opinion sur les bonnes décisions avec pour conséquence un des plus faibles taux de reconnaissance moyen sur la base d'optimisation (dernier avant la recherche stochastique, cf figure 29-b). Cependant, cette faible performance ne semble pas se répercuter sur les résultats de la base de test.

Finalement, les différents taux de reconnaissance ne sont vraiment importants que sur la base de test puisque notre intérêt pour les classifieurs est de s'en servir en généralisation. Dans ce contexte, on peut dire que l'AG Simple fournit en moyenne les **EoC** les plus performants suivi d'assez près par toutes les autres méthodes puisqu'il n'y a qu'une différence de 0,15% entre le meilleur et le pire taux de reconnaissance moyen (respectivement 96,41% et 96,26%).

4.1.2.1 Recherche stochastique

Les résultats obtenus par la recherche stochastique sont comparables en terme de taux de reconnaissance à ceux obtenus par l'utilisation de NSGA. Cependant, le nombre de classifieurs est plus élevé avec cette méthode de recherche qu'avec la recherche dirigée. La figure 30 montre la performance du meilleur ensemble trouvé sur la base d'optimisation pour toutes les cardinalités de 1 à 100, ainsi que les résultats correspondants sur la base de validation.

La recherche stochastique nous permet d'échantillonner l'espace de recherche sans introduire de biais dû à la pression occasionnée par les objectifs des algorithmes génétiques. Ainsi, la figure 31 montre la projection des solutions trouvées par cette méthode pour une itération de 128 000 individus évalués. On peut considérer que ce graphique montre les

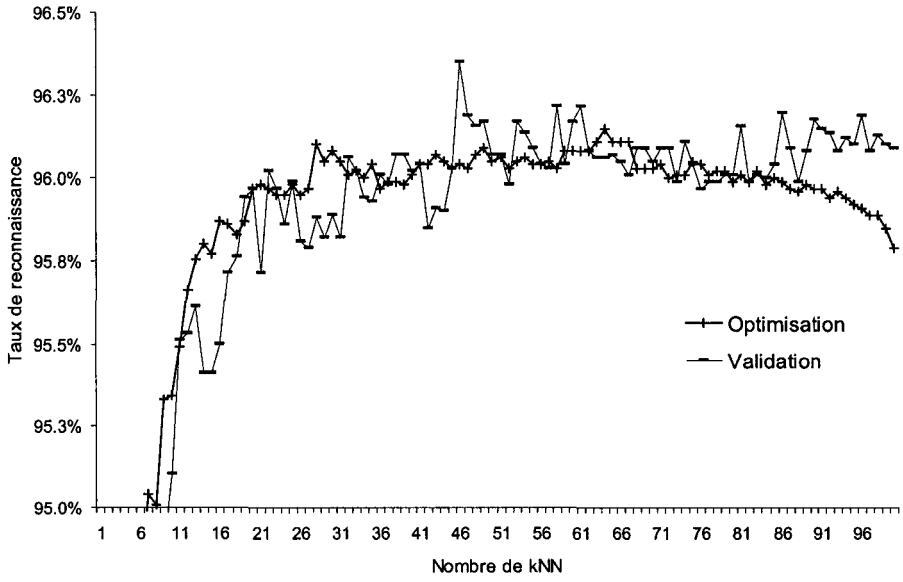


Figure 30 **Résultats de la recherche stochastique sur les bases d'optimisation et de validation.** Bien qu'il existe parfois de grandes différences entre les résultats en optimisation et ceux en validation, la tendance générale est la même avec l'atteinte d'une valeur plateau entre 20 et 30 classifieurs.

relations naturelles entre le taux d'erreur et le nombre de classifieurs de l'ensemble. Par conséquent, il nous renseigne sur l'espace que nous avons à parcourir. On voit qu'il existe une relation entre le nombre de k-**NN** et l'erreur qui rappelle la courbe $f(x) = 1/x$. On remarque aussi que cette relation comporte de fortes variations et qu'on peut espérer trouver les meilleurs ensembles autour d'à peine 20 ou 30 classifieurs, ce qui est tout à fait conforme avec les résultats obtenus.

4.2 Variabilité et reproductibilité

L'étude des tendances centrales d'un échantillon de données à l'aide de la seule moyenne est incomplète. Un premier inconvénient vient du fait que les valeurs de la moyenne ne sont pas réellement des valeurs que l'on retrouve parmi les échantillons. Par exemple, si les **EoC** ont une cardinalité moyenne de 18,9 classifieurs quand ils sont formés grâce

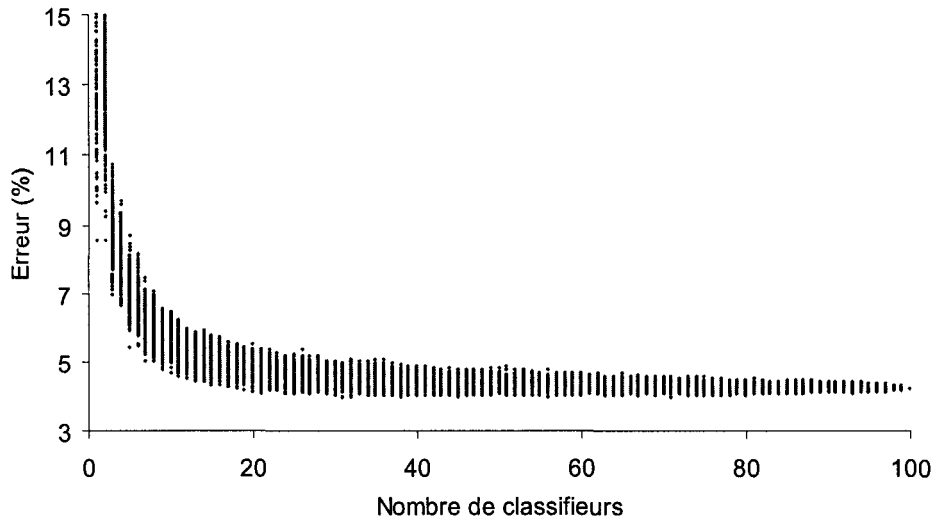


Figure 31 **Échantillonnage stochastique de l'espace de recherche.** Les meilleurs taux de reconnaissance s'obtiennent avec à peine une vingtaine de classifieurs. Plus le nombre de classifieurs est faible, plus la plage des performances possibles augmente.

à la technique *NSGA*-e, aucun des classifieurs n'aura jamais un tel nombre classifieurs puisque la cardinalité d'un ensemble doit être un entier. La valeur médiane répond à ce problème. Aussi, les valeurs moyennes et médianes n'apportent pas d'information sur la variabilité de cette tendance centrale. L'utilisation de diagrammes des quartiles est une façon de représenter à la fois la tendance centrale, par la médiane, et la distribution des échantillons le long d'un axe. On peut ainsi aisément comparer plusieurs échantillons en terme de valeur centrale, de dispersion et même d'éléments aberrants. Cependant, cette comparaison étant de nature plutôt qualitative, l'étude des moyennes et des variances à l'aide d'intervalles de confiance et de tests d'hypothèse est une façon de confirmer plus formellement les observations obtenues par les graphiques. Ces deux approches font l'objet des sections 4.2.1 et 4.2.2.

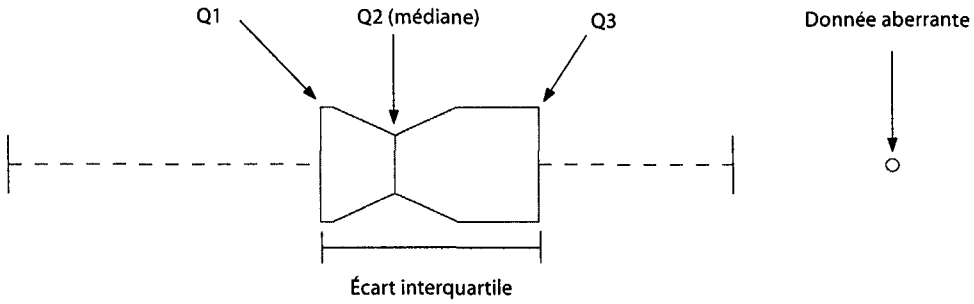


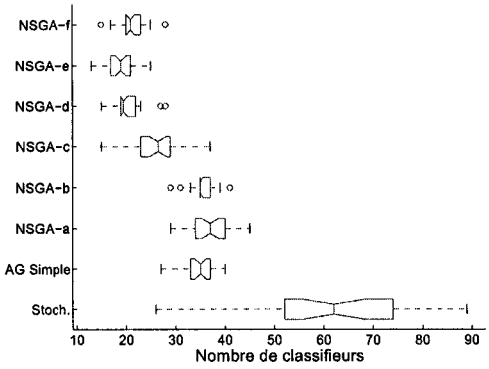
Figure 32 **Définitions d'un diagramme des quartiles [63].** Q2 est la médiane. Q1 est la médiane des valeurs inférieures à Q2 et Q3 est la médiane des valeurs supérieures à Q2. L'écart interquartile est la distance Q1-Q3. Les valeurs minimale et maximale sont aux extrémités des barres horizontales, sauf les données aberrantes qui sont représentées par un cercle.

4.2.1 Analyse des diagrammes de quartiles

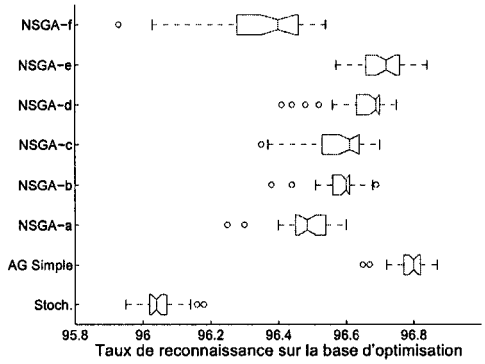
L'utilisation de diagrammes des quartiles est une façon simple de représenter une grande quantité d'information sur une distribution statistique. La figure 32 explique comment interpréter un diagramme des quartiles. Le point Q1 est la médiane des échantillons à gauche de Q2 alors que le point Q3 est la médiane des valeurs à droite de Q2. Le point Q2 est la médiane de l'ensemble des données (représentée par une encoche). On divise ainsi les échantillons en quatre groupes et la moitié des valeurs sont contenues dans la boîte centrale, entre le point Q1 et Q3. L'écart entre Q1 et Q3 est appelée écart interquartile (EI) et sert dans le calcul pour vérifier si une donnée est aberrante. Une donnée x est considérée aberrante si $x < Q1 + 1,5 \times EI$ ou que $x > Q3 + 1,5 \times EI$. La valeur 1,5 est arbitraire mais c'est un choix courant en statistique.

La figure 33 montre quatre diagrammes des quartiles des ensembles trouvés par les méthodes de recherche non déterministes. Les différents axes représentés sont le nombre de classifieurs (a) et les taux de reconnaissance sur les bases d'optimisation (b), de validation (c) et de test (d). Les données aberrantes sont représentées par un cercle.

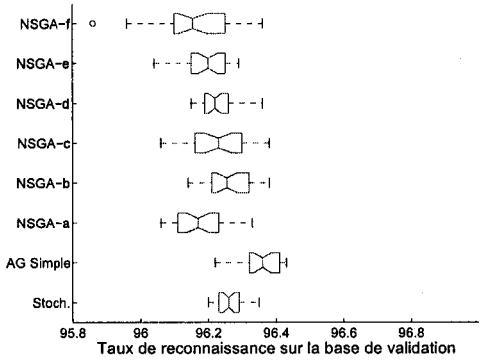
L'étendue de la plage des valeurs de cardinalité des ensembles formés par la recherche stochastique est remarquable, allant de 26 à 89. En comparaison le plus grand nombre de *k*-**NN** obtenu par les autres méthodes est inférieur au premier quartile des cardinalités obtenues par la recherche stochastique. La recherche avec *NSGA*-e donne encore une fois les meilleurs résultats en terme de complexité. On remarque que la plus grande valeur obtenue par cette méthode est plus petite que la médiane de toutes les autres formes de recherche, excepté *NSGA*-d.



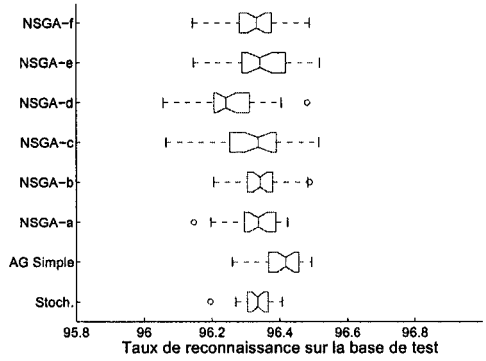
a) Nombre de classifieurs



b) Taux de reconnaissance sur la base d'optimisation



c) Taux de reconnaissance sur la base de validation



d) Taux de reconnaissance sur la base de test

Figure 33 **Diagrammes des quartiles des ensembles obtenus** par les différentes méthodes de recherche.

Pour ce qui est du taux de reconnaissance sur la base de test, 75% des valeurs obtenues par l'AG Simple sont supérieures à la médiane des taux de reconnaissance des autres méthodes. L'AG Simple a de ce fait la meilleure valeur médiane, suivi par *NSGA*-e et *NSGA*-b, ex aequo. La pire valeur obtenue avec l'AG Simple est meilleure que la pire valeur obtenue par toutes les autres méthodes et la dispersion autour de la valeur centrale est faible. L'AG Simple peut donc être considéré comme étant la meilleure méthode pour maximiser le taux de performance et cette dernière est fiable puisqu'il n'y a pas une très grande variation dans les valeurs produites.

Il y a relativement peu de variabilité dans les résultats obtenus (sauf pour la cardinalité des ensembles formés pas la recherche stochastique). C'est *NSGA*-c qui montre la plus grande variabilité de taux de reconnaissance en test avec une plage de près d'un demi pourcent (des valeurs allant de 96,07% à 96,52%). En tenant compte des données aberrantes, c'est la recherche stochastique qui a le moins de variabilité avec une plage de 0,21% suivie de près pas l'AG simple avec une plage de 0,23%.

En plus d'informer sur la variabilité et sur la tendance des valeurs centrales, les diagrammes des quartiles de la figure 33 permettent de visualiser la forme de la distribution des échantillons. Ainsi la distribution des cardinalités obtenues par *NSGA*-e semble symétrique alors que celle de *NSGA*-d ne l'est pas et est composée de deux données aberrantes qui biaiseront le calcul de la variance. En reconnaissance sur la base de validation, seule la distribution de *NSGA*-f est composée d'une donnée aberrante. Mais comme celle-ci se situe à l'extrême gauche (il s'agit de la plus faible performance), aucun des «meilleurs» classifieurs trouvés par chaque méthode ne doit être considéré comme un résultat aberrant ou marginal. Notons au passage que cet **EoC** représenté comme aberrant sur la base de validation est le même qui est considéré aberrant sur la base d'optimisation et sur le graphique des cardinalités. Toujours sur la base de validation, on remarque que toutes les distributions n'ont pas la même symétrie. Plus particulièrement, l'AG Simple et *NSGA*-e ont une distribution qui tend vers la droite, c'est-à-dire vers des taux de reconnaissance

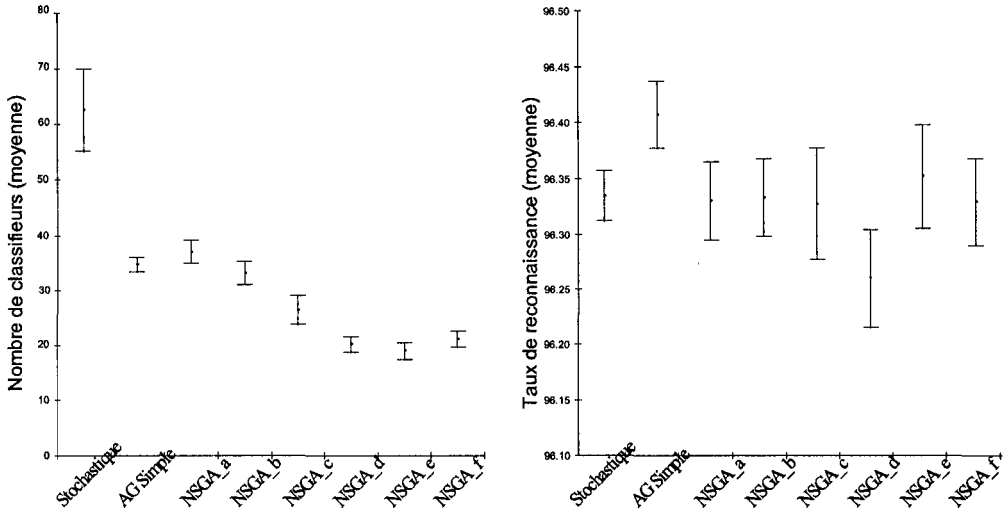
supérieurs. Cette particularité est conservée sur la base de test pour l'AG Simple alors la distribution de *NSGA*-e est plutôt symétrique sur cette dernière base. La signification de cette asymétrie est que la valeur médiane est plus près des valeurs extrêmes élevées que des valeurs extrêmes faibles.

4.2.2 Intervalles de confiance et tests d'hypothèse

La comparaison rigoureuse d'algorithmes de classification demande une attention particulière et la méthodologie pour y parvenir est parfois considérée comme une faiblesse de la communauté de la RF [67]. L'utilisation de plusieurs méthodes de comparaison complémentaires peut aider à mieux discerner les différences entre les groupes d'**EoC** générés et donc de juger des forces et faiblesses des techniques les ayant produits. Les figures 34 et 35 montrent respectivement les moyennes et variances placées dans un intervalle de confiance de 99% pour la cardinalité et le taux de reconnaissance en test. Ce sont finalement ces deux paramètres qui importent puisque les taux de reconnaissances intermédiaires (optimisation et validation) sont biasiés et ne peuvent servir de point de comparaision entre les classifieurs [67]. Les valeurs numériques des intervalles de confiance permettant de tracer ces graphiques sont présentées sous forme de tableaux à l'annexe G.

Ces graphiques nous apprennent qu'il faut être prudent avant d'avancer qu'un algorithme donne en moyenne de meilleurs résultats qu'un autre. Par exemple, pour les moyennes de la cardinalité, on distingue vraiment 4 groupes de moyennes comparables. Si la recherche stochastique est nettement en dehors du lot, les différences entre les autres méthodes est plus subtile (cf figures 34-a). Quant à la figure 34-b), elle montre que les moyennes des taux de reconnaissances partagent en grande partie le même intervalle de confiance, sauf pour l'AG Simple qui a une moyenne nettement supérieure et pour *NSGA*-d qui semble avoir une moyenne vraiment inférieure.

La différence dans les variances est plus difficile à juger qualitativement sur la figure 35. Il est cependant clair que la variance du nombre de classifieurs trouvés par la recherche



a) Nombre de classifieurs

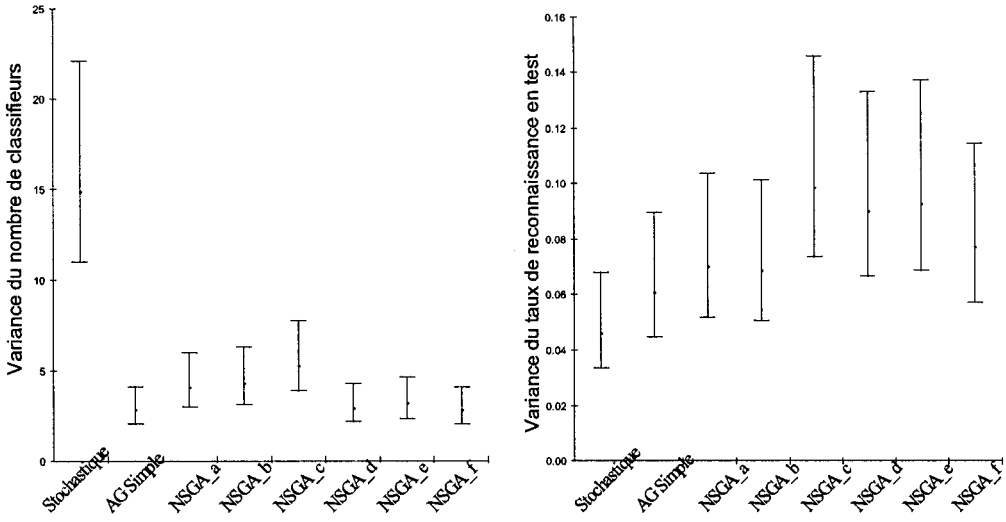
b) Taux de reconnaissance sur la base de test

Figure 34 **Intervalles de confiance des moyennes.** Sur ces graphiques, on peut voir les moyennes (un point) du nombre de classifieurs (a) et du taux de reconnaissance sur la base de test (b) aisni que leur intervalle de confiance (ligne verticale). Les moyennes sont incluses dans cet intervalle 99% du temps [63].

stochastique est très élevée par rapport aux autres (c'était aussi très visible avec les diagrammes des quartiles). Pour mieux juger de leurs différences, des tests d'hypothèses seront nécessaires.

Ces tests viennent ajouter à l'analyse qualitative des diagrammes des quartiles et des diagrammes des intervalles de confiances en fournissant un cadre formel pour juger de l'égalité des moyennes (estimation de la tendance centrale) et des variances (estimation de la dispersion). Pour effectuer le test t de Student et comparer deux moyennes, il faut respecter les conditions suivantes :

1. les échantillons proviennent de populations indépendantes
2. les échantillons proviennent de populations normales
3. les variances des populations sont identiques



a) Nombre de classifieurs

b) Taux de reconnaissance sur la base de test

Figure 35 **Intervalles de confiance des variances.** Sur ces graphiques, on peut voir les variances (un point) du nombre de classifieurs (a) et du taux de reconnaissance sur la base de test (b) ainsi que leur intervalle de confiance (ligne verticale). Les variances sont incluses dans cet intervalle 99% du temps [63].

La première condition devrait être remplie puisque toutes les expériences ont été produites indépendamment les unes des autres. Cependant, l'utilisation d'une base de test commune crée peut-être une certaine dépendance statistique entre les résultats sur cette base. Il n'est donc pas certain que cette condition soir complètement remplie. Néanmoins, l'utilisation d'une base commune de test est indispensable dans notre protocole expérimental pour pouvoir comparer les solutions obtenues. Nous ne comparons pas des ensembles de classifieurs (ou tout autre machine d'apprentissage) mais bien la capacité qu'ont des algorithmes de recherche à trouver des ensembles de classifieurs performants. La mesure de performance doit se faire sur la même base pour être valable. Il est possible qu'une certaine dépendance statistique découle de cette prérogative et donc qu'il y ait un bais dans le résultat des tests t.

Bien que nous ne connaissions pas les distributions des estimateurs (cardinalité et taux de reconnaissance en test) et que nous ne pouvons pas en assurer la normalité, la population de 30 échantillons permet malgré tout d'effectuer les tests t. En effet, le Théorème de la limite centrale nous assure que les moyennes des estimateurs sont distribuées selon une loi normale indépendamment de la distribution de ceux-ci. Le test t devient alors presque équivalent au test Z, qui lui ne requiert pas la normalité des populations [2, 63].

Les graphiques de la figure 35 nous permettent de considérer les variances comme étant égales pour les besoins du test t. En effet, le test tolère facilement un facteur de deux ou trois entre les variances supposées égales [63]. Le cas de la variance de la cardinalité avec la recherche stochastique est plus litigieux puisque sa variance est très largement supérieure aux autres.

Les valeurs numériques des moyennes et des variances à comparer sont reportées aux tableaux XXIV et XXV de l'annexe G. Toutes les moyennes et les variances des résultats en cardinalité et en performance sur la base de test ont été comparées deux à deux afin de déterminer si leurs différences sont significatives (avec un seuil de signification $\alpha = 0,01$). Les tableaux XI et XII montrent le résultat de ces tests. On trouve dans le triangle supérieur les résultats de la comparaison des variances et dans le triangle inférieur les résultats de la comparaison des moyennes. Le seuil de signification de 1% étant arbitraire, le détail des valeurs numériques utilisées pour prendre les décisions et créer ces tableaux est donné à l'annexe H. Notons au passage que tous les résultats de ces tests n'ont pas le même intérêt. En effet, il est plus intéressant de vérifier que les taux de reconnaissance moyens les plus élevés sont significativement plus élevés que les autres que de vérifier si la différence entre deux taux intermédiaires est significative. Néanmoins, toutes les combinaisons de tests ont été effectuées et calculées et tous les résultats sont présentés.

Le test t de Student consiste à rejeter ou non l'hypothèse H_0 selon laquelle deux moyennes $\bar{x_1}$ et $\bar{x_2}$ sont identiques (donc que $\bar{x_1} - \bar{x_2} = 0$). L'hypothèse H_0 est rejetée si elle a un

degré de signification p inférieur à un seuil de signification α. Règle de décision : Si $p < \alpha \Rightarrow$ on rejette H_0 et on considère les moyennes comme étant différentes.

Le test F de Fisher consiste à rejeter ou non l'hypothèse H_0 selon laquelle deux variances σ_1 et σ_2 sont identiques. L'hypothèse H_0 est rejetée si elle a un degré de signification p inférieur à un seuil de signification α. Règle de décision : Si $p < \alpha \Rightarrow$ on rejette H_0 et on considère les variances comme étant différentes.

La probabilité de commettre une erreur de première espèce est donc fixée par le paramètre α. Une erreur de première espèce survient lorsque que l'hypothèse H_0 est rejetée alors qu'elle est vraie. Une erreur de deuxième espèce arrive lorsque H_0 est acceptée alors qu'elle est fausse. Lors de comparaisons multiples, telles que présentées aux tableaux XI et XII, la probabilité qu'une erreur de première espèce se produise dans au moins une comparaison augmente avec le nombre de tests effectués. Des facteurs d'ajustement de la valeur t ou α existent pour pallier à ce problème, la plus populaire étant l'ajustement de Bonferroni [63, 67]. Ces méthodes qui visent à diminuer la probabilité d'erreur de première espèce ne font cependant pas l'unanimité notamment en raison du fait qu'elles ont tendance à augmenter la probabilité d'erreur de deuxième espèce [63]. C'est pourquoi, aucune correction n'a été faite sur les tests d'hypothèses. Il faut cependant garder à l'esprit que si chacune des 28 comparaisons de moyennes de la cardinalité présentées au tableau XI a une probabilité d'erreur de première espèce α, la probabilité qu'une telle erreur se produise au moins une fois est de $1 - (1 - \alpha)^{28}$, c'est-à-dire près d'une chance sur quatre. De la même façon, on peut calculer la probabilité qu'au moins une erreur de première espèce survienne dans l'ensemble des quatre groupes de comparaisons (moyennes et variances de la cardinalité et du taux de performance). On obtient ainsi une probabilité d'environ 68%.

On constate qu'il y a quatre groupes de moyennes de cardinalité. Ainsi, *NSGA*-d, *NSGA*-e et *NSGA*-f ont une moyenne significativement plus petites que les autres méthodes alors

Tableau XI

Résultats des tests t et Fisher pour la cardinalité ($\alpha = 0,01$)

								VARIANCE
	Stoch.	AG S.	*NSGA*-a	*NSGA*-b	*NSGA*-c	*NSGA*-d	*NSGA*-e	*NSGA*-f
Stoch.		S	S	S	S	S	S	S
AG S.	S		NS	NS	S	NS	NS	NS
NSGA-a	S	NS		NS	NS	NS	NS	NS
NSGA-b	S	NS	NS		NS	NS	NS	NS
NSGA-c	S	S	S	S		S	S	S
NSGA-d	S	S	S	S	S		NS	NS
NSGA-e	S	S	S	S	S	NS		NS
NSGA-f	S	S	S	S	S	NS	S	
	MOYENNE							

S : *Différence signifi cative*
NS : *Différence non signifi cative*

que la différence entre leur moyenne respective n'est pas significative (sauf entre *NSGA*-e et *NSGA*-f). Vient ensuite *NSGA*-c, suivi d'un groupe constitué de *NSGA*-a, *NSGA*-b et de AG Simple. Finalement, c'est la recherche stochastique qui a la plus grande moyenne et cette différence est significative. Par contre, au niveau de la variance, il y a peu de différences significatives entre ces échantillons. Seule la recherche stochastique a une variance significativement plus élevée par rapport à toutes les autres méthodes. Mis à part *NSGA*-c qui a aussi une variance élevée par rapport à la plupart des techniques de recherche, on peut considérer que toutes les méthodes restantes ont une variance équivalente.

Pour ce qui est des taux de reconnaissance sur la base de test, deux moyennes ont une tendance significativement différentes des autres. Celle de l'AG Simple est supérieure aux autres alors que celle de *NSGA*-d est inférieure. Les autres méthodes ont une moyenne comparable, même *NSGA*-f qui avait pourtant une moyenne plus faible sur la base d'optimisation. Finalement, seule la recherche stochastique se démarque significativement par une plus faible variance par rapport à plusieurs autres types de recherche.

Tableau XII

Résultats des tests t et Fisher pour la performance en test ($\alpha = 0,01$)

								VARIANCE
	Stoch.	AG S.	*NSGA*-a	*NSGA*-b	*NSGA*-c	*NSGA*-d	*NSGA*-e	*NSGA*-f
Stoch.		NS	NS	NS	S	S	S	S
AG S.	S		NS	NS	NS	NS	NS	NS
NSGA-a	NS	S		NS	NS	NS	NS	NS
NSGA-b	NS	S	NS		NS	NS	NS	NS
NSGA-c	NS	S	NS	NS		NS	NS	NS
NSGA-d	S	S	S	S	S		NS	NS
NSGA-e	NS	S	NS	NS	NS	S		NS
NSGA-f	NS	S	NS	NS	NS	S	NS	
	MOYENNE							

S : *Différence signifi cative*
NS : *Différence non signifi cative*

Bien que l'hypothèse d'égalité entre deux variances ait été rejetée plusieurs fois, on peut considérer les tests t valables en raison de leur tolérance à une certaine différence des variances mentionnées plus haut. Cependant, les tests t comparant la moyenne de la cardinalité de la recherche stochastique aux autres méthodes de recherche le sont pas nécessairement. Par contre, l'étude des diagrammes des quartiles et des intervalles de confiance montrent très clairement que la recherche stochastique a tendance a créer des **EoC** ayant une cardinalité considérablement plus élevée que les autres méthodes. Pour cette raison, nous pouvons croire que cette différence est significative et donc que les résultats des tests t sont valides malgré la violation d'une des conditions du test. Cet exemple montre d'ailleurs l'avantage d'utiliser plusieurs méthodes d'analyse complémentaires pour mieux juger des différences entre différents algorithmes de recherche.

La plupart des tests ont produit un degré de signification p ayant une grande distance au seuil $\alpha = 0,01$ (voir tableaux XXVII et XXVIII de l'annexe H). Il y a cependant quelques cas où la décision de rejeter où non H_0 aurait pu basculer (dans un sens ou dans l'autre) en modifiant légèrement le test : en changeant le seuil de signification pour une valeur

$\alpha = 0,05$ ou en effectuant une correction sur la valeur de de t ou de F par exemple. Ce sont les cas où une erreur de première et de seconde espèce sont le plus susceptibles de se produire. Il s'agit principalement de la cardinalité moyenne de AG Simple versus celle de *NSGA*-a et de celle de *NSGA*-e versus *NSGA*-f. Pour la moyenne du taux de reconnaissance en test, la comparaison de *NSGA*-c avec *NSGA*-d ainsi que celle de *NSGA*-e avec AG Simple ont donné les taux de signification le plus proche du seuil α. Ce dernier cas est le plus intéressant puisque si une erreur de première espèce est effectivement survenue, cela signifie que leur moyenne de performance en test ne sont pas différentes. On pourrait ainsi s'objecter à la conclusion du tableau XV de la section 4.5, à savoir que l'AG Simple est la meilleure méthode dans le cas où l'on ne s'intéresse qu'à maximiser le taux de reconnaissance. On peut cependant, encore une fois, faire appel aux diagrammes de des quartiles pour montrer que même si les deux moyennes sont identiques (ce que rien ne prouve), la distribution de l'AG Simple tend vers la droite (hauts taux de reconnaissance) alors que celle de *NSGA*-e est plutôt symétrique et que 75% des solutions de l'AG Simple sont plus performantes que la médiane des solutions obtenues avec *NSGA*-e.

Finalement, il est possible de faire un test t pour comparer une moyenne à une valeur théorique. Il est particulièrement intéressant de savoir que non seulement *NSGA*-e et AG Simple ont les plus hauts de reconnaissance moyen en test (respectivement 96,35% et 96,41%), mais qu'en plus ces moyennes ont une différence significative avec l'ensemble de référence (100 *k*-**NN**) qui a un taux de 96,28% (avec $\alpha = 0,01$). En effet, le test t entre *NSGA*-e et les 100 *k*-**NN** donne une valeur de $t = 4.29$ pour un niveau de signification $p = 0.0002$. Le résultat du test t entre AG Simple est quant à lui sans appel avec une valeur de $t = 11.56$ pour un niveau de signification $p << 1 \times 10^{-4}$. Le gain en performance par rapport à un ensemble de 100 *k*-**NN** non optimisé est donc statistiquement significatif.

4.3 Le rôle des mesures de diversité

Quatre expériences ont mis en jeu une mesure de diversité dans la sélection de classifieurs, soit *NSGA*-{a,b,c et d}. Ce que les résultats ont montré c'est qu'elles ont toutes permis de créer des **EoC** performants.

Par contre, toutes les méthodes de recherche (avec et sans utilisation de la diversité) ont donné de bons résultats et les quatre expériences donnant les premiers quartiles (Q1) les plus faibles en reconnaissance sur la base de test sont celles utilisant une mesure de diversité. D'ailleurs, aucune de ces méthodes ne se démarque réellement si ce n'est que *NSGA*-d a une médiane de reconnaissance en test relativement faible, plus petite que le premier quartile de toutes les autres méthodes.

Ces observations indiquent que les mesures utilisées ne sont pas des plus appropriées pour faire de la sélection de classifieurs, du moins dans le contexte où elles ont été appliquées. De plus, la minimisation et la maximisation d'une même mesure de diversité, soit l'ambiguïté (*NSGA*-{a et d}) ont donné des résultats similaires. *NSGA*-a donnant des taux de reconnaissance légèrement supérieurs alors que *NSGA*-d est meilleur du point de vue complexité. Si deux objectifs aussi contradictoires donnent des résultats comparables, c'est que ces objectifs n'ont pas une influence très importante dans la recherche de solutions.

4.4 Analyse des erreurs

Jusqu'à maintenant, seule une analyse globale des résultats a été exposée, sans références aux classes. Les matrices de confusion permettent de voir quelles classes sont les plus difficiles à reconnaître et avec quelles autres classes elles sont confondues. Sur les matrices de confusion des tableaux XIII et XIV, les colonnes correspondent aux classes attendues et les lignes aux décisions du classifieur. Les nombres étant exprimés en pourcentage, la somme de chaque colonne fait 100% puisque chacune des observations est assignée à une

Tableau XIII

Matrice de confusion d'un k-**NN** (en pourcentage sur base de test)

	Classes attendues									
	0	1	2	3	4	5	6	7	8	9
0	97,69	0,03	0,05	0,05	0,15	0,00	0,25	0,00	2,90	0,15
1	0,17	95,25	0,12	0,05	0,05	0,08	0,11	0,05	0,94	0,03
2	0,07	1,07	97,22	0,79	0,22	0,23	0,03	0,20	0,23	0,02
3	0,35	0,75	1,49	96,26	0,02	3,31	0,02	0,65	0,65	0,76
4	0,31	0,03	0,03	0,00	95,47	0,10	0,07	1,17	0,75	0,95
5	0,07	0,05	0,04	0,90	0,00	93,93	0,39	0,00	0,72	0,05
6	0,68	1,22	0,25	0,07	1,53	0,31	99,00	0,03	0,56	0,00
7	0,00	0,58	0,32	1,02	0,14	0,16	0,00	94,12	0,21	3,69
8	0,39	0,83	0,39	0,25	0,61	0,97	0,14	0,44	91,41	0,27
9	0,25	0,65	0,05	0,65	1,74	0,58	0,00	3,60	1,48	93,96

Tableau XIV

Matrice de confusion de l'ensemble de 100 k-**NN** (en pourcentage sur base de test)

	Classes attendues									
	0	1	2	3	4	5	6	7	8	9
0	98,12	0,00	0,03	0,03	0,15	0,02	0,14	0,00	4,33	0,20
1	0,18	97,67	0,05	0,08	0,03	0,02	0,17	0,06	0,90	0,06
2	0,07	0,99	98,17	0,60	0,20	0,13	0,00	0,23	0,22	0,03
3	0,17	0,12	0,73	97,38	0,00	4,46	0,00	0,46	0,35	0,53
4	0,19	0,00	0,05	0,02	96,88	0,29	0,09	1,80	0,77	1,29
5	0,05	0,00	0,02	0,28	0,00	93,63	0,44	0,00	0,44	0,02
6	0,71	0,61	0,34	0,03	1,19	0,27	99,02	0,02	0,54	0,00
7	0,00	0,40	0,42	1,15	0,18	0,21	0,00	96,50	0,26	3,33
8	0,31	0,32	0,17	0,15	0,10	0,32	0,15	0,08	90,86	0,17
9	0,18	0,12	0,00	0,25	1,21	0,30	0,00	1,01	1,19	94,27

et une seule classe ; la valeur (i, i) sur la diagonale représente les taux de reconnaissance de la classe i. La figure 36 montre, sous forme d'une matrice de confusion, des exemples mal classés par l'ensemble de 100 k-**NN** sur la base de test.

Selon ces matrices, la classe la plus difficile à reconnaître est le «8», pour lequel un k-**NN** ($k = 1$) a un taux de reconnaissance sur la base de test de 92,41% et l'ensemble de 100 k-**NN** a un taux de 90,86%. C'est aussi la classe plus difficile pour tous les **EoC** dont les matrices de confusions sont présentées à l'annexe I. Elle est le plus souvent confondue

avec le «0». Inversement, la classe ayant le plus haut taux de reconnaissance est le «6», que ce soit pour un *k*-**NN** unique ou pour chacun des **EoC** étudiés.

Les mêmes confusions semblent se produire que ce soit avec un *k*-**NN** unique ou les **EoC** : le «1» et le «7» sont rarement confondus avec le «0» alors que c'est très fréquent pour le «8». Globalement, les matrices de confusion des **EoC** et du *k*-**NN** sont semblables, si ce n'est que les matrices des **EoC** ont tendance à avoir des valeurs plus fortes sur la diagonale (puisque ceux-ci ont un taux de reconnaissances plus élevé). Ces résultats suggèrent que les confusions dépendent des caractéristiques et sont plutôt indépendantes des sous-espaces aléatoires. Des études plus poussées seraient cependant nécessaires pour valider cette hypothèse.

4.5 Méthodes de recherche à privilégier

Bien que les méthodes de recherche impliquant une mesure de diversité permettent de trouver des **EoC** intéressants, aucune d'entre elles n'a montré d'avantage particulier. Par contre, les autres techniques ont des avantages et des inconvénients qui leur sont propres et qui peuvent les rendre plus ou moins intéressantes pour la sélection de *k*-**NN**, selon l'application visée. Ainsi, si la complexité est très importante, on choisira *NSGA*-e alors que si c'est la performance qui prime l'AG Simple semble être la meilleure solution. En pratique, si l'on peut se permettre d'avoir une solution avec une légère perte de performance par rapport à l'ensemble original (et donc encore significativement au-dessus du taux de reconnaissance d'un *k*-**NN** unique), alors l'ordonnancement est une approche qui pourrait être envisagée en raison de son extrême simplicité de mise en oeuvre. De plus, cette méthode permet de fixer *a priori* la complexité désirée. Le tableau XV résume les avantages et inconvénients des différentes méthodes.

Si les performances sur la base d'optimisation comportent quelques différences significatives d'un **EoC** à l'autre, les différences dans les résultats en généralisation (base de test) sont en moyenne très faibles et non significatives sauf pour l'AG Simple (moyenne

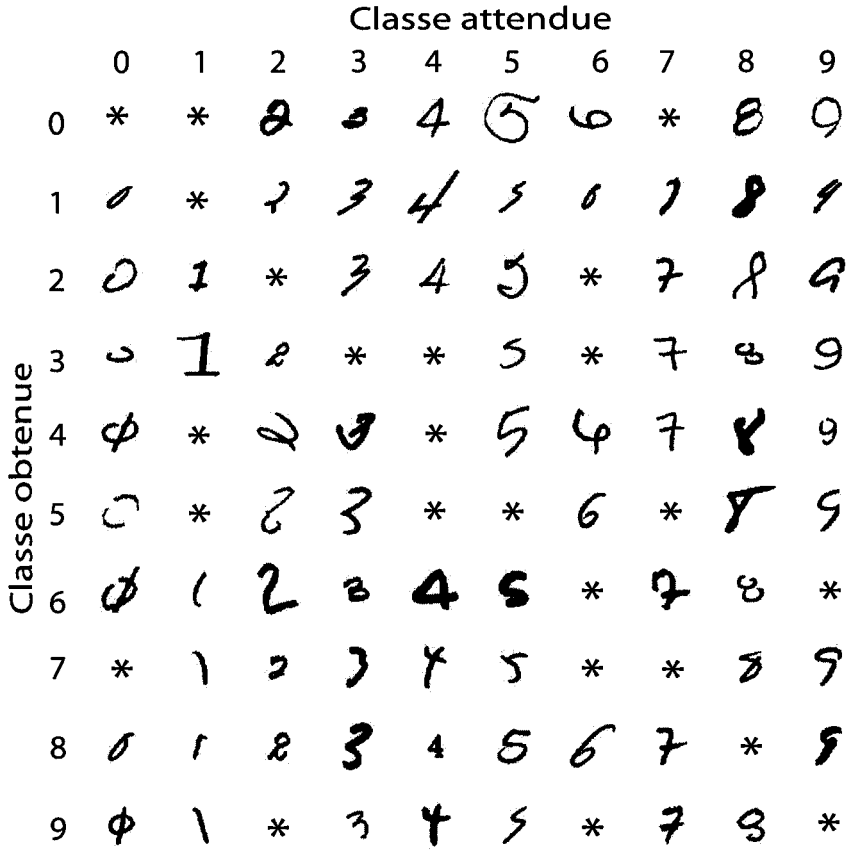


Figure 36 **Exemples de confusion.** Exemples de la base de test qui n'ont pas été reconnus par l'ensemble de 100 *k*-**NN**, présentés sous forme de matrice de confusion. Ainsi, la première colonne montre des exemples de la classe «0» qui ont été mal reconnus. Sur la première ligne ont peut voir des exemples de différentes classes ayant été reconnus comme étant un «0». Les astérisques (*) indiquent qu'il n'y a eu aucune confusion (aucun «7» n'a été confondu avec un «0» et aucun «2» n'a été pris pour un «9».) La diagonale étant les bonnes décisions, elle n'est composée que d'astérisques.

légèrement plus élevée) et pour *NSGA*-c (moyenne légèrement plus faible). Par contre, la différence en complexité (nombre de classifieurs de l'ensemble) passe du simple au triple entre NSGA-e et la méthode d'ordonnancement. Les **EoC** ainsi présentés se distinguent plus par leur complexité que par leur performance.

Tableau XV

Comparaison des avantages et inconvénients des différentes méthodes de recherche

Méthode	Avantage (s)	Inconvénient (s)
Ordonnancement	Très simple à mettre en oeuvre. Déterministe. Choix de la cardinalité.	Légère perte de performance.
Stochastique	Plus faible variabilité dans la performance en test.	Très grande variabilité en complexité.
AG Simple	Meilleure performance.	Complexité relativement élevée.
NSGA-e	Complexité la plus faible. Bonne performance.	
Autre méthodes	Pas de résultat particulier.	

Des expériences complémentaires mettant quelques-uns des meilleurs **EoC** obtenus dans un contexte d'application plus réalistes (système de rejet et caractéristiques absentes) sont présentées au prochain et dernier chapitre. Cette partie du mémoire fait un retour sur les résultats en donnant une interprétation plus générale et conclut le projet en proposant de nouvelles pistes de recherche.

CHAPITRE 5

DISCUSSION ET CONCLUSION

Ce chapitre est consacré à la présentation de quelques expériences complémentaires ainsi qu'à la conclusion générale du projet. Les différentes méthodes de sélection de classifieurs ayant été étudiées dans un contexte bien contrôlé, il était intéressant de conclure en plaçant les **EoC** obtenus dans un contexte d'utilisation plus réel. Deux éléments ont été introduits et étudiés à cette fin : le rejet et la robustesse aux caractéristiques absentes.

5.1 Retour sur les résultats

L'utilisation d'ensembles de *k*-**NN** avec apprentissage par représentation partielle de l'information permet l'augmentation des taux de reconnaissance par rapport à l'utilisation d'un unique *k*-**NN**. On a voulu savoir comment sélectionner un sous-ensemble de ces classifieurs pour former un meilleur **EoC**, c'est-à-dire plus performant et moins complexe. L'hypothèse que l'utilisation d'une mesure de diversité pouvait aider à guider la recherche a été infirmée tandis qu'aucune méthode ne s'est montrée supérieure aux autres de tous points de vue. Les sous-sections qui suivent traitent de ces deux points.

5.1.1 L'utilité de la diversité

Les résultats des expériences montrent que les différentes mesures de diversité testées n'ont pas un impact positif significatif dans l'optimisation d'un **EoC** par sélection de classifieurs. Peut-on en conclure que le concept de diversité est inutile pour autant ?

On sait que par sa nature, l'**EoC** étudié possède déjà une diversité intrinsèque. En effet, un ensemble de *k*-**NN** projeté dans des sous-espaces aléatoires est un ensemble ayant une diversité d'opinions naturelle. Cette diversité aide certainement l'**EoC** à atteindre ses niveaux de performances puisque sans elle, il ne serait pas possible d'avoir des résultats supérieurs à ceux d'un *k*-**NN** unique. Par contre, les résultats suggèrent qu'une fois un

certain niveau de diversité atteint, l'augmentation de celui-ci ne se traduit pas par une augmentation du taux de reconnaissance.

Pour augmenter la diversité, il faut des classifieurs ayant des opinions différentes. Cette diversité d'opinion entraîne nécessairement des erreurs dans les décisions des classifieurs puisque si deux experts ont des opinions divergentes, au moins un des deux doit avoir tort. Cette constatation est vraie quelle que soit la mesure utilisée pour représenter le concept de diversité. L'augmentation de la diversité entraîne donc l'augmentation des erreurs faites par les classifieurs individuellement. Il est probable qu'à partir d'un certain niveau, les gains dus à l'augmentation de la diversité soient annulés par l'augmentation de l'erreur des classifieurs. Cela expliquerait pourquoi l'ensemble de *k*-**NN** projetés dans les sous-espaces aléatoires ne bénéficie pas d'un gain en diversité.

Les expériences menées ne permettent pas d'apporter une conclusion générale sur l'utilité de l'utilisation d'une mesure de diversité pour la sélection de classifieurs. Elles montrent seulement que cette dernière n'est pas utile dans le cadre de la sélection de classifieurs possédant une diversité naturelle comme les *k*-**NN** projetés dans des sous-espaces aléatoires. Le rôle de la diversité dans la création d'**EoC** performants n'est donc pas mis en cause. De plus, le concept de diversité étant mal défini, il est possible que d'autres façons de mesurer celle-ci produisent des résultats différents. Les métriques utilisées dans le projet ne peuvent prétendre englober la totalité du concept de diversité.

5.1.2 Les méthodes de sélection de classifieurs à retenir

Les différentes techniques utilisées pour former des ensembles occupent un large spectre de difficulté, allant de la simple méthode d'ordonnancement aux algorithmes génétiques multi-objectifs, beaucoup plus délicats à mettre en oeuvre. Une fois que la base de données de vote est précalculée, le premier cas ne nécessite qu'un simple tri qui peut s'effectuer sur un ordinateur de poche. Dans le second cas, malgré les précalculs de votes, une grappe de plusieurs ordinateurs fonctionnant en parallèle a été nécessaire. Aussi, le nombre de

paramètres à ajuster pour pouvoir utiliser correctement un AG n'est pas négligeable, spécialement avec des AG multi-objectifs puisque vient s'ajouter le paramètre du rayon de la niche. La lourdeur de la technique est-elle justifiée par un gain substantiel ?

On ne peut répondre que partiellement par l'affirmative. C'est-à-dire que cela dépend beaucoup du contexte de l'application. On peut imaginer que si la performance est critique, chaque fraction de pourcentage de taux de reconnaissance pouvant être gagné doit l'être, peu importe le coût computationnel occasionné. On choisira alors un AG Simple maximisant la performance de l'ensemble. Si en plus on a une forte contrainte quant à la complexité du système, le coût additionnel d'un AG multi-objectif minimisant conjointement le taux d'erreur et le nombre de classifieurs sera justifié. C'est en effet la méthode qui permet d'obtenir les ensembles les plus petits, ensembles qui ont de très hauts taux de performance.

Par contre, dans un contexte d'application où le taux de reconnaissance n'est pas critique à une fraction de pourcentage près, la méthode d'ordonnancement des classifieurs demeure la plus simple à mettre en oeuvre et la plus souple puisqu'elle permet le choix d'une cardinalité d'ensemble arbitraire. Les ensembles ainsi formés, bien que légèrement moins performants que l'ensemble original, apportent tout de même un gain substantiel en taux de reconnaissance par rapport à un *k*-**NN** simple.

5.2 Stratégies de rejet

Lors de la mise en oeuvre d'un système de reconnaissance de formes, on s'attend généralement à ce qu'il y ait un système de rejet. De cette façon, on peut fixer un certain taux de confiance lors de la classification des données et rejeter vers un autre système (ou un opérateur humain) celles dont le taux de confiance n'atteint pas un seuil minimal. Il est ainsi possible d'ajuster le système pour avoir un taux d'erreur maximal. Nous avons voulu vérifier si, dans notre contexte d'ensemble de *k*-**NN**, on pouvait instaurer une manière de mesurer ce taux de confiance et fixer une règle de rejet.

Pour tester le comportement des courbes erreur-rejet, nous avons choisi un échantillon des meilleurs **EoC** obtenus par les méthodes d'optimisations exposées précédemment. Chaque point d'une courbe erreur-rejet se construit en fixant un seuil de confiance en deçà duquel les observations sont rejetées. Lorsque ce seuil est nul, on ne rejette aucune observation et on parle de l'erreur à zéro rejet (E_0). Lorsque ce seuil augmente, l'erreur diminue jusqu'à atteindre une valeur nulle[1]. Une courbe erreur-rejet montre donc la diminution du taux d'erreur d'un classifieur en fonction du taux de rejet. On construit généralement ces courbes une première fois sur une base d'optimisation pour déterminer à quelle valeur de seuil correspond un taux d'erreur donné. On peut ensuite mettre en oeuvre le classifieur en fixant le taux d'erreur désiré du système grâce au seuil ainsi trouvé.

La décision de rejeter ou non une observation dépend d'un taux de confiance qui est représenté par la probabilité *a posteriori* d'appartenance d'une observation à une classe. Deux méthodes pour calculer $P(C_i|x)$ dans un contexte d'ensembles de classifieurs à votes coexistent dans la littérature, celles de Xu [76] et de Hansen [29] (voir annexe E). La première méthode consiste à calculer des probabilités d'erreurs en fonction des matrices de confusion sur une base d'apprentissage pour chaque classifieur de l'ensemble et à combiner ces probabilités pour estimer $P(C_i|x)$. La seconde méthode d'estimation est beaucoup plus simple puisqu'elle n'est autre que le rapport entre le nombre de classifieurs ayant voté pour une classe et le nombre total de classifieurs de l'ensemble.

Les deux méthodes d'estimation ont été comparées et elles sont apparues équivalentes dans le contexte de notre problème (cf figure 37). C'est-à-dire que pour un taux de rejet donné, il n'y a pas une méthode qui a un taux d'erreur minimum. Dans ce cas, il est préférable d'utiliser la méthode la plus simple et la moins coûteuse en terme de calcul, c'est-à-dire la méthode de Hansen.

[1] Il est toujours possible de créer un classifieur qui ne fasse pas d'erreur, il suffit de rejeter 100% des observations. Notons que si un tel classifieur à un taux d'erreur nul, il n'est pas d'une très grande utilité puisque sa performance (taux de reconnaissance) est nulle elle aussi.

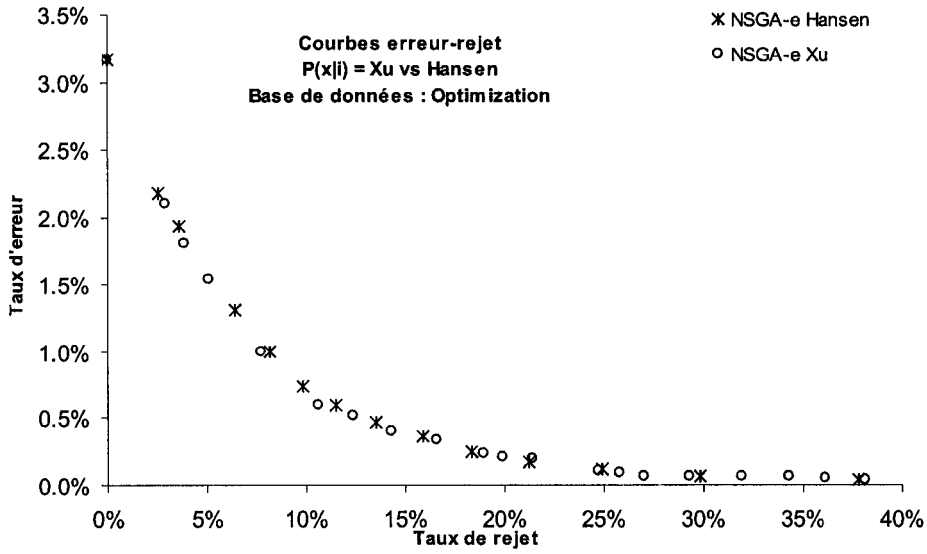


Figure 37 **Comparaison des méthodes de Xu et de Hansen pour estimer $P(C_i|x)$.** Les courbes erreur-rejet se chevauchent pour les deux méthodes d'estimation des probabilités *a posteriori*.

La façon de calculer $P(C_i|x)$ déterminée, un ensemble d'**EoC** a été choisi pour montrer leur comportement sur les courbes erreur-rejet (cf figure 38). On remarque que tous les **EoC** ont un comportement très proche les uns des autres et que leur profil de courbe erreur-rejet sont presque identiques. L'échelle choisie pour présenter ces courbes relativise d'ailleurs les différences entre les **EoC**.

La méthode de rejet en fixant un seuil à partir d'une estimation de probabilité *a posteriori* a été proposée pour la première fois par Chow au début des années 70 [12]. Aujourd'hui, des auteurs comme Fumera proposent plutôt d'utiliser un seuil différent pour chacune des classes puisque chacune de celles-ci ne présentent pas le même niveau de difficulté [25]. En variant la tolérance du seuil de confiance selon la difficulté de chaque classe, on peut normalement diminuer le taux global de rejet tout en maintenant le taux d'erreur. Il conviendrait donc d'étudier l'implantation d'un tel système de rejet lors de la mise en oeuvre d'un **EoC** si l'application exige d'optimiser les courbes erreur-rejet.

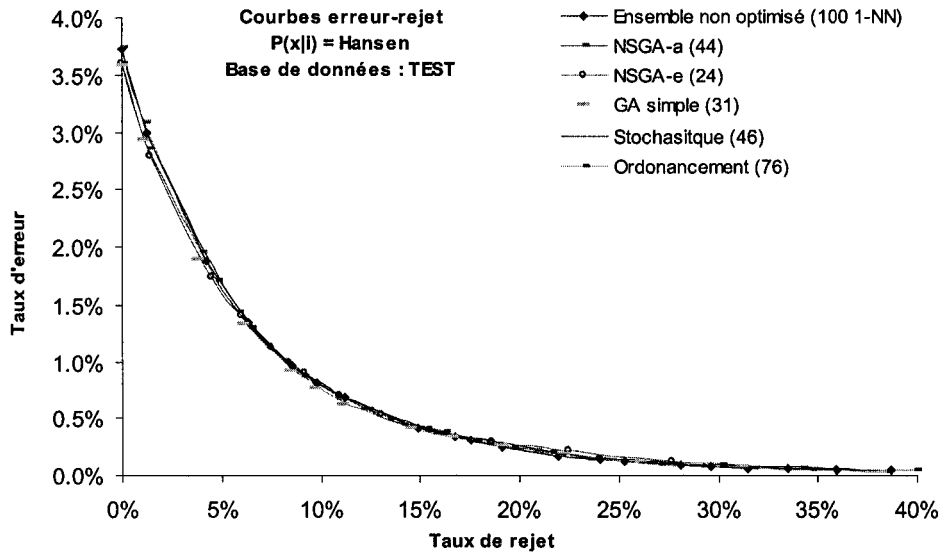


Figure 38 **Courbes d'erreur-rejet obtenues par différents EoC sur la base de test.** L'ensemble des **EoC** étudiés ont le même profil de courbe erreur-rejet sur la base de test. Il faut par exemple rejeter environ 7% des exemples pour diminuer l'erreur à 1% quel que soit l'**EoC** utilisé.

5.3 Robustesse aux caractéristiques absentes

Une caractéristique est dite absente lorsqu'elle n'est pas disponible au moment de la classification. C'est un problème réel qui peut survenir lorsqu'un système de reconnaissance de formes est mis en application. Lorsqu'une ou plusieurs caractéristiques ne sont pas disponibles, on peut soit rejeter la donnée à classer, soit utiliser une stratégie pour tenter de la classer en dépit du problème de représentation que cela peut poser. Les sous-espaces aléatoires étant déjà une sorte d'espace aux caractéristiques absentes, on a voulu vérifier leur robustesse à ce bruit d'entrée. Quatre classifieurs ou ensembles de classifieurs ont été retenus pour l'expérience.

1. Un seul 1-NN dans l'espace de représentation complet

2. L'ensemble non optimisé de 100 1-NN chacun utilisant une représentation à 32 caractéristiques
3. Un ensemble de 31 1-NN trouvé avec un AG Simple
4. Un ensemble de 24 1-NN trouvé avec *NSGA*-e

Ces deux derniers ensembles sont les plus performants sur la base de validation générés après 30 itérations de leur algorithme d'optimisation respectif.

L'expérience consiste à mesurer le taux de reconnaissance de ces différents classifieurs pour différents taux de caractéristiques absentes (CA). Pour chacune des données à classer, un nombre déterminé de caractéristiques sont considérées absentes, i.e. : elle ne sont pas disponible pour la classification. Les caractéristiques absentes sont tirées de façon aléatoire selon une loi de distribution uniforme parmi les 132 caractéristiques normalement disponibles. Un nouvel ensemble de CA est tiré pour chaque donnée à classer.

Trois stratégies pour faire face aux CA ont été comparées :

1. **Valeur moyenne** La valeur moyenne consiste à remplacer la valeur de la caractéristique absente par sa valeur moyenne telle que calculée sur la base d'apprentissage.
2. **Retrait** Le retrait consiste à retirer la caractéristique absente de l'ensemble des caractéristiques, projetant ainsi les classifieurs affectés dans des nouveaux sous-espaces plus petits.
3. **Méthode des Usable Classifiers (UC)** Proposée par Krause [43], la méthode des UC ne retient pour faire la classification que l'ensemble des classifieurs n'utilisant pas les CA dans leur représentation du problème. Selon la méthode originellement proposée, on doit créer de nouveaux classifieurs pour remplacer ceux qui ne sont pas utilisés et ainsi maintenir un certain taux de performance. Dans notre cas, puisque le nombre de classifieurs a été fixé à 100 au départ, il n'est pas possible d'en créer de nouveaux. Les classifieurs mis de côté ne sont donc pas remplacés.

Les expériences ont été faites sur la base de validation et sur la base de test ; les classifieurs ont été entraînés avec la base d'apprentissage de 5000 exemples (***TRAIN5k***).

La performance est définie comme étant le rapport entre le nombre d'exemples reconnus et le nombre d'exemples présentés. Pour pouvoir comparer les différents classifieurs entre eux, la perte de performance (exprimée en dB) a aussi été mesurée. Cette perte, mesurée par rapport à la performance sans CA ($Perf_0$) se calcule ainsi :

$$\text{Perte de performance} = 10\log(\text{Performance}/\text{Perf}_0) \tag{5.1}$$

On peut ainsi comparer le taux de reconnaissance absolu des différents classifieurs en fonction du taux de CA et aussi comparer leur robustesse, c'est à dire leur capacité à maintenir leur performance initiale.

5.3.1 Résultats des expériences sur les CA

La première constatation est que la méthode inspirée par Krause est impraticable dans notre étude puisque le nombre de classifieurs disponibles pour la classification diminue trop rapidement avec l'augmentation du taux de CA. En effet, la méthode des UC proposée originellement met de côté les classifieurs utilisant des caractéristiques parmi les CA et les remplace par d'autres créés ad hoc afin de conserver la cardinalité de l'ensemble de départ. Dans le cadre de notre étude, l'ensemble des classifieurs disponibles est au nombre de 100 et il n'y a pas de création, seulement de la sélection de classifieurs. Le graphique de la figure 39 montre la rapidité de la baisse du nombre de classifieurs disponibles. Avec aussi peu que 6% de CA, il ne reste en moyenne que 13 classifieurs alors qu'il y en avait 100 au départ (ensemble 100 *k*-**NN**). Très rapidement (vers 10% de CA), on se retrouve avec un ensemble vide de classifieurs disponibles, ce qui interdit toute classification. Ce phénomène est évidemment accentué pour les **EoC** qui ont une cardinalité de départ inférieure à 100, puisqu'il suffit de moins de 3% de CA pour diviser par deux le nombre de classifieurs disponibles. Si le système avait eu plus de classifieurs au départ, disons 1000 à la place de 100, la perte de 50% de classifieurs signifierait tout de même la conservation

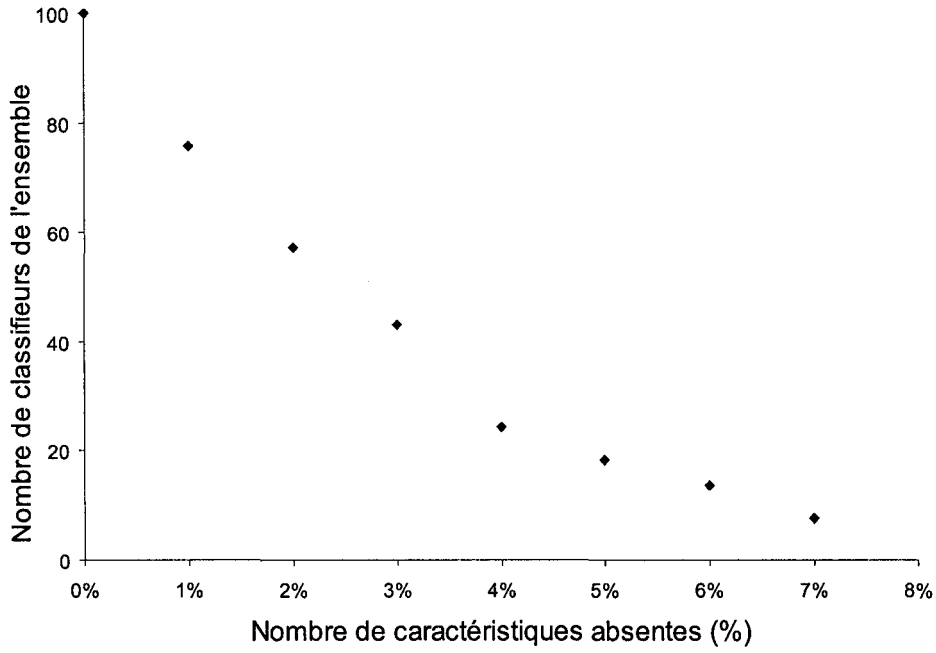


Figure 39 **Évolution du nombre de classifieurs avec la méthode de Krause.** Le nombre de classifieurs utilisables diminue rapidement avec l'augmentation du nombre de caractéristiques absentes. Avec un taux de CA de seulement 2,5%, déjà la moitié des classifieurs ne sont plus disponibles. Avec 7% de CA, il reste en moyenne moins de 8 classifieurs disponibles.

de 500 UC. Cependant, si un système nécessite 1000 classifieurs pour fonctionner correctement et que la moitié n'est pas utilisable dès 3% de CA, le problème reste identique. Si avec près de 10% de CA, un certain nombre de classifieurs sont encore disponibles (ce qui n'est pas certain), ce nombre diminuera tout de même très rapidement avec l'augmentatin de CA. La méthode des UC est donc mal adaptée au contexte de la sélection de classifieurs. Par contre, on pourrait imaginer des adaptations qui la rendrait praticable. Par exemple, plutôt que d'ignorer un classifieur dès qu'une seule caractéristique est absente, la méthode fonctionnerait sans doute mieux si elle permettait un certain taux de CA. Ainsi, seuls les classifieurs dépassant ce seuil seraient mis de côté.

La seconde constatation est que la stratégie de retrait est supérieure à celle de la valeur moyenne et que cette différence s'amplifie à mesure que le taux de CA augmente comme le montre la figure 40.

Finalement, il reste à comparer les différents classifieurs entre eux. L'utilisation de la valeur moyenne ne donnant pas de bons résultats, les comparaisons de la figure 41 n'utilisent que la méthode de retrait des CA. Les classifieurs optimisés[2], qui ont au départ une performance supérieure, maintiennent leur avantage jusqu'à environ 70% de CA (cf figure 41-a). À partir de ce seuil, le *k*-**NN** simple est plus avantageux que les ensembles optimisés. La figure 41-b montre la perte de performance, des différents classifieurs en fonction du taux de CA. Cette diminution est plus rapide pour les ensembles optimisés que pour le *k*-**NN** simple.

5.3.2 Conclusions sur la robustesse aux CA

Les ensembles optimisés sont moins robustes aux CA que le *k*-**NN**, c'est-à-dire que leur performance diminue plus rapidement avec l'augmentation du taux de CA. Par contre, le fait que les ensembles optimisés soient plus performants lorsqu'il n'y a pas de CA, leur confère un avantage puisqu'ils restent plus performants, dans l'absolu, que le *k*-**NN**. Cette supériorité est valable jusqu'aux cas extrêmes où il y a plus de 70% de CA.

La robustesse du *k*-**NN** s'explique par le même argument que la supériorité de la méthode de retrait par rapport au remplacement par la valeur moyenne. Dans les expériences préliminaires, on a mesuré qu'un ensemble de *k*-**NN** dans des sous-espaces aléatoires de cardinalité 16 avaient à peu près la même performance qu'un ensemble dont les composantes sont projetées dans un sous-espace de cardinalité 32. L'élimination de caractéristiques revient à projeter dans un sous-espace de plus petite cardinalité alors que le remplacement

[2] Le terme *ensembles optimisés* fait référence aux **EoC** produit après sélection de classifi eurs par une méthode d'optimisation (*NSGA*-e, AG Simple, etc.). Il n'ont pas été optimisés pour faire face aux caractéristiques absentes.

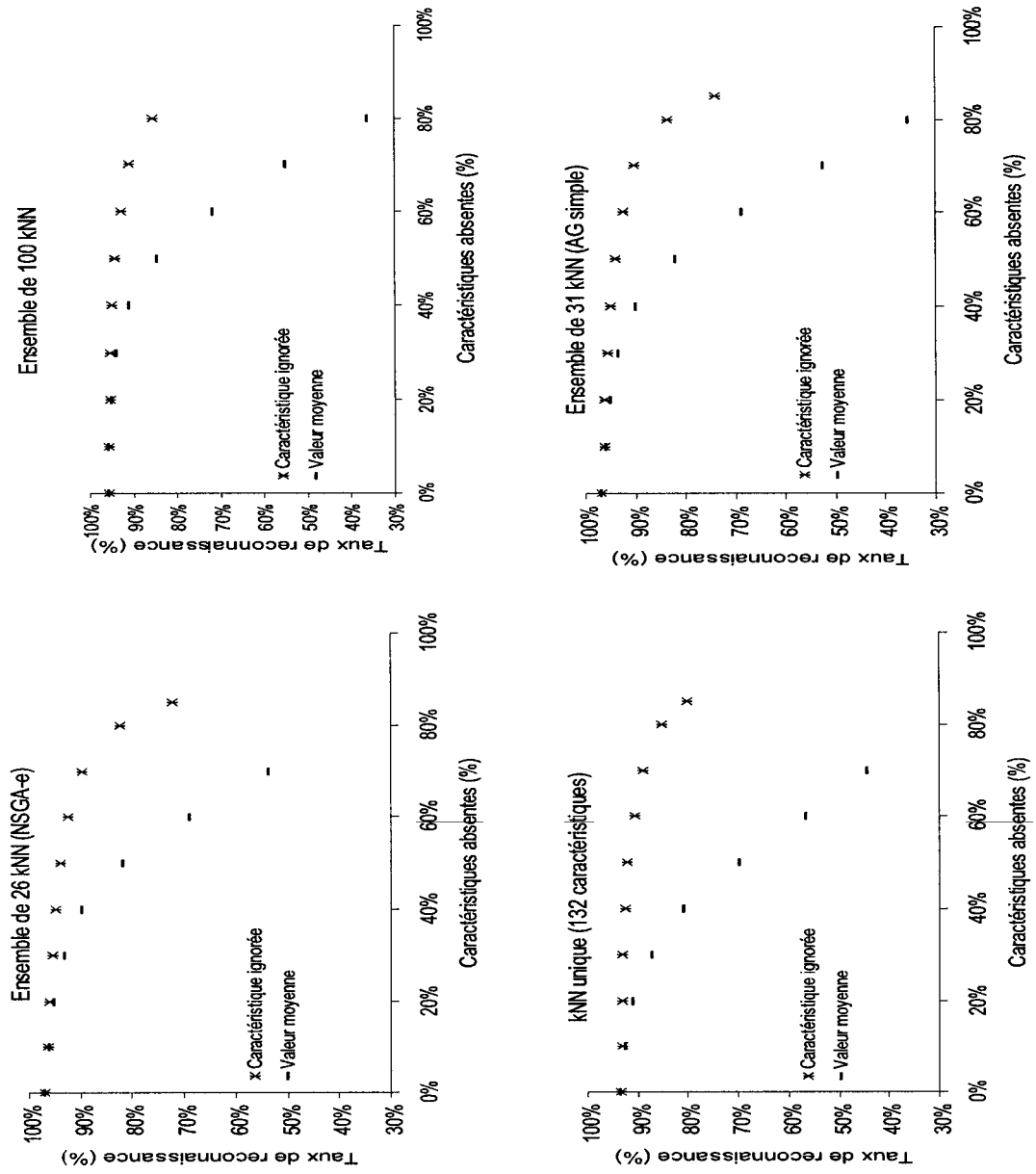
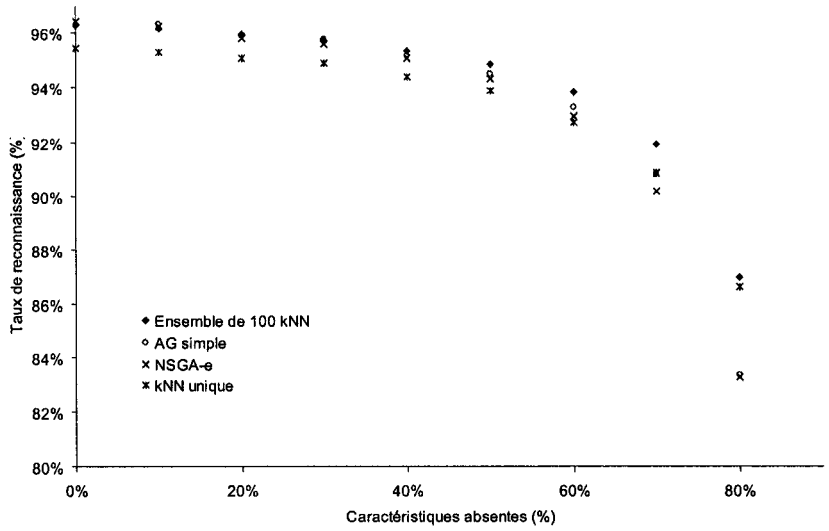


Figure 40 **Évolution des taux de reconnaissance avec le nombre de caractéristiques absentes** (sur la base de validation). Quel que soit l'**EoC** étudié, ignorer les caractéristiques absentes est plus efficace que de les remplacer par leur valeur moyenne et plus le taux de CA augmente, plus la différence entre les deux stratégies s'accentue. En ignorant les caractéristiques absentes, les **EoC** maintiennent des taux de reconnaissance très élevés même en situation extrême : taux de reconnaissance haut-dessus de 80% avec 80% de CA.

a)



b)

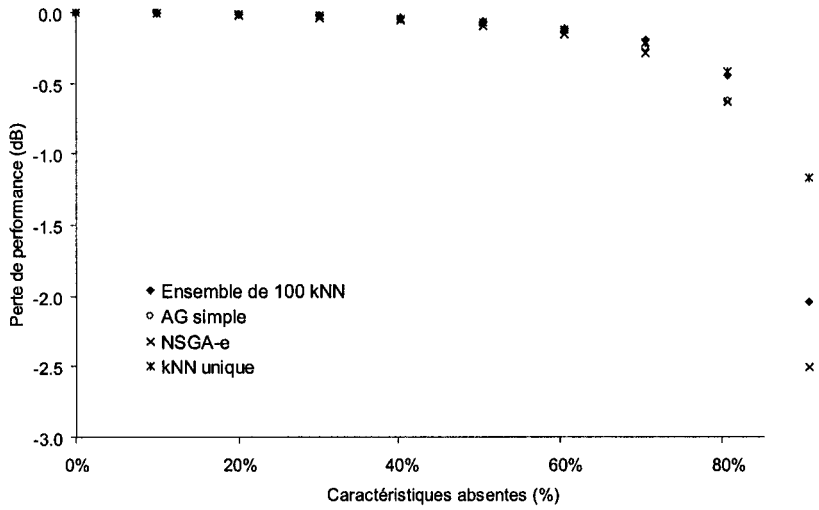


Figure 41 **Perte de performance en fonction du taux de caractéristiques absentes.** a) Les **EoC** plus performants le demeurent jusqu'à des taux de CA élevés (autour de 60% ou 70%), après quoi ils se font rattraper. b) La perte de performance est plus lente pour le *k*-**NN** simple et l'ensemble non optimisé de 100 *k*-**NN**. Raison pour laquelle ils finissent par rattraper en performance les autres **EoC**.

par la valeur moyenne ajoute de l'information qui, vraisemblablement, n'aide pas à la reconnaissance.

De plus, en étudiant les 100 *k*-**NN** de cardinalité 32 générés, on s'aperçoit que le *k*-**NN** le moins performant de l'ensemble a un taux de reconnaissance de près de 83% et que le taux moyen est de 87%. On s'attend donc à ce que le *k*-**NN** simple, avec une cardinalité de 132, ait un taux de reconnaissance de cet ordre de grandeur lorsque 75% des caractéristiques sont absentes (25% x 132 caractéristiques = 33 caractéristiques). C'est ce que l'on trouve puisqu'à 70% de CA, le *k*-**NN** conserve un taux de reconnaissance de 89% et un de 85% lorsque 80% des caractéristiques sont absentes.

On aura remarqué que l'ensemble créé avec l'AG Simple est plus robuste que l'ensemble résultant de la recherche avec *NSGA*-e et que ce dernier a une cardinalité inférieure (24 contre 31). Ces deux ensembles sont eux-même moins robustes que l'ensemble de 100 *k*-**NN**. Il semblerait que la taille de l'ensemble ait un impact sur la robustesse aux CA. Mais il est aussi vrai que ces différences entre les performances sont faibles pour la plupart des taux de CA. En effet, ce n'est qu'à partir du cas extrême où 50% ou 60% des caractéristiques sont absentes que l'écart entre les performances ou les robustesses se remarque. Ce fait indique que c'est peut-être le grand nombre de caractéristiques qui joue le plus grand rôle dans la robustesse aux CA. Puisque ces caractéristiques comportent une redondance assez forte [55], les classifieurs sont moins sensibles à l'absence d'une grande proportion de celles-ci.

5.4 Conclusion

Dans le présent travail, nous avons tenté de vérifier s'il était possible de concevoir des ensembles de classifieurs non paramétriques par sélection de classifieurs pour en diminuer la complexité tout en améliorant la performance du système.

Nous avons découvert que la minimisation conjointe de la complexité et de l'erreur était la façon de créer un ensemble optimisant le mieux ces deux objectifs (division du nombre de classifieurs par 5 en moyenne avec une légère augmentation du taux de reconnaissance). Pour concevoir un ensemble ayant une performance maximale, il vaut mieux utiliser un algorithme de recherche à un seul objectif (augmentation du taux de reconnaissance supérieure mais division du nombre de classifieurs par seulement 3). Contrairement à nos attentes, il n'a pas été possible de démontrer un avantage significatif à l'utilisation d'une mesure de diversité comme critère d'optimisation. À notre connaissance, c'était la première fois qu'était étudiée de manière exhaustive la façon de faire de la sélection de classifieurs de type *k*-**NN** basé sur le paradigme des sous-espaces aléatoires. L'application systématique de tests statistiques pour valider les résultats des stratégies de sélection de classifieurs a été rendu possible grâce à l'utilisation d'une grappe d'ordinateur et à la création de base de données de votes précalculés. Cette validation statistique est rarement mise en oeuvre dans le domaine.

Cependant, malgré la grande quantité d'expériences et de temps de calcul effectué dans le projet, plusieurs aspects des différentes questions n'ont été abordées que superficiellement et des travaux supplémentaires seront nécessaires pour affiner les connaissances du domaine.

Par exemple, qu'arriverait-il si on maximisait uniquement la diversité ? Est-ce qu'au-delà d'un certain seuil quantifiable et prévisible le taux de reconnaissance diminuerait ? Ou s'il ne diminue pas, atteindrait-il un plateau ? Si oui, y aurait-il une façon de déterminer *a priori* la valeur de diversité optimale pour un **EoC** donné (et une mesure de diversité donnée) ?

Peut-être qu'une analyse plus fine de la distribution des votes permettrait de définir une mesure de diversité plus riche qu'un simple scalaire à maximiser. De plus, une telle analyse pourrait être utile dans l'étude de fonctions de fusion plus complexes que le vote à majorité

simple. On a par exemple remarqué que l'ambiguïté d'une décision était généralement très faible lorsque l'ensemble prend la bonne décision et qu'elle est généralement très élevée dans le cas contraire. L'ambiguïté, si elle ne s'est pas avérée utile dans l'optimisation d'**EoC** pourrait peut-être apporter une information importante au niveau de la fusion des décisions.

Puisque chaque classifieur projeté dans un sous-espace est aussi un sous-ensemble de caractéristiques, la sélection de classifieurs s'apparente à la sélection de caractéristiques. Une première analyse a montré que si les caractéristiques de l'ensemble de départ suivaient une loi de distribution discrète uniforme, alors les caractéristiques des ensembles optimisés suivaient la même loi. Ce fut du moins le cas pour les ensembles obtenus par nos méthodes de recherche. Il serait peut-être intéressant de projeter les classifieurs de l'ensemble de départ, non pas dans des sous-espaces aléatoires mais plutôt dans des sous-espaces optimisant certains critères comme la redondance de certaine caractéristiques de façon à obtenir une distribution de caractéristiques non uniforme maximisant le taux de reconnaissance de l'ensemble ou la robustesse à l'absence de certaine caractéristiques que l'on sait plus vulnérables. De même, pour des applications dont on sait *a priori* qu'elles risquent de comporter un fort taux de caractéristiques absentes, il faudrait étudier la possibilité de faire la sélection de classifieurs en maximisant conjointement la performance et la robustesse aux CA.

Lorsqu'une donnée est rejetée par un premier système, elle est souvent redirigée vers un second système et ultimement vers un opérateur humain. Cet opérateur devient en quelque sorte partie intégrante du système de reconnaissance de formes. La place de l'humain ne se limite pas à la phase ultime du processus de RF. En effet, celui-ci peut intervenir à toutes les étapes et interagir avec l'automate de reconnaissance. C'est notamment le cas lorsque l'environnement dans lequel le système de RF évolue est dynamique. L'expert humain peut alors réajuster certains paramètres au niveau du système de création d'**EoC** ou à celui de l'**EoC** lui-même. De nouvelles tendances de recherche étudient justement les

EoC dans un contexte changeant [47] bien que l'humain ne soit généralement pas pris en compte dans la boucle. Après des années de recherche dans l'automatisation des systèmes de RF, le temps est peut-être venu d'y intégrer l'opérateur *homo sapiens* et, pourquoi pas, d'optimiser les systèmes en fonction des compétences particulières de ce dernier.

ANNEXE A

DÉMONSTRATION DU THÉORÈME DU JURY CONDORCET

Cette annexe expose une démonstration du théorème du Jury de Condorcet. Dans une première partie, un rappel des probabilités est présenté suivi de la présentation des inégalités de Markov et de Chebyshev. La seconde partie utilise ces outils pour démontrer le théorème de Condorcet.

Fondements théoriques

Rappels sur les probabilités [75]. $f(x)$ est une probabilité si :

$$f(x) \geq 0 \tag{A.1}$$

$$\int_{-\infty}^{\infty} f(x) = 1 \tag{A.2}$$

$$P(x \leq a) = \int_{-\infty}^{a} f(x)dx \tag{A.3}$$

$$P(x \geq a) = 1 - P(x \leq a) = \int_{a}^{\infty} f(x)dx \tag{A.4}$$

Définition de la moyenne, la variance et des moments.

Moments : Le moment r de x est définit comme suit :

$$\xi_r \equiv E[x^r] = \int_{-\infty}^{\infty} x^r \cdot f(x)dx \qquad \text{avec } r \in \mathbb{N} \tag{A.5}$$

Le moment d'ordre 1 ξ_1 est la moyenne μ :

$$\mu \equiv \langle x \rangle = E[x] = \int_{-\infty}^{\infty} x \cdot f(x)dx \tag{A.6}$$

Moments centrés :

$$m_r \equiv E[(x-\mu)^r] \qquad \text{avec } r \in \mathbb{N} \tag{A.7}$$

Le moment centré d'ordre 2 est appelé variance de x :

$$\sigma^2 = \int_{-\infty}^{\infty} (x-\mu)^2 \cdot f(x)dx \Longleftrightarrow \sigma^2 = E[(x-\mu)^2] = E[x^2] - \mu^2 \tag{A.8}$$

Espérance :

$E[x] = \mu$

$E[(x-\mu)] = \sigma^2 = E[x^2] - \mu$

Inégalité de Markov

$$P(x \geq a) \leq \frac{\langle x \rangle}{a} \qquad \text{avec } x \geq 0 \tag{A.9}$$

Démonstration.

$$\langle x \rangle = \int_0^{\infty} x \cdot f(x)dx = \int_0^{a} x \cdot f(x)dx + \int_a^{\infty} x \cdot f(x)dx \tag{A.10}$$

$P(x)$ =densité de probabilité

Puisque $x \geq 0$

$$\int_0^{a} x \cdot f(x)dx \geq 0$$

et

$$\int_a^{\infty} x \cdot f(x)dx \geq 0$$

$$\Rightarrow \langle x \rangle \geq \int_a^{\infty} x \cdot f(x)dx \tag{A.11}$$

Dans l'intégrale, $x \in [a, \infty[\Rightarrow x \geq a$

$$\Rightarrow \int_a^\infty x \cdot f(x)dx \geq \int_a^\infty a \cdot f(x)dx = a \cdot \underbrace{\int_a^\infty f(x)dx}_{P(x \geq a)} \tag{A.12}$$

$$\Leftrightarrow \langle x \rangle \geq a \cdot P(x \geq a)$$

$$\Leftrightarrow P(x \geq a) \leq \frac{\langle x \rangle}{a} \rightarrow q.e.d.$$

Inégalité de Chebyshev

$$P(|x - \mu| \geq k\sigma) \leq \frac{1}{k^2} \qquad \forall k \geq 0 \tag{A.13}$$

Démonstration.

Dans l'inégalité de Markov, on redéfinit :

$$\sigma^2 \equiv \int_{-\infty}^\infty (x - \mu)^2 \cdot f(x)dx \geq \int_\delta^\infty (x - \mu)^2 \cdot f(x)dx = \int_{|x-\mu| \geq \delta} (x - \mu)^2 \cdot f(x)dx$$

$$\geq \delta^2 \int_{|x-\mu| \geq \delta} f(x)dx$$

$$= \delta^2 P[(x - \mu) \geq \delta]$$

ou, en définissant $\delta \equiv k\sigma$, où k est une constante

$$P[(x - \mu) \geq k\sigma] \leq \frac{1}{k^2} \rightarrow q.e.d.$$

Théorème de Condorcet

Théorème A.0.1 *Soit un jury de n personnes. Chacune a une opinion : vraie ou fausse. On note k le nombre de jury ayant une opinion vraie. Si p est la probabilité d'erreur de chaque juré (identique pour tous) et si $F(n)$ est la probabilité qu'une majorité de jurés aient la bonne opinion (donc que $k > \frac{n}{2}$), alors le Théorème du jury de Condorcet dit que :*

$$Si\ p < \frac{1}{2},\ alors\ \lim_{n\to\infty} F(n) = 1$$

En d'autres termes, la probabilité que le jury produise une majorité de bonnes opinions tend vers 1 quand le nombre d'experts dans le comité tend vers l'infini si chaque expert a une probabilité de décision supérieure au hasard ($\frac{1}{2}$). On suppose des erreurs indépendantes.

Démonstration

La probabilité que exactement k jurés sur n se trompent vaut :

$$\binom{n}{k} p^k \cdot q^{n-k} \qquad avec \binom{n}{k} = \frac{n!}{k!(n-k)!}$$

avec $q = 1 - p$ représentant la compétence.

Soit F(n) la probabilité qu'une majorité de jurés aient la bonne opinion (donc $k > \frac{n}{2}$)

$$F(n) = \sum_{k \leq \frac{n}{2}} \binom{n}{k} p^k \cdot q^{n-k}$$

On choisit la variable aléatoire v_i, $i = 1, 2, ..., n$, pour le juré i telle que :

$$v_i = \begin{cases} 1 & \text{si le juré à raison, avec probabilité q} \\ 0 & \text{si le juré à tort, avec probabilité p} \end{cases}$$

$$\langle v_i \rangle = \sum v_i \cdot f(v_i) = 1 \cdot q + 0 \cdot p = q$$

$$VAR(v_i) = \sigma_{v_i} = (v_i - q)^2 = pq$$

Soit $V = \sum_{i=1}^{n} v_i$.

$$\langle V \rangle = \sum_{i=1}^{n} V \cdot f(v) = \sum_{i=1}^{n} \underbrace{\left(\sum_{i=1}^{n} v_i f(v_i) \right)}_{\langle v_i \rangle = q} = \sum_{i=1}^{n} q = nq = \mu$$

$$VAR(V) = \sum_{i=1}^{n} (V - \langle V \rangle)^2 \cdot f(V)$$

Dans le cas de variables indépendantes,

$$VAR(\sum) = \sum (VAR)$$

$$\Rightarrow VAR(V) = VAR\left(\sum_{i=1}^{n} v_i \right) = \sum_{i=1}^{n} VAR(v_i) = npq = \sigma^2$$

Or : $p < \frac{1}{2} \Rightarrow q > \frac{1}{2}$, donc $nq > \frac{n}{2} \Leftrightarrow \mu > \frac{n}{2}$

Probabilité d'un mauvaise décision majoritaire :

$$P\left(V < \frac{n}{2}\right) = 1 - F(n)$$

$$P\left(V < \frac{n}{2}\right) \leq P\left(V \leq \frac{n}{2}\right) = P\left(-V \geq -\frac{n}{2}\right) = P\left(\mu - V \geq \mu - \frac{n}{2}\right)$$

$$\mu > \frac{n}{2} \Rightarrow \mu - \frac{n}{2} > 0 \Rightarrow \mu - V > 0 \Rightarrow \mu - v = |v - \mu|$$

d'où

$$P\left(V < \frac{n}{2}\right) \leq P\left(|V - \mu| \geq \mu - \frac{n}{2}\right)$$

en utilisant l'inégalité de Chebyshev pour $t = \frac{\mu - \frac{n}{2}}{\sigma}$, on obtient :

$$\forall t \geq 0, \qquad P(|V - \mu| \geq t\sigma) \leq \frac{1}{t^2}$$

C'est à dire que

$$P\left(|V-\mu| \geq \frac{\mu-\frac{n}{2}}{\sigma} \cdot \sigma\right) \leq \frac{1}{\left(\frac{\mu-\frac{n}{2}}{\sigma}\right)^2}$$

ou

$$P\left(|V-\mu| \geq \mu-\frac{n}{2}\right) \leq \frac{\sigma^2}{(\mu-\frac{n}{2})^2}$$

donc

$$P\left(V<\frac{n}{2}\right)=1-F(n) \leq \frac{\sigma^2}{(\mu-\frac{n}{2})^2}$$

$$\Rightarrow F(n) \geq 1-\frac{\sigma^2}{(\mu-\frac{n}{2})^2}=1-\frac{npq}{(nq-\frac{n}{2})^2}$$

Finalement, en calculant la limite :

$$\lim_{n\to\infty} \frac{npq}{(nq-\frac{n}{2})^2}=0$$

$$\Rightarrow \lim_{n\to\infty} F(n)=1 \rightarrow q.e.d$$

ANNEXE B

DÉTAILS DES MESURES DE DIVERSITÉS

Il n'y a pas de façon de mesurer la diversité d'ensembles de classifieurs qui fasse consensus dans la littérature. Plusieurs auteurs ont proposé différentes mesures pour tenter d'exprimer ce concept avec différents degrés de succès. On trouvera dans cette annexe le détail du calcul des quelques-unes des mesures de diversité les plus utilisées, ce qui inclut toutes celles ayant été employées dans le présent travail.

Entropie

Entropie (TI) ($\widetilde{E}$) - Mesure de type *globale* sans oracle. Il s'agit de la définition classique de la mesure du désordre en théorie de l'information. Elle a été utilisée notamment par Tibshirani [69] et par Cunningham [14] comme mesure de diversité dans le contexte d'ensembles de classifieurs.

$$\widetilde{E} = \frac{1}{M} \cdot \sum_{x=1}^{M} \sum_{k=1}^{K} - P_k^x \cdot \log(P_k^x)$$

avec

- P_k^x = Fréquence d'occurrence de la classe k pour l'exemple x
- M = nombre d'exemples
- K = nombre de classes

Entropie (Kuncheva)

Entropie Kuncheva) ($\widetilde{E_K}$) - Mesure de type *globale* sans oracle, il ne s'agit pas de l'entropie au sens généralement admis en théorie de l'information mais d'une définition donnée par Kuncheva [49]. Cette mesure se retrouve très souvent parmi celles utilisées comme référence lors de la comparaisons des différentes façons de mesurer la diversité. Cette mesure a l'avantage d'être sans oracle et peut donc être utilisée en phase de généralisation.

$$\widetilde{E}_K = \frac{1}{N} \cdot \sum_{j=1}^{N} \frac{1}{L - \lceil L/2 \rceil} \cdot min \{l(Z_j), L - l(Z_j)\}$$

- Z : ensemble des exemples et $Z_j \in \{Z\}$
- $l(Z_j)$: nombre de classifieurs qui ont reconnus Z_j
- $l(Z_j) = \sum_{i=1}^{L} y_{i,j}$
- L :Nombre de classifieurs de l'ensemble
- $\lceil \bullet \rceil$: arrondi à l'entier supérieur

Q$_{average}$

$Q_{average}$ (Q_{avr}) - Mesure de type *par paire* avec oracle. Basée sur la statistique Q proposée au début du XX^e s. par Yule [77], cette mesure a été largement utilisée par Kuncheva [49] dans le contexte d'**EoC**. La valeur Q est comprise entre -1 et 1. Pour des classifieurs statistiquement indépendants, l'espérance de Q est 0. Les classifieurs qui ont tendance à reconnaître correctement les mêmes objets auront des valeurs Q positives et les classifieurs qui tendent à faire des erreurs sur des objets différents auront des valeurs Q négatives.

$$Q_{avr} = \frac{2}{L \cdot (L-1)} \cdot \sum_{i=1}^{L-1} \sum_{k=i+1}^{L} Q_{ik}$$

avec :

$$Q_{ik} = \frac{N^{11}N^{00} - N^{10}N^{10}}{N^{11}N^{00} + N^{10}N^{10}}$$

- N^{ab} = Nombre de cas dans Z où N^{ab} se produit.
- L = Nombre de classifieurs de l'ensemble
- $a, b = \begin{cases} 1 & \text{si le classifieur a raison} \\ 0 & \text{sinon} \end{cases}$

Fault Majority

Fault Majority (FM) - Mesure de type *par paire* avec oracle. Proposée récemment par Ruta [64], cette mesure a été mise au point pour le contexte spécifique des ensembles

utilisant le vote à majorité simple comme fonction de fusion et constitue de ce fait une mesure importante de notre étude.

$$FM = \sum_{j=\lceil L/2 \rceil}^{L} \sum_{i^*=I}^{\lceil L/2 \rceil} Z_{i^*,j}$$

$$Z_{i^*,j} = \begin{cases} \sum_{k=1}^{N} \cdot \left[1 - y_{i,k} | m(x_k) = 0\right] / N & \text{si } j = 0 \\ M \cdot \sum_{k=1}^{N} \cdot \left[1 - y_{i,k} | m(x_k) = 0\right] / Nj & \text{si } j \neq 0 \end{cases}$$

avec :

- M : Nombre de classifieurs
- $m(x_i)$: Nombre de classifieurs en erreur sur une observation
  $x_i = M - \sum_{j=1}^{M} y_{i,j}$
  où $y_{i,j} = \begin{cases} 1 & \text{désicion correcte} \\ 0 & \text{sinon} \end{cases}$
  pour le classifieurs #j et l'observation #i
- N : Nombre d'observations

Normalized Fault Majority (NFM) - Il s'agit de la mesure FM normalisée par le nombre de classifieurs, opération nécessaire dans le cas d'optimisation de **EoC** de cardinalité différente.

$$NMF = \frac{FM}{M}$$

Ambiguïté

Ambiguïté ($\widetilde{A}$) [14] - Proposée par Krogh [44], c'est une mesure de type *globale* sans oracle utilisée par Oliveira [58] dans un contexte d'optimisation à l'aide d'algorithmes

génétiques multi-objectifs et par Cunnigham [14] dans une recherche de type escalade. Cette mesure a déjà montré des résultats intéressants dans des contextes proches du nôtre (sélection de classifieurs, algorithmes génétiques multi-objectifs) et constitue donc de ce fait la mesure de diversité de base de notre expérimentation.

$$\widetilde{A} = \frac{1}{|K| \cdot |I|} \sum_{i \in I} \sum_{k \in K} a_i(k)$$

- I est l'ensemble des classifieurs
- K est l'ensemble des exemples de la base de données
- $|\bullet|$ est le cardinal de l'ensemble
- $a_i(k)$ est l'ambiguïté du i^e classifieur pour l'observation k et se calcule ainsi :

$$a_i(k) = \begin{cases} 0 & ClasseV_i(k) = Classe\widetilde{V}_i(k) \\ 1 & sinon \end{cases}$$

$ClasseV_i$ étant la classe choisie par le classifieur i et $Classe\widetilde{V}_i$ la classe choisie par l'ensemble.

ANNEXE C

BASES DE DONNÉES DE VOTES PRÉCALCULÉES

Cette annexe présente la nomenclature des bases de données de votes précalulés, le format des fichiers de votes ainsi que la façon d'en créer de nouveaux.

Chacune de ces bases dépend de deux bases, une d'apprentissage et une de généralisation. Chacune des 6 bases d'apprentissage présentée dans le mémoire a été couplée à toutes les bases de généralisation (***VAL1***, ***VAL2***, ***VAL3*** et ***HSF_7***), pour un sous-total de $6 \times 4 = 24$ bases de votes. De plus, la totalité de la base d'entraînement (***HSF_0123***) a été couplée avec les bases ***HSF_3***, ***HSF_7*** et ***HSF_4*** ce qui fait un grand total de $24 + 3 = 27$ bases de votes. Toutes ces bases n'ont pas été utilisées dans le projet mais demeurent disponibles pour d'éventuels travaux ultérieurs.

Chaque base est sous la forme d'un fichier ASCII. Les deux premières lignes sont formées de la façon suivante :

- *Nombre total de caractéristiques dans la base de données* : 132
- *Étiquettes de la base de données de test (bonnes réponses)* : 0 1 2 3 4 ...

Les lignes subséquentes obéissent au schéma suivant :

- *32 caractéristiques définissant le classifieur N* : 9 22 102 47 ...
- *Sorties du classifieur N* : 0 1 2 3 4 ...
- *32 caractéristiques définissant le classifieur N+1* : 3 52 17 87 ...
- *Sorties du classifieur N+1* : 0 1 2 3 8 ...

Le premier caractère d'une ligne de votes est un espace alors que le premier caractère d'une ligne de caractéristiques est un chiffre, ceci afin de permettre une séparation facile des ces deux types d'information à l'aide de filtres UNIX (*grep* par exemple).

Pour créer ces bases de données, il faut utiliser le programme *prevote* et disposer des fichiers suivants :

- Fichier de caractéristiques (subspaces)
- Une base d'apprentissage
- Une base de test

Chacune des lignes du fichier de caractéristiques représente un sous-espace, et donc un *k*-**NN**. Il y a autant de lignes que de *k*-**NN** et autant de colonnes que de caractéristiques (32 dans le cas des bases présentées ici.)

La commande pour créer une nouvelle base de votes est la suivante :
./prevote k1 f32 t100 T50 -T TRAIN -t TEST -s subspaces

où :

- kn est le nombre de plus proches voisins à utiliser par les *k*-**NN** (k=1 dans l'exemple)
- fn est pour déterminer le nombre de caractéristiques des sous-espaces (f=32 dans l'exemple)
- tn est le nombre d'exemples à utiliser dans le fichier de la base de test (t=100 dans l'exemple)
- Tn est le nombre d'exemples à utiliser dans le fichier de la base d'apprentissage (t=50 dans l'exemple)
- TRAIN est le nom du fichier de la base d'apprentissage
- TEST est le nom du fichier de la base de test
- subspaces est le nom du fichier de caractéristiques

Tableau XVI

Bases de données de votes précalculés

Base d'apprentissage	Bases d'observations à classer	Nom de la base résultante
TRAIN500	***VAL1***	***prev500_val1***
	VAL2	***prev500_val2***
	VAL3	***prev500_val3***
	HSF_7	***prev500_CCVl***
TRAIN1K	***VAL1***	***prev1k_val1***
	VAL2	***prev1k_val2***
	VAL3	***prev1k_val3***
	HSF_7	***prev1k_CCVl***
TRAIN5K	***VAL1***	***prev5k_val1***
	VAL2	***prev5k_val2***
	VAL3	***prev5k_val3***
	HSF_7	***prev5k_CCVl***
TRAIN10K	***VAL1***	***prev10k_val1***
	VAL2	***prev10k_val2***
	VAL3	***prev10k_val3***
	HSF_7	***prev10k_CCVl***
TRAIN10K	***VAL1***	***prev25k_val1***
	VAL2	***prev25k_val2***
	VAL3	***prev25k_val3***
	HSF_7	***prev25k_CCVl***
TRAIN50K	***VAL1***	***prev50k_val1***
	VAL2	***prev50k_val2***
	VAL3	***prev50k_val3***
	HSF_7	***prev50k_CCVl***
HSF_0123	***HSF_3***	***prev195k_CCValid***
	HSF_7	***prev195k_CCVl***
	HSF_4	***prev195k_CCTs***

ANNEXE D

RÉSULTATS DES EXPÉRIENCES PRÉLIMINAIRES

Afin de déterminer les paramètres des k-**NN** individuels, plusieurs expériences ont été menées faisant varier la cardinalité f des sous-espaces, le nombre de plus proches voisins k et la taille de la base d'apprentissage. Les six tableaux qui suivent montrent les résultats obtenus pour chaque taille de base d'apprentissage testée (500, 1000, 5 000, 10 000, 25 000 et 50 000 prototypes). Des ces expériences découle le choix d'utiliser des k-**NN** avec une valeur de $k = 1$ dans des sous-espaces de 32 caractéristiques et entraînés sur une base de 5000 prototypes.

Tous les résultats sont calculés sur la base d'optimisation. Le dernier tableau résume ces résultats pour $k = 1$, valeur étant retenue tout au long du projet.

Tableau XVII

Résultats en fonction de k et de la cardinalité f. Base d'entraînement : ***TRAIN500***

Nombre de prototypes = 500

k	k-**NN** unique	$f = 8$	$f = 16$	$f = 32$	$f = 64$	MLP
1	86,69%	88,50%	89,61%	89,43%	87,74%	88,95%
3	86,00%	87,14%	89,12%	88,90%	87,15%	88,95%
5	86,07%	87,34%	88,54%	88,15%	87,69%	88,95%
20	85,17%	84,77%	85,70%	86,07%	85,80%	88,95%

Tableau XVIII

Résultats en fonction de k et de la cardinalité f. Base d'entraînement : ***TRAIN1k***

Nombre de prototypes = 1000

k	k-**NN** unique	$f = 8$	$f = 16$	$f = 32$	$f = 64$	MLP
1	88,14%	90,69%	91,27%	90,90%	89,42%	90,72%
3	87,61%	90,00%	91,29%	90,87%	89,15%	90,72%
5	87,21%	89,36%	90,13%	89,84%	88,28%	90,72%
20	86,83%	86,32%	87,94%	88,63%	87,98%	90,72%

Tableau XIX

Résultats en fonction de k et de la cardinalité f. Base d'entraînement : ***TRAIN5k***

Nombre de prototypes = 5000

k	k-NN unique	$f = 8$	$f = 16$	$f = 32$	$f = 64$	MLP
1	93,23%	93,94%	95,55%	95,86%	94,72%	94,47%
3	92,99%	94,18%	95,31%	95,67%	94,64%	94,47%
5	92,93%	92,97%	94,82%	95,17%	94,48%	94,47%
20	91,40%	91,11%	93,56%	93,80%	92,33%	94,47%

Tableau XX

Résultats en fonction de k et de la cardinalité f. Base d'entraînement : ***TRAIN10k***

Nombre de prototypes = 10 000

k	k-NN unique	$f = 8$	$f = 16$	$f = 32$	$f = 64$	MLP
1	94,76%	95,22%	96,84%	96,87%	96,16%	97,82%
3	94,71%	94,90%	96,20%	96,57%	95,95%	97,82%
5	94,70%	94,79%	96,25%	96,62%	95,90%	97,82%
20	93,27%	93,37%	95,15%	95,49%	94,34%	97,82%

Tableau XXI

Résultats en fonction de k et de la cardinalité f. Base d'entraînement : ***TRAIN25k***

Nombre de prototypes = 25 000

k	kNN unique	$f = 8$	$f = 16$	$f = 32$	$f = 64$	MLP
1	96,13%	96,08%	97,55%	97,73%	97,19%	98,65%
3	96,17%	95,48%	97,36%	97,65%	96,97%	98,65%
5	95,95%	95,73%	97,07%	97,54%	96,77%	98,65%
20	95,13%	94,45%	96,31%	96,66%	96,01%	98,65%

Tableau XXII

Résultats en fonction de k et de la cardinalité f. Base d'entraînement : ***TRAIN50k***

Nombre de prototypes = 50 000

k	k-**NN** unique	$f = 8$	$f = 16$	$f = 32$	$f = 64$	MLP
1	96,90%	96,55%	97,88%	98,22%	97,75%	98,87%
3	96,88%	96,19%	97,63%	98,16%	97,75%	98,87%
5	96,96%	95,81%	97,56%	97,96%	97,63%	98,87%
20	96,22%	95,86%	96,79%	97,43%	97,15%	98,87%

Tableau XXIII

Résumé des résultats pour $k = 1$

Taux de reconnaissance pour $k = 1$ et différentes tailles de bases d'apprentissage

Nb de prototypes	knn simple	$f = 8$	$f = 16$	$f = 32$	$f = 64$	MLP
500	86,69%	88,50%	89,61%	89,43%	87,74%	88,95%
1000	88,14%	90,69%	91,27%	90,90%	89,42%	90,72%
5000	93,23%	93,94%	95,55%	95,86%	94,72%	95,89%
10000	94,76%	95,22%	96,84%	96,87%	96,16%	97,82%
25000	96,13%	96,08%	97,55%	97,73%	97,19%	98,65%
50000	96,90%	96,55%	97,88%	98,22%	97,75%	98,87%

ANNEXE E

ESTIMATION DE LA PROBABILITÉ *A POSTERIORI*

Il y a deux méthodes courantes pour estimer la probabilité *a posteriori* $P(C_i|x)$ dans le contexte d'ensemble de classifieurs à vote. Une première est basée sur les matrices de confusion qui a été proposée par Xu [76] et une seconde, utilisée pas Hansen [29], qui se calcule comme une simple moyenne. Les deux auteurs ont rapporté obtenir des résultats satisfaisants mais en pratique, la méthode de Hansen est beaucoup plus simple à implanter et demande moins d'effort de calcul comme le montrent les équations de cette annexe.

Calcul de $P(C_i|x)$ selon la technique proposée par Xu

Soit un ensemble de N classifieurs indexés par la lettre $k = 1, ..., N$ et MC_k la matrice de confusion du k^e classifieur. On note $e_k(x)$ la sortie du k^e classifieur lorsqu'il observe x en entrée.

$$MC_k = \begin{pmatrix} n_{11}^k & n_{12}^k & \dots & n_{1M}^k \\ n_{21}^k & n_{22}^k & \dots & n_{2M}^k \\ \vdots & \vdots & \ddots & \vdots \\ n_{M1}^k & n_{M2}^k & \dots & n_{MM}^k \end{pmatrix} \tag{E.1}$$

Dans cette matrice, chaque ligne i correspond à une classe C_i et chaque colonne j à l'événement $e_k(x) = j$. On obtient cette matrice en utilisant une base d'apprentissage de M classes. Le nombre total d'exemples est

$$N^k = \sum_{i=1}^{M}\sum_{j=1}^{M} n_{ij}^k \tag{E.2}$$

parmi lequel le nombre d'exemples de chaque classe est

$$n_{i\cdot}^k = \sum_{j=1}^{M} n_{ij}^k \ , i = 1, ..., M \tag{E.3}$$

et le nombre d'exemples assignés à la classe j par le classifieur k est

$$n_{\cdot j}^k = \sum_{i=1}^{M} , j = 1, ..., M. \tag{E.4}$$

Avec la connaissance de la matrice de confusion MC_k, on déduit les probabilités conditionnelles $P(x \in C_i | e_k(x) = j)$, c'est-à-dire la probabilité que l'observation x appartienne à la classe C_i sachant que le classifieur k a choisi la classe C_j.

$$P(x \in C_i | e_k(x) = j) = \frac{n_{ij}^k}{n_{\cdot j}^k} = \frac{n_{ij}^k}{\sum_{i=1}^{M} n_{ij}^k} , i = 1, ..., M \tag{E.5}$$

Par simplicité, cette probabilité conditionnelle est notée $P(C_i | e_k(x))$ et l'estimation de la probabilité *a posteriori* $P(C_i|x)$ est

$$P(C_i|x) = \eta \cdot \prod_{k=1}^{N} P(C_i | e_k(x)) \tag{E.6}$$

avec η qui est une constante qui permet de s'assurer que $\sum_{i=1}^{M} P(C_i|x) = 1$. Cette constante s'obtient de la façon suivante :

$$\eta = \frac{1}{\sum_{i=1}^{M} \prod_{k=1}^{N} P(C_i | e_k(x))} \tag{E.7}$$

Calcul de $P(C_i|x)$ selon la technique proposée par Hansen

Soit un ensemble de N classifieurs. Le nombre de votes dans l'ensemble pour la classe i étant donnée l'entrée x est notée $v(i|x)$. La probabilité *a posteriori* se calcule donc ainsi :

$$P(C_i|x) = \frac{v(I(x)|x)}{N} \tag{E.8}$$

avec :

$$I(x) = \arg \max_{i=1}^{N} v(i|x) \tag{E.9}$$

On remarquera que ce calcul correspond au complément de la mesure de diversité appelée ambiguïté (ambiguïté + $P(C_i|x) = 1$). En effet, l'ambiguïté mesure le nombre de classifieurs n'ayant pas voté pour la décision majoritaire alors que cette définition de $P(C_i|x)$ mesure le nombre de voix obtenu par la décision majoritaire. Maximiser l'ambiguïté correspond donc à minimiser la moyenne, sur tous les exemples, de cette définition de $P(C_i|x)$.

ANNEXE F

ENSEMBLES DE SOLUTIONS OBTENUS

Les figures de cette annexe montrent les ensembles de solutions dans lesquels se trouvent les **EoC** du tableau IX, c'est-à-dire les **EoC** les plus performants trouvés par chacune des méthodes. Notons que les solutions de l'AG Simple sont projetés artificiellement sur deux axes (performance et complexité) puisque seul le taux de reconnaissance est utilisé dans la recherche. Les autres ensembles de solutions sont projetés dans leur espace naturel. Les ensembles de solutions obtenus avec algorithmes génétiques multi-objectifs sont représentés par leur front de Pareto pour les mesures faites avec la base d'optimisation. Ces mêmes points sont ensuite évalués sur la base de validation et leur performance actualisée sur le graphique (mais pas leur second objectif dont on conserve la valeur mesurée en optimisation).

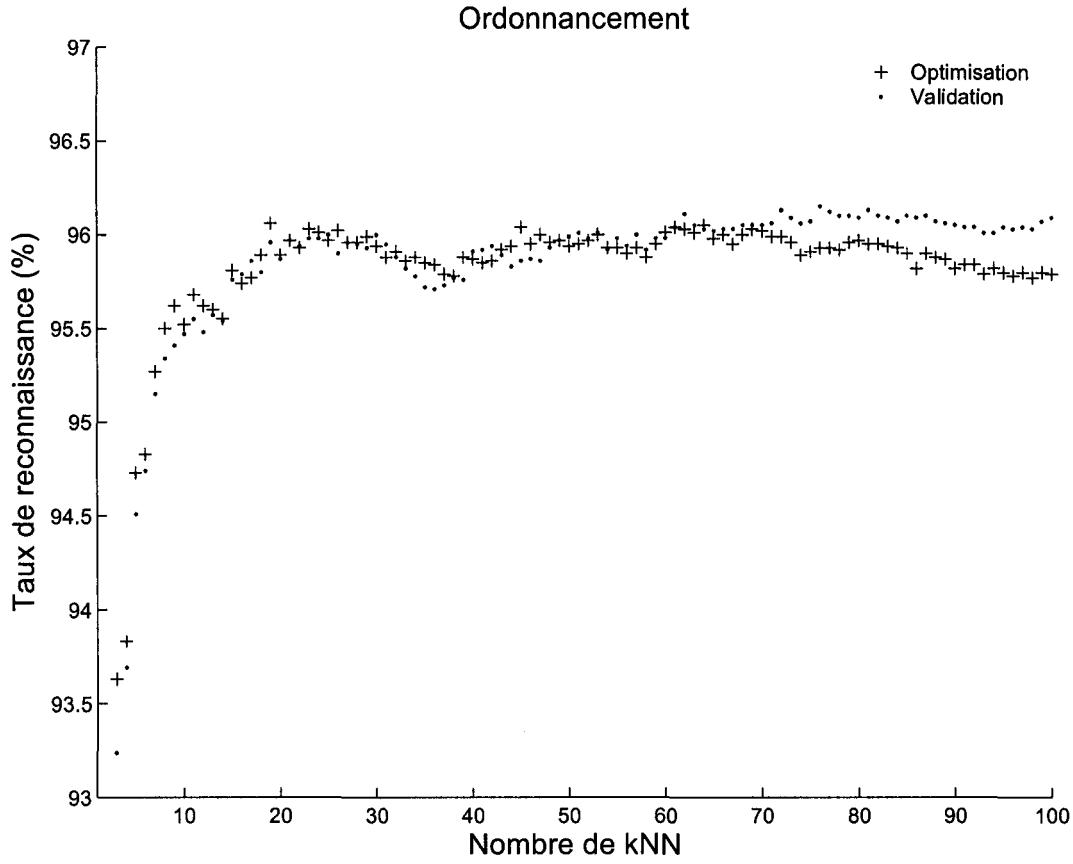


Figure 42 **Ensemble des solutions d'où provient le meilleur EoC obtenu à l'aide de la méthode d'ordonnancement.** Les classifieurs sont triés par ordre décroissant de performance (base d'optimisation). On forme 100 ensembles, chacun contenant les N meilleurs classifieurs, $N \in \{1, 2, ..., 100\}$. Le meilleur **EoC** est choisi sur la base de validation.

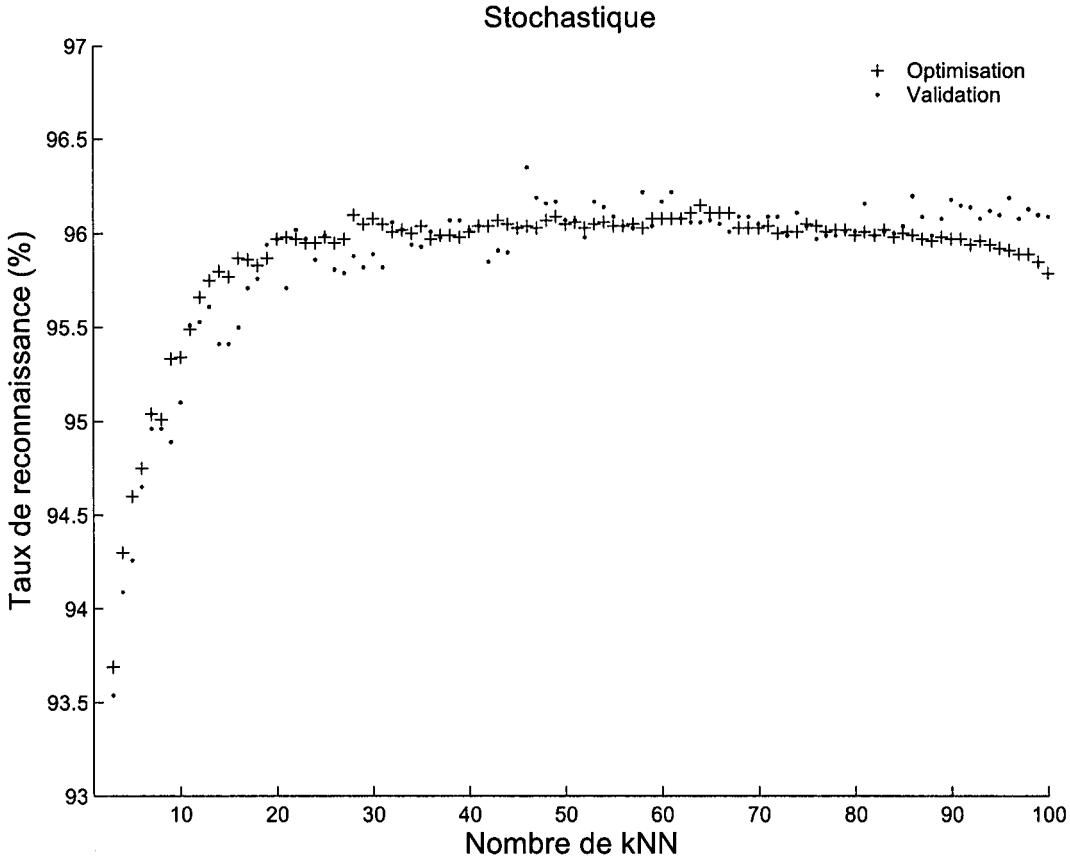


Figure 43 **Ensemble-solutions d'où provient le meilleur EoC obtenu à l'aide de la recherche stochastique.** Les 100 classifieurs sont ajoutés un à un à l'ensemble dans un ordre aléatoire. Le processus est répété autant de fois qu'il est nécessaire pour évaluer le même nombre de solution qu'avec les recherches basées sur les algorithmes génétiques. L'ensemble des solutions est constitué de la meilleure solution évaluée pour chaque cardinalité (mesurées sur la base d'optimisation). Le meilleur **EoC** est choisi sur la base de validation.

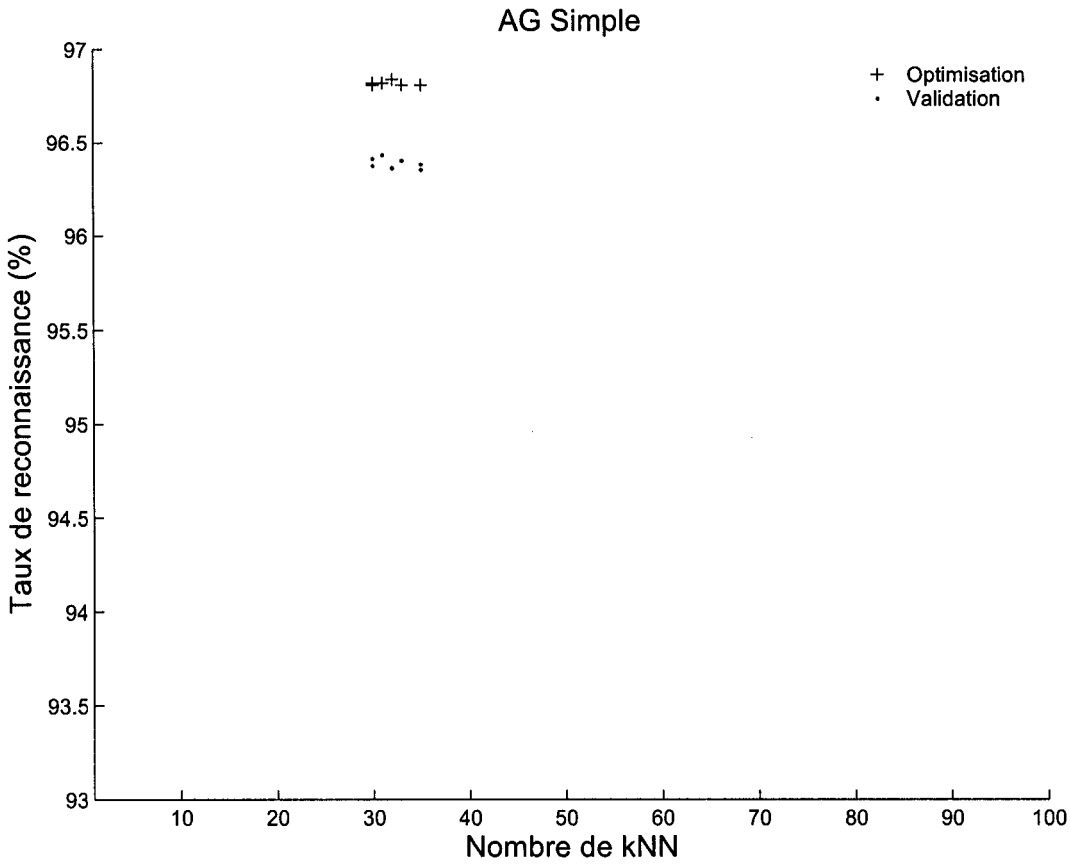


Figure 44 **Ensemble-solutions d'où provient le meilleur EoC obtenu à l'aide de l'AG Simple.** Seule la performance, mesurée sur la base d'optimisation, est utilisée pour guider la recherche. L'ensemble-solutions est constitué de la totalité des solutions de la dernière génération. Le meilleur **EoC** est choisi sur la base de validation.

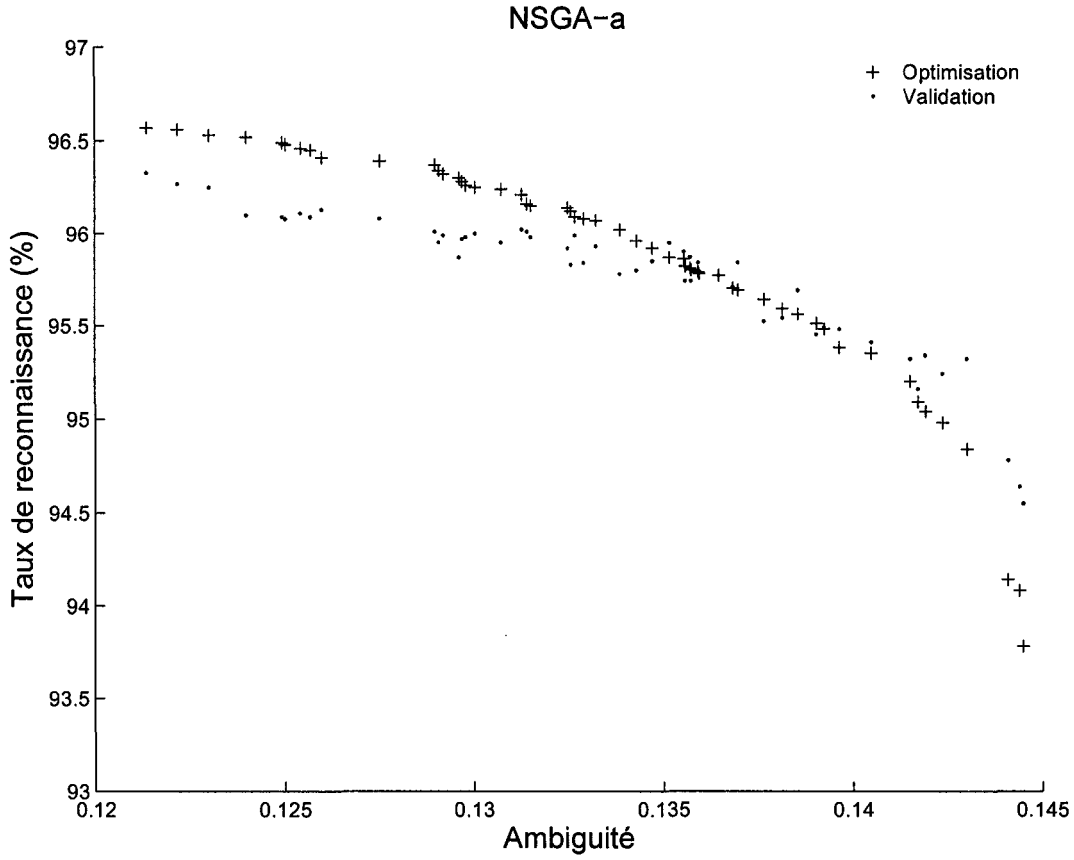


Figure 45 **Ensemble-solutions d'où provient le meilleur EoC obtenu à l'aide de *NSGA*-a.** La performance et l'ambiguïté sont conjointement maximisées sur la base d'optimisation. L'ensemble-solutions est composé du front de Pareto à la dernière génération. Le meilleur **EoC** est choisi sur la base de validation.

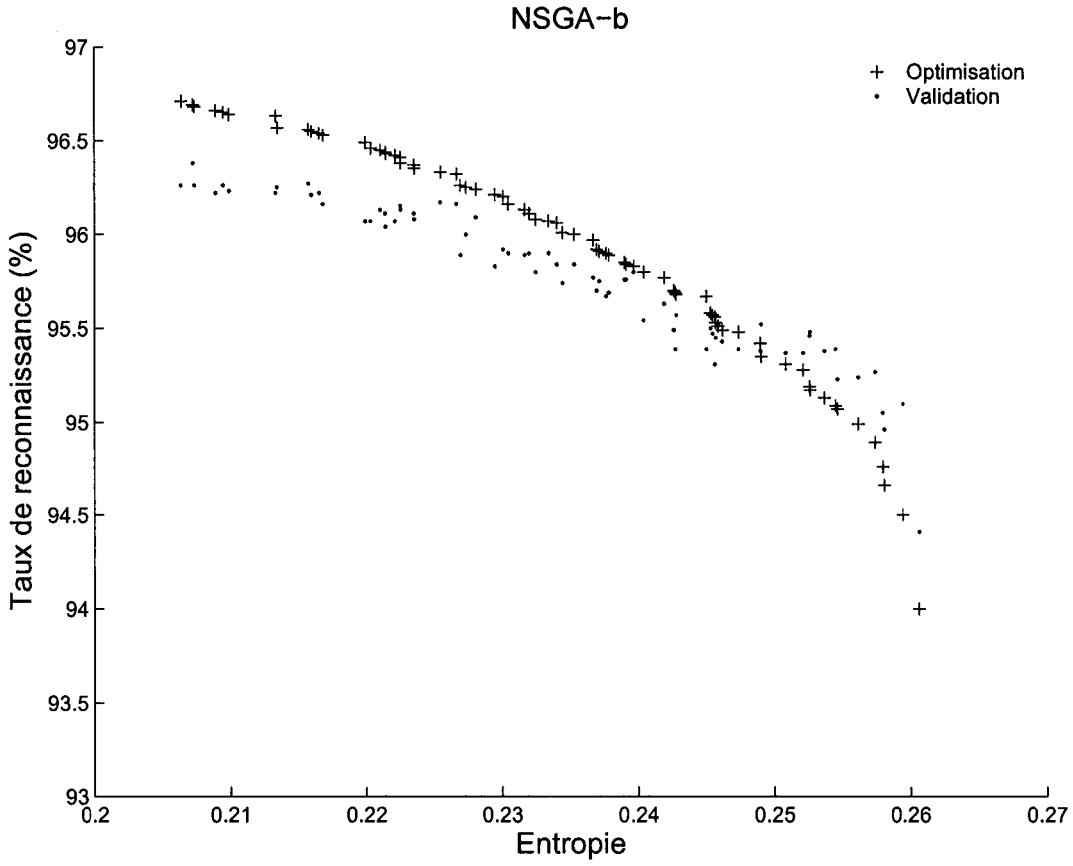


Figure 46 **Ensemble-solutions d'où provient le meilleur EoC obtenu à l'aide de *NSGA*-b.** La performance et l'entropie sont conjointement maximisées sur la base d'optimisation. L'ensemble-solutions est composé du front de Pareto à la dernière génération. Le meilleur **EoC** est choisi sur la base de validation.

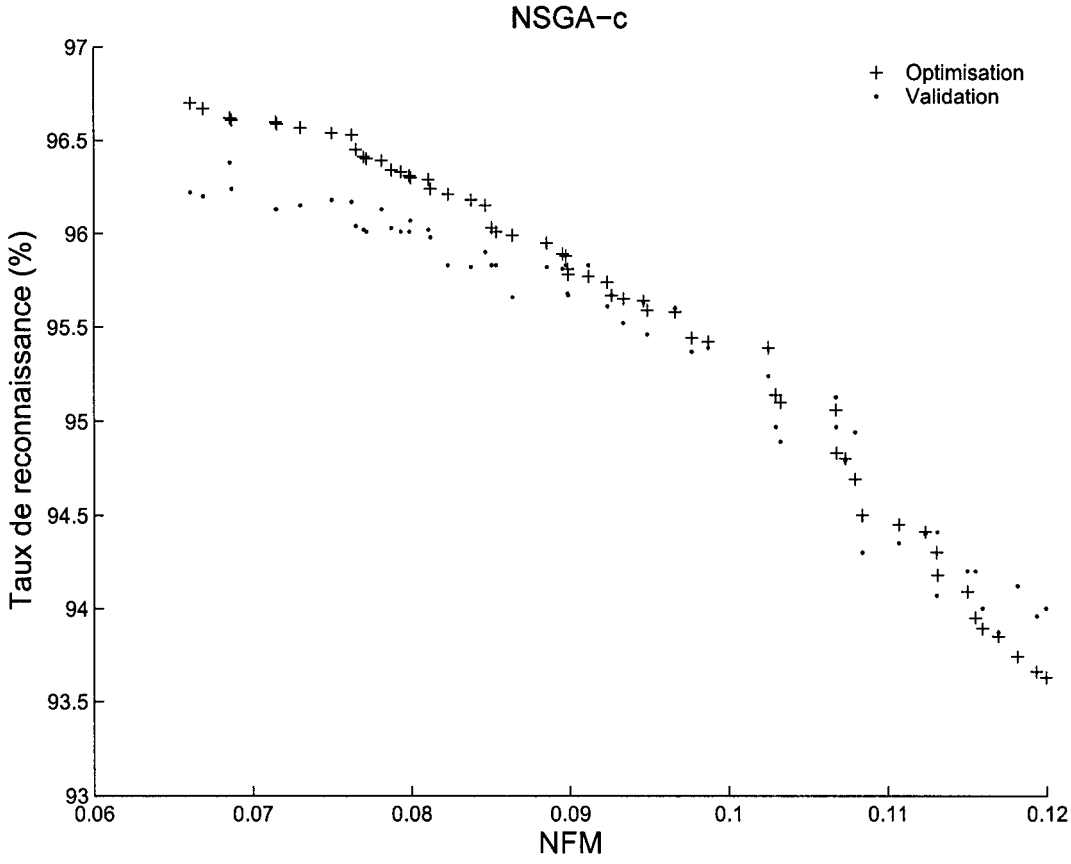


Figure 47 **Ensemble-solutions d'où provient le meilleur EoC obtenu à l'aide de *NSGA*-c.** La performance et NFM sont conjointement maximisées sur la base d'optimisation. L'ensemble-solutions est composé du front de Pareto à la dernière génération. Le meilleur **EoC** est choisi sur la base de validation.

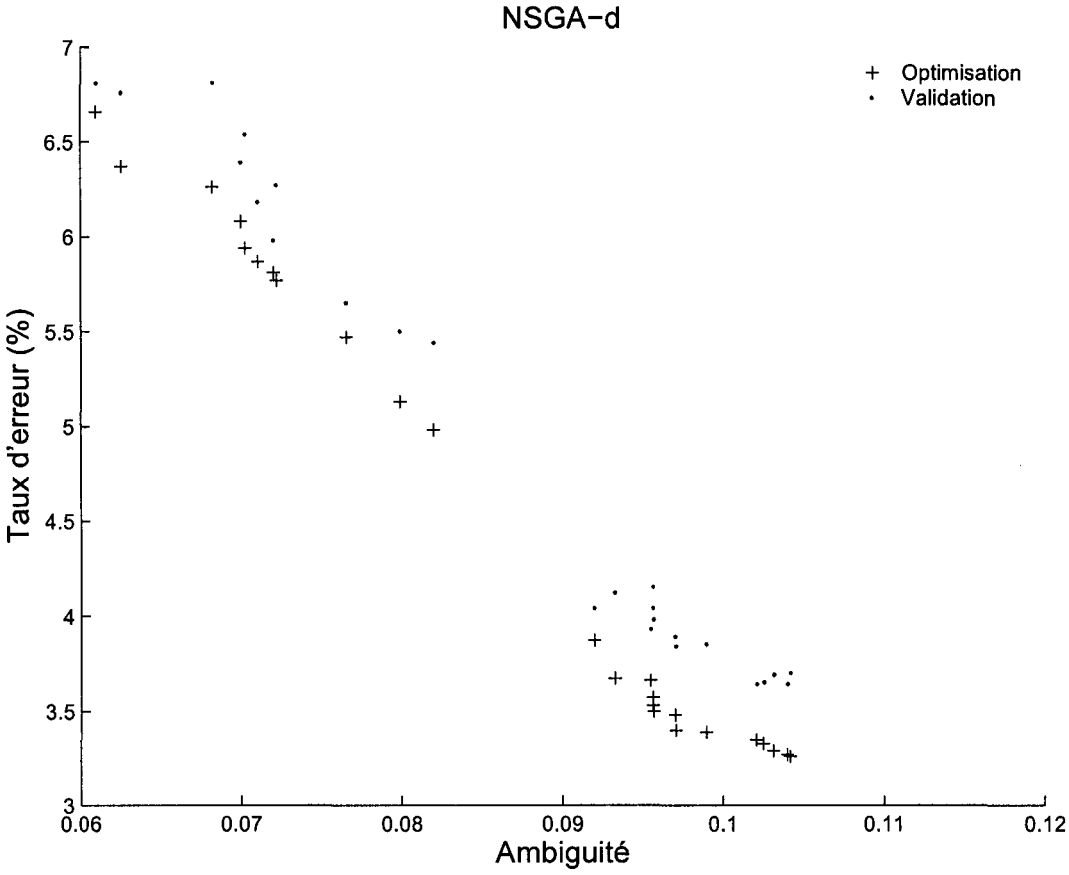


Figure 48 **Ensemble-solutions d'où provient le meilleur EoC obtenu à l'aide de *NSGA*-d.** L'erreur et l'ambiguïté sont conjointement minimisées sur la base d'optimisation. L'ensemble-solutions est composé du front de Pareto à la dernière génération. Le meilleur **EoC** est choisi sur la base de validation.

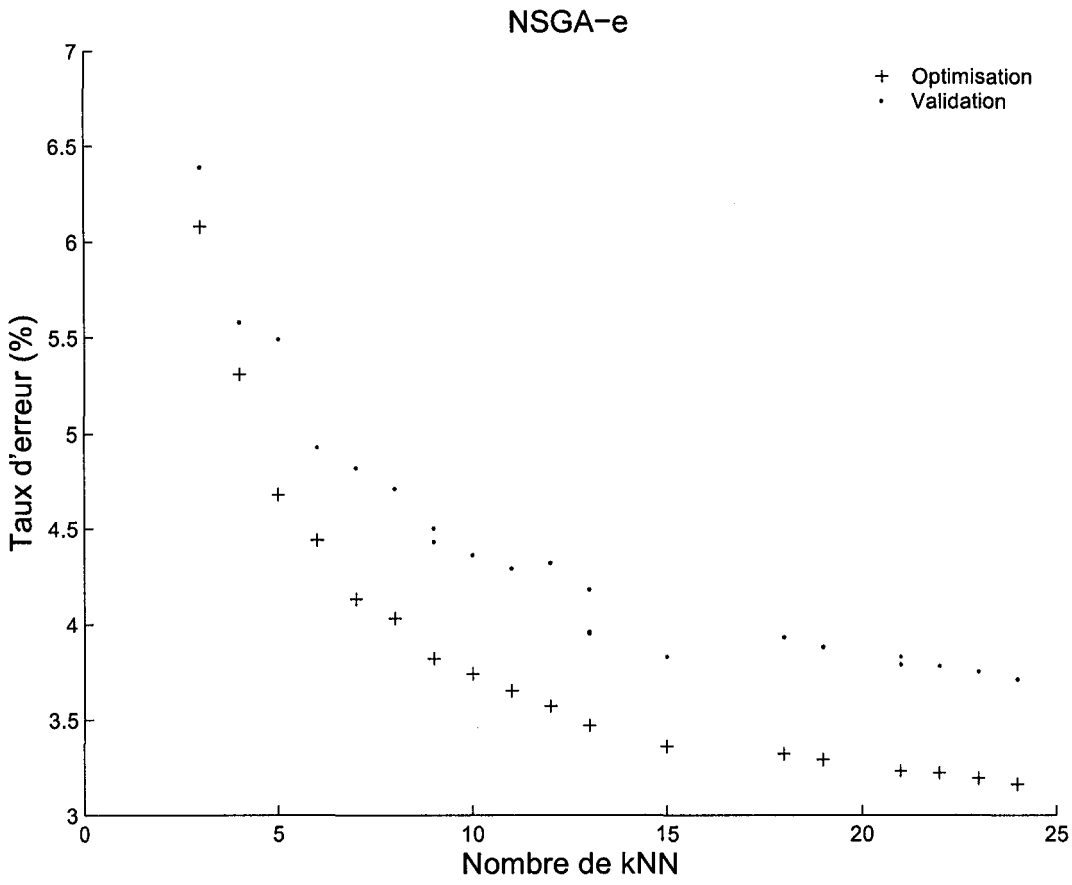


Figure 49 **Ensemble-solutions d'où provient le meilleur EoC obtenu à l'aide de *NSGA*-e.** L'erreur et la cardinalité sont conjointement minimisées sur la base d'optimisation. L'ensemble-solutions est composé du front de Pareto à la dernière génération. Le meilleur **EoC** est choisi sur la base de validation.

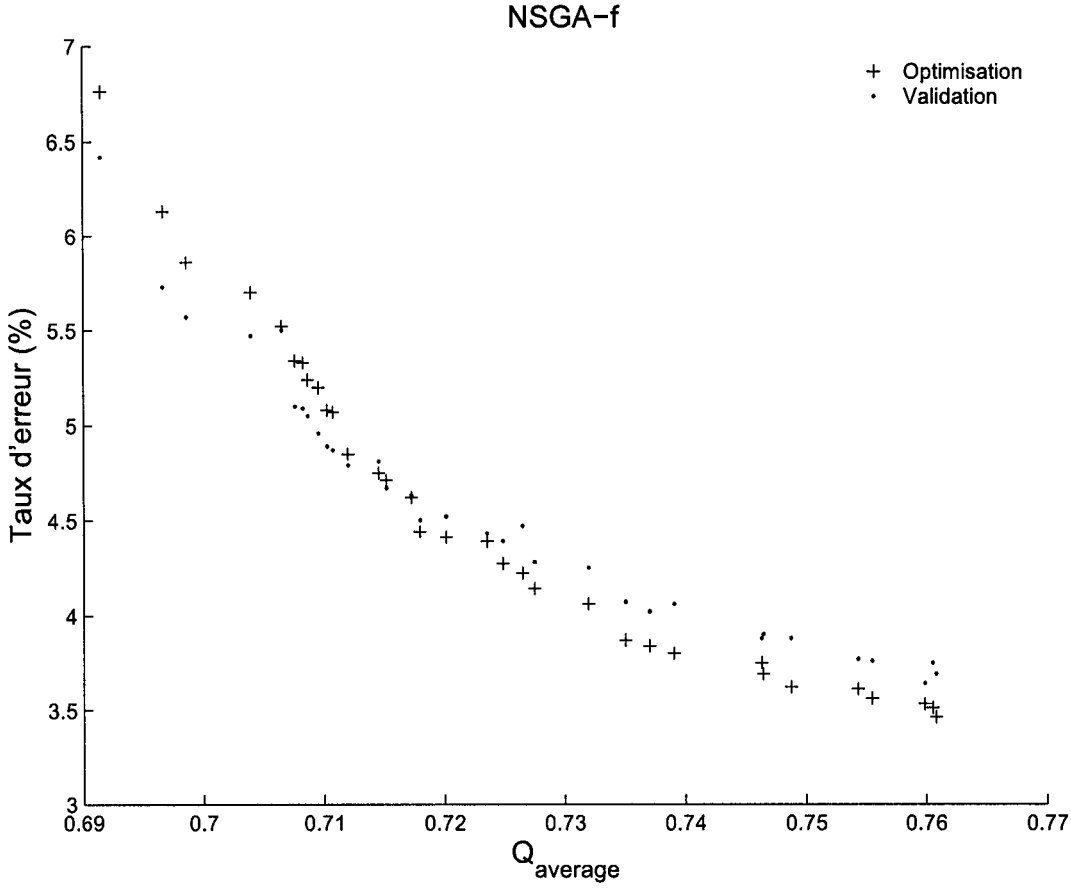


Figure 50 **Ensemble des solutions d'où provient le meilleur EoC obtenu à l'aide de *NSGA*-f.** L'erreur et la mesure de $Q_{average}$ sont conjointement minimisées sur la base d'optimisation. L'ensemble-solutions est composé du front de Pareto à la dernière génération. Le meilleur **EoC** est choisi sur la base de validation.

ANNEXE G

VALEURS NUMÉRIQUES DES INTERVALLES DE CONFIANCE

Les tableaux de cette annexe montrent les valeurs numériques des moyennes et variances de la cardinalité des **EoC** et de leurs taux de reconnaissance sur la base de test. Ces valeurs sont placées dans un intervalle de confiance à 99%.

Tableau XXIV

Moyennes, écart-types et intervalles de confiance à 99% de la cardinalité des ensembles

	Moyenne	Erreur-type	Borne inf.	Borne sup.	Écart-type	Borne inf.	Borne sup.
Stochastique	62.50	2.709	55.03	69.97	14.839	11.046	22.060
AG Simple	34.67	0.506	33.27	36.06	2.771	2.063	4.119
NSGA-a	36.97	0.739	34.93	39.00	4.047	3.013	6.017
NSGA-b	33.03	0.774	30.90	35.17	4.238	3.155	6.301
NSGA-c	26.30	0.952	23.68	28.92	5.214	3.881	7.751
NSGA-d	20.17	0.528	18.71	21.62	2.890	2.151	4.296
NSGA-e	18.87	0.573	17.29	20.45	3.137	2.336	4.664
NSGA-f	21.03	0.506	19.64	22.43	2.773	2.064	4.122

Tableau XXV

Moyennes, écart-types et intervalles de confiance à 99% du taux de reconnaissance en test

	Moyenne	Erreur-type	Borne inf.	Borne sup.	Écart-type	Borne inf.	Borne sup.
Stochastique	96.33	0.0083	96.31	96.36	0.0457	0.0340	0.0680
AG Simple	96.41	0.0110	96.38	96.44	0.0603	0.0449	0.0896
NSGA-a	96.33	0.0127	96.30	96.37	0.0697	0.0518	0.1035
NSGA-b	96.33	0.0124	96.30	96.37	0.0681	0.0507	0.1013
NSGA-c	96.33	0.0179	96.28	96.38	0.0981	0.0730	0.1458
NSGA-d	96.26	0.0163	96.21	96.30	0.0894	0.0665	0.1329
NSGA-e	96.35	0.0168	96.31	96.40	0.0921	0.0686	0.1369
NSGA-f	96.33	0.0140	96.29	96.37	0.0767	0.0571	0.1140

ANNEXE H

DÉTAILS DES TESTS T ET F

Cette section détaille le résultat des valeurs obtenues lors des calculs des valeurs t et F permettant de tester respectivement l'égalité de deux moyennes et de deux variances.

Le test t de Student consiste à rejeter ou non l'hypothèse H_0 selon laquelle deux moyennes $\bar{x_1}$ et $\bar{x_2}$ sont identiques (donc que $\bar{x_1} - \bar{x_2} = 0$). L'hypothèse H_0 est rejetée si elle a un degré de signification p inférieure à un seuil de signification α. Règle de décision : Si $p < \alpha \Rightarrow$ on rejete H_0 et on considère les moyennes comme étant différentes.

Le test F de Fisher consiste à rejeter ou non l'hypothèse H_0 selon laquelle deux variances σ_1 et σ_2 sont identiques. L'hypothèse H_0 est rejetée si elle a un degré de signification p inférieure à un seuil de signification α. Règle de décision : Si $p < \alpha \Rightarrow$ on rejete H_0 et on considère les variances comme étant différentes.

Le tableau XXVI montre différents seuils de significations α et les valeurs de t et F correspondant aux échantillons étudiés. Les tableaux XXVII et XXVIII montrent l'ensemble des valeurs calculées pour effectuer les tests d'hypothèses pour respectivement la cardinalité des ensembles et leur taux de reconnaissance sur la base de test.

Tableau XXVI

Valeurs critiques pour les tests t (bilatéral) et F. Degrés de liberté = 29

	t	F
$\alpha = 1\%$	2,7564	1,8608
$\alpha = 5\%$	2,0452	2,4234

Tableau XXVII

Valeurs obtenues pour faire les tests de Student et de Fisher (cardinalité)

Méthodes	$\bar{x_1} - \bar{x_2}$	t	p	F	p
Stochastique-AG Simple	27,83	10,0991	0,0000	28,6774	0,0000
Stochastique-*NSGA*-a	25,53	9,0927	0,0000	13,4441	0,0000
Stochastique-*NSGA*-b	29,47	10,4583	0,0000	12,2570	0,0000
Stochastique-*NSGA*-c	36,20	12,6065	0,0000	8,1003	0,0000
Stochastique-*NSGA*-d	42,33	15,3378	0,0000	26,3682	0,0000
Stochastique-*NSGA*-e	43,63	15,7574	0,0000	22,3686	0,0000
Stochastique-*NSGA*-f	41,47	15,0456	0,0000	28,6388	0,0000
AG Simple-*NSGA*-a	-2,30	-2,5685	0,0128	0,4688	0,0456
AG Simple-*NSGA*-b	1,63	1,7667	0,0825	0,4274	0,0253
AG Simple-*NSGA*-c	8,37	7,7615	0,0000	0,2825	0,0011
AG Simple-*NSGA*-d	14,50	19,8371	0,0000	0,9195	0,8227
AG Simple-*NSGA*-e	15,80	20,6742	0,0000	0,7800	0,5077
AG Simple-*NSGA*-f	13,63	19,0490	0,0000	0,9987	0,9971
NSGA-a-*NSGA*-b	3,93	3,6763	0,0005	0,9117	0,8051
NSGA-a-*NSGA*-c	10,67	8,8520	0,0000	0,6025	0,1785
NSGA-a-*NSGA*-d	16,80	18,5041	0,0000	1,9613	0,0748
NSGA-a-*NSGA*-e	18,10	19,3601	0,0000	1,6638	0,1764
NSGA-a-*NSGA*-f	15,93	17,7893	0,0000	2,1302	0,0460
NSGA-b-*NSGA*-c	6,73	5,4888	0,0000	0,6609	0,2706
NSGA-b-*NSGA*-d	12,87	13,7381	0,0000	2,1513	0,0433
NSGA-b-*NSGA*-e	14,17	14,7144	0,0000	1,8250	0,1109
NSGA-b-*NSGA*-f	12,00	12,9770	0,0000	2,3365	0,0255
NSGA-c-*NSGA*-d	6,13	5,6356	0,0000	3,2552	0,0022
NSGA-c-*NSGA*-e	7,43	6,6910	0,0000	2,7614	0,0079
NSGA-c-*NSGA*-f	5,27	4,8850	0,0000	3,5355	0,0011
NSGA-d-*NSGA*-e	1,30	1,6693	0,1004	0,8483	0,6607
NSGA-d-*NSGA*-f	-0,87	-1,1853	0,2407	1,0861	0,8255
NSGA-e-*NSGA*-f	-2,17	-2,8342	0,0063	1,2803	0,5100

Tableau XXVIII

Valeurs obtenues pour faire les tests de Student et de Fisher (taux de reconnaissance sur la base de test)

Méthodes	$\bar{x_1} - \bar{x_2}$	t	p	F	p
Stochastique-AG Simple	-0,0729	-5,2765	0,0000	0,5755	0,1427
Stochastique-*NSGA*-a	0,0042	0,2765	0,7831	0,4311	0,0268
Stochastique-*NSGA*-b	0,0012	0,0770	0,9389	0,4504	0,0355
Stochastique-*NSGA*-c	0,0071	0,3594	0,7206	0,2175	0,0001
Stochastique-*NSGA*-d	0,0747	4,0729	0,0001	0,2618	0,0005
Stochastique-*NSGA*-e	-0,0177	-0,9433	0,3495	0,2465	0,0003
Stochastique-*NSGA*-f	0,0049	0,3022	0,7636	0,3554	0,0069
AG Simple-*NSGA*-a	0,0771	4,5844	0,0000	0,7491	0,4414
AG Simple-*NSGA*-b	0,0740	4,4578	0,0000	0,7826	0,5135
AG Simple-*NSGA*-c	0,0800	3,8063	0,0003	0,3779	0,0108
AG Simple-*NSGA*-d	0,1475	7,4966	0,0000	0,4550	0,0379
AG Simple-*NSGA*-e	0,0552	2,7457	0,0080	0,4284	0,0257
AG Simple-*NSGA*-f	0,0778	4,3689	0,0001	0,6176	0,2004
NSGA-a-*NSGA*-b	-0,0031	-0,1716	0,8643	1,0448	0,9069
NSGA-a-*NSGA*-c	0,0029	0,1318	0,8956	0,5045	0,0704
NSGA-a-*NSGA*-d	0,0704	3,4054	0,0012	0,6074	0,1854
NSGA-a-*NSGA*-e	-0,0219	-1,0395	0,3029	0,5718	0,1382
NSGA-a-*NSGA*-f	0,0007	0,0381	0,9698	0,8244	0,6066
NSGA-b-*NSGA*-c	0,0059	0,2728	0,7860	0,4828	0,0545
NSGA-b-*NSGA*-d	0,0735	3,5821	0,0007	0,5813	0,1500
NSGA-b-*NSGA*-e	-0,0189	-0,9018	0,3709	0,5473	0,1102
NSGA-b-*NSGA*-f	0,0038	0,2014	0,8411	0,7891	0,5276
NSGA-c-*NSGA*-d	0,0676	2,7887	0,0071	1,2040	0,6205
NSGA-c-*NSGA*-e	-0,0248	-1,0101	0,3167	1,1335	0,7380
NSGA-c-*NSGA*-f	-0,0022	-0,0956	0,9242	1,6343	0,1920
NSGA-d-*NSGA*-e	-0,0924	-3,9419	0,0002	0,9415	0,8722
NSGA-d-*NSGA*-f	-0,0697	-3,2426	0,0020	1,3574	0,4156
NSGA-e-*NSGA*-f	0,0226	1,0344	0,3053	1,4417	0,3301

ANNEXE I

MATRICES DE CONFUSION

Tableau XXIX

Matrice de confusion du meilleur ensemble obtenu par ordonnancement (en pourcentage sur base de test)

	Classes attendues									
	0	1	2	3	4	5	6	7	8	9
0	98,20	0,00	0,03	0,03	0,19	0,02	0,15	0,00	4,06	0,19
1	0,15	97,56	0,03	0,06	0,03	0,02	0,15	0,06	0,90	0,05
2	0,08	0,96	98,09	0,69	0,20	0,13	0,00	0,23	0,20	0,03
3	0,15	0,17	0,71	97,22	0,00	4,46	0,00	0,33	0,38	0,51
4	0,20	0,00	0,07	0,02	97,05	0,27	0,10	2,04	0,72	1,38
5	0,05	0,02	0,02	0,35	0,00	93,72	0,46	0,00	0,44	0,02
6	0,68	0,63	0,42	0,05	1,15	0,32	98,93	0,02	0,54	0,00
7	0,00	0,43	0,43	1,12	0,14	0,21	0,00	96,19	0,24	3,31
8	0,25	0,37	0,15	0,15	0,08	0,22	0,20	0,08	91,24	0,24
9	0,20	0,10	0,02	0,30	1,10	0,28	0,00	1,23	1,15	94,19

Cette annexe présente les matrices de confusion, sur la base de test, des meilleurs ensembles obtenus par toutes les méthodes de recherche au terme des 30 itérations. Les matrices de confusion permettent de voir quelles classes sont les plus difficiles à reconnaître et avec quelles autres classes elles sont confondues. Sur les matrices de confusion qui suivent, les colonnes correspondent aux classes attendues et les lignes aux décisions du classifieur. Les nombres étant exprimés en pourcentage, la somme de chaque colonne fait 100% puisque chacune des observations est assignée à une et une seule classe ; la valeur (i, i) sur la diagonale représente les taux de reconnaissance de la classe i.

Tableau XXX

Matrice de confusion du meilleur ensemble obtenu par recherche stochastique (en pourcentage sur base de test)

	Classes attendues									
	0	1	2	3	4	5	6	7	8	9
0	98,17	0,00	0,05	0,05	0,20	0,03	0,20	0,00	4,23	0,19
1	0,23	97,67	0,06	0,06	0,06	0,02	0,20	0,06	0,88	0,05
2	0,12	0,99	98,31	0,62	0,17	0,17	0,00	0,35	0,15	0,05
3	0,17	0,12	0,48	97,30	0,00	4,39	0,00	0,40	0,36	0,53
4	0,15	0,00	0,05	0,02	96,95	0,17	0,14	1,77	0,78	1,28
5	0,04	0,00	0,00	0,33	0,00	93,79	0,44	0,00	0,51	0,02
6	0,64	0,61	0,37	0,03	1,15	0,34	98,81	0,02	0,49	0,00
7	0,00	0,42	0,46	1,18	0,16	0,24	0,00	96,51	0,24	3,53
8	0,27	0,31	0,19	0,17	0,10	0,24	0,20	0,07	90,98	0,25
9	0,18	0,12	0,00	0,22	1,15	0,27	0,00	1,00	1,23	94,01

Tableau XXXI

Matrice de confusion du meilleur ensemble obtenu par AG Simple (en pourcentage sur base de test)

	Classes attendues									
	0	1	2	3	4	5	6	7	8	9
0	98,00	0,00	0,02	0,03	0,14	0,02	0,19	0,02	3,60	0,12
1	0,18	97,78	0,08	0,09	0,06	0,03	0,20	0,06	0,87	0,06
2	0,08	0,94	98,19	0,85	0,25	0,13	0,00	0,32	0,23	0,03
3	0,18	0,12	0,65	97,15	0,02	3,93	0,00	0,66	0,36	0,43
4	0,26	0,02	0,09	0,02	97,33	0,20	0,10	1,77	0,99	1,38
5	0,07	0,02	0,02	0,32	0,00	94,30	0,32	0,00	0,42	0,02
6	0,71	0,69	0,24	0,05	0,80	0,37	99,00	0,00	0,51	0,00
7	0,02	0,42	0,53	1,15	0,14	0,19	0,00	96,51	0,26	3,10
8	0,27	0,15	0,15	0,10	0,10	0,22	0,19	0,05	91,12	0,29
9	0,20	0,08	0,02	0,22	1,11	0,28	0,00	0,78	1,49	94,49

Tableau XXXII

Matrice de confusion du meilleur ensemble obtenu par *NSGA*-a (en pourcentage sur base de test)

	Classes attendues									
	0	1	2	3	4	5	6	7	8	9
0	98,08	0,00	0,05	0,03	0,19	0,03	0,14	0,00	4,33	0,17
1	0,20	97,69	0,03	0,08	0,03	0,05	0,20	0,06	0,81	0,11
2	0,08	1,04	98,41	0,79	0,22	0,12	0,00	0,50	0,32	0,05
3	0,20	0,10	0,55	97,42	0,00	3,81	0,00	0,60	0,35	0,43
4	0,24	0,02	0,05	0,05	97,12	0,15	0,09	1,50	0,89	1,26
5	0,05	0,02	0,00	0,26	0,00	94,44	0,42	0,00	0,55	0,02
6	0,64	0,68	0,31	0,05	1,22	0,42	99,00	0,02	0,64	0,00
7	0,00	0,38	0,43	1,06	0,16	0,24	0,00	96,82	0,26	3,25
8	0,29	0,24	0,15	0,12	0,03	0,17	0,15	0,14	90,39	0,24
9	0,18	0,07	0,00	0,13	0,98	0,25	0,00	0,53	1,34	94,39

Tableau XXXIII

Matrice de confusion du meilleur ensemble obtenu par *NSGA*-b (en pourcentage sur base de test)

	Classes attendues									
	0	1	2	3	4	5	6	7	8	9
0	98,10	0,00	0,05	0,03	0,22	0,02	0,19	0,00	4,14	0,19
1	0,24	97,56	0,03	0,09	0,03	0,05	0,20	0,08	0,81	0,09
2	0,10	1,02	98,24	0,80	0,17	0,18	0,00	0,34	0,23	0,07
3	0,20	0,10	0,56	97,12	0,02	3,41	0,00	0,83	0,33	0,46
4	0,24	0,00	0,10	0,02	97,21	0,20	0,05	1,52	0,95	1,21
5	0,05	0,04	0,00	0,42	0,00	94,77	0,46	0,00	0,48	0,02
6	0,63	0,64	0,24	0,05	1,00	0,42	99,02	0,02	0,53	0,00
7	0,00	0,40	0,53	1,20	0,19	0,21	0,00	96,63	0,29	3,26
8	0,24	0,32	0,22	0,14	0,07	0,17	0,08	0,08	90,46	0,20
9	0,17	0,15	0,00	0,12	1,05	0,27	0,00	0,68	1,64	94,41

Tableau XXXIV

Matrice de confusion du meilleur ensemble obtenu par *NSGA*-c (en pourcentage sur base de test)

	Classes attendues									
	0	1	2	3	4	5	6	7	8	9
0	98,10	0,00	0,05	0,05	0,20	0,03	0,25	0,00	3,89	0,20
1	0,17	97,93	0,03	0,09	0,03	0,03	0,23	0,08	0,91	0,06
2	0,12	0,94	98,41	0,77	0,25	0,13	0,00	0,69	0,34	0,05
3	0,18	0,15	0,38	97,22	0,00	3,69	0,00	0,63	0,35	0,36
4	0,22	0,05	0,12	0,03	97,34	0,17	0,10	1,65	0,95	1,46
5	0,04	0,04	0,00	0,26	0,00	94,62	0,37	0,00	0,51	0,02
6	0,66	0,51	0,27	0,03	0,83	0,29	98,95	0,00	0,47	0,00
7	0,02	0,37	0,50	1,28	0,14	0,27	0,00	96,51	0,27	3,21
8	0,27	0,17	0,22	0,10	0,07	0,17	0,08	0,07	90,81	0,24
9	0,20	0,05	0,00	0,13	1,08	0,28	0,00	0,55	1,34	94,31

Tableau XXXV

Matrice de confusion du meilleur ensemble obtenu par *NSGA*-d (en pourcentage sur base de test)

	Classes attendues									
	0	1	2	3	4	5	6	7	8	9
0	98,13	0,00	0,05	0,05	0,17	0,03	0,19	0,02	3,68	0,14
1	0,23	97,67	0,06	0,09	0,05	0,05	0,20	0,06	0,81	0,06
2	0,05	0,99	98,29	0,89	0,32	0,13	0,00	0,40	0,35	0,03
3	0,18	0,12	0,60	97,17	0,02	3,91	0,00	0,55	0,35	0,40
4	0,34	0,02	0,09	0,02	97,16	0,24	0,07	1,91	0,97	1,36
5	0,05	0,02	0,02	0,33	0,00	94,32	0,37	0,00	0,46	0,02
6	0,58	0,68	0,27	0,03	0,88	0,41	99,00	0,00	0,53	0,00
7	0,00	0,42	0,43	1,07	0,16	0,14	0,00	96,29	0,22	3,04
8	0,19	0,27	0,17	0,12	0,10	0,20	0,17	0,05	91,25	0,32
9	0,22	0,05	0,00	0,22	1,10	0,25	0,00	0,91	1,24	94,56

Tableau XXXVI

Matrice de confusion du meilleur ensemble obtenu par *NSGA*-e (en pourcentage sur base de test)

	Classes attendues									
	0	1	2	3	4	5	6	7	8	9
0	98,10	0,00	0,00	0,05	0,22	0,03	0,22	0,02	3,39	0,12
1	0,17	97,88	0,06	0,09	0,03	0,03	0,17	0,09	0,88	0,14
2	0,08	0,92	98,42	0,96	0,22	0,13	0,00	0,45	0,30	0,03
3	0,20	0,10	0,45	97,02	0,00	4,27	0,00	0,41	0,43	0,43
4	0,27	0,05	0,10	0,03	97,22	0,22	0,12	1,80	0,89	1,53
5	0,05	0,02	0,02	0,33	0,00	94,04	0,33	0,00	0,48	0,02
6	0,63	0,58	0,24	0,03	1,07	0,34	99,00	0,03	0,56	0,00
7	0,00	0,42	0,46	1,25	0,16	0,18	0,00	96,56	0,26	3,29
8	0,29	0,17	0,17	0,08	0,07	0,19	0,15	0,02	91,39	0,22
9	0,18	0,07	0,05	0,13	0,96	0,23	0,00	0,78	1,28	94,13

Tableau XXXVII

Matrice de confusion du meilleur ensemble obtenu par *NSGA*-f (en pourcentage sur base de test)

	Classes attendues									
	0	1	2	3	4	5	6	7	8	9
0	97,96	0,00	0,03	0,05	0,17	0,03	0,20	0,00	4,12	0,20
1	0,14	97,73	0,03	0,11	0,02	0,03	0,20	0,11	0,88	0,12
2	0,10	1,01	98,41	0,85	0,17	0,08	0,00	0,40	0,30	0,05
3	0,28	0,05	0,50	96,98	0,00	3,96	0,00	0,73	0,31	0,40
4	0,19	0,00	0,10	0,03	97,28	0,31	0,10	1,41	0,94	1,19
5	0,05	0,04	0,00	0,35	0,00	94,19	0,39	0,00	0,42	0,04
6	0,69	0,56	0,29	0,05	0,97	0,39	98,95	0,00	0,68	0,00
7	0,00	0,38	0,46	1,26	0,14	0,14	0,00	96,74	0,32	3,12
8	0,32	0,29	0,15	0,14	0,03	0,19	0,15	0,03	90,35	0,29
9	0,23	0,17	0,00	0,15	1,18	0,35	0,00	0,73	1,51	94,51

ANNEXE J

COMPLEXITÉ DES *K*-NN ET DES ENSEMBLES DE *K*-NN

Cette annexe présente et compare la complexité computationnelle d'un k-**NN** simple et celle d'un ensemble de k-**NN** projetés dans des sous-espaces aléatoires telle que présenté par Ho [33].

Complexité d'un k-NN simple

Supposons n prototypes d'apprentissage dans un espace de caractéristiques de dimension d. Le calcul de la distance euclidienne entre une observation à classer et un prototype est de complexité $O(d)$. Pour trouver le prototype le plus proche d'une observation, la méthode la plus simple consiste à calculer la distance entre cette observation et les n prototypes puis à comparer chaque distance obtenue avec celle que l'on considère la plus proche. La complexité computationnelle d'un k-**NN** est donc d'ordre $O(d \cdot n^2)$.

Complexité d'un ensemble de k-NN

Un ensemble de M classifieurs de type k-**NN** projetés dans des sous-espaces aléatoires de cardinalité c n'a pas nécessairement une complexité $O(M \cdot d \cdot n^2)$ puisque le coût en calcul des distances peut être réparti. La distance euclidienne se calcule ainsi :

$$D_e = \sqrt{\sum_{0<f\leq d} (x'_f - x_f)^2}$$

Les termes $(x'_f - x_f)^2$ n'ont besoin d'être calculés qu'une seule fois par observation à classer, peu importe le nombre de classifieurs. Ce calcul a une complexité $O(d)$. Ainsi, le calcul de distance se fait en sommant les $(x'_f - x_f)^2$ correspondants aux sous-espaces choisis. Ce calcul est donc d'ordre $O(c)$ (avec $c < d$) et la complexité d'un ensemble d'un nombre M de k-**NN** est $O(M \cdot c \cdot n^2 + d)$. Ou plus généralement, en négligeant le terme constant $O(d)$, la complexité d'un ensemble de k-**NN** est $O(M \cdot c \cdot n^2)$.

La complexité d'un **EoC** de M classifieurs est donc $M \cdot c/d$ fois celle d'un k-**NN** simple. Dans les deux cas, le facteur dominant est le nombre de prototypes n puisque ce dernier entraîne une progression quadratique du nombre d'opérations.

Par exemple, lors de nos expériences, nous avons formé un ensemble optimisé de 24 classifieurs dans des sous-espaces de 32 caractéristiques. Cet ensemble ne nécessite que $24 \cdot 32/132 \approx 5,8$ plus d'opérations qu'un k-**NN** unique dans un espace de 132 caractéristiques, avec à la clef un taux de reconnaissance supérieur de plus de 3% sur la base de test.

BIBLIOGRAPHIE

[1] F. M. Alkoot and J. Kittler. Moderating k-nn classifiers. *Pattern Analysis & Apppli-cation*, 5(3) :326–332, 2002.

[2] G. Baillargeon. *Méthodes statistiques de l'ingénieur*, volume 1. Les éditions SMG, Trois-Rivières, Canada, 1990.

[3] S D. Bay. Nearest neighbor classification from multiple feature subsets. *Intelligent Data Analysis*, 3(3) :191–209, 1999.

[4] S. D. Bay. Combining nearest neighbor classifiers through multiple feature subsets. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 37–45, Madison, Wisconsin USA, 24-27 juillet 1998. Morgan Kaufmann Publishers Inc.

[5] W. W. Bledsoe. Some results on multicategory pattern recognition. *J. ACM*, 13(2) :304–316, 1966.

[6] I. Bloch. Information combination operators for data fusion : A comparative review with classification. *IEEE. Transaction on Systems, Man and Cybernetics*, 26(1) :52–67, September 1996.

[7] M. M. Bongard. *Pattern Recognition*. Spartan Books, New York, États-Unis, 1970.

[8] L. Breiman. Bagging predictors. *Machine Learning*, 24(2) :123–140, 1996.

[9] E. K. Burke, S. Gustafson, and G. Kendall. Diversity in genetic programming : An analysis of measures and correlation with fitness. *IEEE Transactions on Evolutionary Computation*, 8(1) :47–62, 2004.

[10] D. Cheng and H. Yan. Recognition of handwritten digits based on contour information. *Pattern Recognition*, 31(3) :235–255, 1998.

[11] N. Chomsky. *Syntatic Structures*. Mouton and Co, La Haye, Pays-Bas, 1957.

[12] C.K. Chow. On optimum recognition error and reject tradeoff. *IEEE Transaction on Information Theory*, 16(12) :41–46, 1970.

[13] T. M. Cover and P. E. Hart. Nearest neigbor pattern classification. *IEEE Transaction on Information Theory*, IT-13(1) :123–140, janvier 1967.

[14] P. Cunningham and John Carney. Diversity versus quality in classification ensembles based on feature selection. In *Machine Learning : ECML 2000, 11th European Conference on Machine Learning*, volume 1810, pages 109–116, Barcelone, Espagne, 30 mai-2 juin 2000. Springer.

[15] P. Cunningham and G. Zenobi. Case representation issues for case-based reasoning from ensemble research. In *Case-Based Reasoning Research and Development, 4th International Conference on Case-Based Reasoning*, volume 2080 of *Lecture Notes in Computer Science*, pages 146–157, Vancouver, Canada, 30 juillet-2 août 2001. Springer.

[16] B. V. Dasarathy, editor. *Nearest Neighbor (NN) Norms : NN Pattern Classification Techniques*. IEEE Computer Society Press, Los Alamitos, CA, 1991.

[17] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Baffins Lane, Chichester, West Sussex, PO19 1UD, Royaume-Uni, 2001.

[18] T. G. Dietterich. Ensemble methods in machine learning. In J. Kittler and F. Roli, editors, *Proceedings of first International Workshop on Multiple Classifier System*, pages 1–15, Cagliari, Italie, 20-23 juin 2000. Springer-Verlag.

[19] T. G. Dietterich. Ensemble learning. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*. MIT Press, 2002.

[20] P. Domingos. A unified bias-variance decomposition and its applications. In *Proc. 17th International Conf. on Machine Learning*, pages 231–238, Stanford, CA, États-Unis, 29 juin-2 juillet 2000. Morgan Kaufmann, San Francisco, CA.

[21] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, 2nd edition, 2001.

[22] H. Foundalis. Harry foundalis - research on the bongard problems [en-ligne]. http ://www.cs.indiana.edu/~hfoundal/research.html [consulté en octobre 2004].

[23] Y. Freund and R. Schapire. Experiments with a new boosting algorithm. In *Proc. of 13th International Conference on Machine Learning*, pages 148–156, Bari, Italie, 3-6 juillet 1996.

[24] G. Fumera, F. Roli, and G. Giacinto. Reject option with multiple thresholds. *Pattern Recognition*, 33(12) :2099–2101, 1999.

[25] G. Fumera, F. Roli, and G. Giacinto. Multiple reject thresholds for improving classification reliability. In *Proceedings of the Joint IAPR International Workshops on Advances in Pattern Recognition*, pages 863–871. Springer-Verlag, 2000.

[26] H. T. Glantz. On the recognition of information with a digital computer. *J. ACM*, 4(2) :178–188, 1957.

[27] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multi-modal function optimisation. In John J. Grefenstette, editor, *Proc. of 2nd International Conference on Genetic Algorithms and Their Applications*, pages 41–49, Cambridge, Massachusetts, États-Unis, 1987. Lawrence Erlbaum Associates, Inc.

[28] C. Guerra-Salcedo and D. Whitley. Genetic approach to feature selection for ensemble creation. In *Proc. of Genetic and Evolutionary Computation Conference*, pages 236–243, Orlando, États-Unis, 13-17 juillet 1999.

[29] L. K. Hansen, C. Liisberg, and P. Salamon. The error-reject tradeoff. *Open Systems & Information Dynamics*, 4(2) :159–184, 1997.

[30] T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 20(8) :832–844, 1998.

[31] T. K. Ho. Complexity of classification problems and comparative advantages of combined classifiers. In J. Kittler and F. Roli, editors, *Multiple Classifier System*, pages 97–106. Springer-Verlag, 2000.

[32] T. K. Ho. Multiple classifier combination : Lessons and next steps. In A. Kandel and H. Bunke, editors, *Hybrid Methods in Pattern Recognition*, pages 171–198. World Scientific, 2002.

[33] T.K. Ho. Nearest neighbors in random subspaces. In *Proceedings of the 2th Int'l Workshop on Statistical Techniques in Pattern Recognition*, pages 640–648, Sydney, Australie, August 1998.

[34] D. R. Hofstadter. *Godel, Escher, Bach : An Eternal Golden Braid.* Basic Books, Inc., 1979.

[35] J. Hu and Y. Yan. Structural primitive extraction and coding for handwritten numeral recognition. *Pattern Recogniton*, 31 :493–509, 1998.

[36] Itqon, S. Kaneko, and S. Igarashi. Combining multiple k-nearest neighbor classifiers using feature combinations. *IECI (Indonesian Society on Electrical Electronics, Communication and Information)*, 2(3) :23–31, 2000.

[37] P. C. Jackson. *Introduction to Artificial Intelligence.* Dover Publications Incorporated, 1985.

[38] A. K. Jain, R. Duin, and J. Mao. Statistical pattern recognition : A review. Rapport technique MSU-CSE-00-5, Department of Computer Science, Michigan State University, East Lansing, Michigan, March 2000.

[39] M. I. Jordan. Why the logistic function ? a tutorial discussion on probabilities and neural networks (MIT computational cognitive science report 9503). Rapport technique, Massachusetts Institute of Technology, 1995.

[40] H-C. Kim, S. Pang, H-M. Je, D. Kim, and S.Y. Bang. Pattern classification using support vector machine ensemble. In *16th International Conference on Pattern Recognition (ICPR'02) Volume 2*, pages 160–163, Québec (Québec), Canada, 11-15 août 2002. IEEE Computer Society.

[41] J. Kittler and F. M. Alkoot. Sum versus vote fusion in multiple classifier systems. *IEEE Transaction on Pattern Analysis ans Machine Intelligence*, 25(1) :110–115, 2003.

[42] J. Kittler, M. Hatef, R. Duin, and J. Matas. On combining classifiers. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 20(3) :226–239, 1998.

[43] S. Krause and R. Polikar. An ensemble of classifiers approach for the missing feature problem. In *Proceedings of the International Joint Conference on Neural Networks*, volume 1, pages 553–558, Portland, Oregon, États-Unis, 20-24 juillet 2003.

[44] A. Krogh and J. Vedelsby. Neural network ensembles, cross validation, and active learning. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 231–238. The MIT Press, 1995.

[45] L. I. Kuncheva and L. C. Jain. Designing classifier fusion systems by genetic algorithms. *IEEE-EC*, 4(4) :327, November 2000.

[46] L.I. Kuncheva. Reducing the computational demand of the nearest neighbor classifier. In *School of Informatics Symposium on Computing*, pages 61–64, Aberystwyth, Royaume-Uni, 2001.

[47] L.I. Kuncheva. Classifier ensembles for changing environment. In *Proceedings 5th Int. Workshop on Multiple Classifiers Systems*, volume 3077 of *Lecture Notes in Computer Science*, pages 1–15, Cagliari, Italie, 9-11 juin 2004. Springer-Verlag.

[48] L.I. Kuncheva. That elusive diversity in classifier ensembles. In *Proc. of the first Iberian Conference on Pattern Recognition and Image Analysis*, pages 1126–1138, Mallorca, Espagne, juin 2003.

[49] L.I. Kuncheva and C. J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2) :181–207, 2003.

[50] L.I. Kuncheva and C. J. Whitaker. Ten measures of diversity in classifier ensembles :limits for two classifiers. In *Proc. of IEEE Workshop on Intelligent Sensor Processing*, pages 1–10, Birmingham, Royaume-Uni, février 2001.

[51] L. Lam. Classifier combinations : Implementations and theoretical issues. In J. Kittler and F. Roli, editors, *Multiple Classifier System*, pages 77–86, Cagliari, Italie, 20-23 juin 2000. Springer-Verlag.

[52] A. Linhares. A glimpse at the metaphysics of bongard problems. *Artificial Intelligence*, 121(1-2) :251–270, 2000.

[53] L. Miclet. *Méthodes structurelles pour la reconnaissance des formes*. Eyrolles, Paris, 1984.

[54] L. Micó and J. Oncina. Comparison of fast nearest neighbor classifiers for handwritten character recognition. *Pattern Recognition Letters*, 19(3-4) :351–365, 1998.

[55] M. E. Morita, L. S. Oliveira, and R. Sabourin. Supervised and unsupervised feature selection for ensemble of classifiers. In *Proceedings on the 9^{th} International Workshop on Frontiers in Handwriting Recognition*, Tokyo, Japon, 26-29 octobre 2004.

[56] M. Nadler and E. P. Smith. *Pattern Recognition Engineering*. John Wiley & Sons, New York, 1993.

[57] S. Nitzan and J. Paroush. *Collective Decision Making : An Economic Outlook*. Cambridge University Press, Cambridge, 1985.

[58] L. S. Oliveira. *Automatic Recognition of Handwritten Numerical Strings*. Thèse de doctorat, École de Technologie Supérieure, Université du Québec, Montréal, Québec, Canada, 2003.

[59] L. S. Oliveira, R. Sabourin, F. Bortolozzi, and C.Y. Suen. Automatic recognition of handwritten numerical strings : A recognition and verification strategy. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 24(11) :1438–1454, 2002.

[60] D. W. Opitz. Feature selection for ensembles. In *Proceedings of the 16^{th} national conference on artificial intelligence and eleventh innovation applications of AI conference on Artificial intelligence and innovative applications of artificial intelligence*, pages 379–384. American Association for Artificial Intelligence, 1999.

[61] R. C. Prather and L. M. Uhr. Discovery and learning techniques for pattern recognition. In *Proceedings of the 1964 19th ACM national conference*, pages 42.201–42.2010. ACM Press, janvier 1964.

[62] A. Rosenfeld. Picture processing by computer. *ACM Comput. Surv.*, 1(3) :147–176, 1969.

[63] B. Rosner. *Fundamentals of Biostatistics, 5th Edition*. Duxbury Press, 1999.

[64] D. Ruta and B. Gabrys. A theoretical analysis of the limits of majority voting errors for multiple classifier systems. *Pattern Analysis Application*, 2002.

[65] D. Ruta and B. Gabrys. Classifier selection for majority voting. *Information Fusion*, 2004.

[66] D. Ruta and B. Gabrys. Analysis of the correlation between majority voting error and the diversity measures in multiple classifier system. In *Proceedings of the 4th International Symposium on Soft Computing*, Paisley, UK, 26 - 29 june 2001.

[67] S. Salzberg. On comparing classifiers : A critique of current research and methods. *Data Mining and Knowledge Discovery*, 1 :1–12, 1999.

[68] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3) :221–248, 1995.

[69] R. Tibshirani. Bias, variance and prediction error for classification rules. Rapport technique, Statistics Department, University of Toronto, 1996.

[70] G. Tremblay, R. Sabourin, and P. Maupin. Optimizing nearest neighbours in random subspaces using a multi-objective genetic algorithm. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 1, pages 208–211, Cambridge, Royaume-Uni, 23-26 août 2004. IEEE.

[71] G. Tremblay, R. Sabourin, and P. Maupin. Optimisation des ensembles de classifieurs de type plus proche voisin générés par la méthode des sous-espaces aléatoires - TR 2003-323. Rapport technique, Centre R & D pour la défense Canada, Valcartier, Québec, Canada, décembre 2003.

[72] G. Tremblay, R. Sabourin, and P. Maupin. Utilisation des algorithmes génétiques pour l'optimisation d'ensembles de classificateurs - TR 2003-322,. Rapport technique, Centre R & D pour la défense Canada, Valcartier, Québec, Canada, décembre 2003.

[73] K. Tumer and N. Oza. Decimated input ensembles for improved generalization. In *Proceedings of the Int. Joint Conf. on Neural Networks*, Washington DC, États-Unis, 10-16 juillet 1999.

[74] G. Valentini and T. Dietterich. Low bias bagged support vector machines. In *Proc. International Conf. on Machine Learning*, pages 776–783, Washington DC, États-Unis, 21-24 août 2003.

[75] J.W. Woods and H. Stark. *Probability and Random Processes with Applications to Signal Processing (3^{rd} Edition)*. Prentice Hall, 2001.

[76] L. Xu, A. Krzyzak, and C.Y. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(3) :418–435, 1992.

[77] U. Yule. On the association of attributes in statistics. *Philosophical Transactions of the Royal Society of London*, 194 :257–319, 1900.

[78] B. Zhang, M. Fu, and H. Yan. A nonlinear neural network model of mixture of local principal component analysis : application to handwritten digits recogntion. *Pattern Recognition*, 34(2) :203–214, 2001.

[79] A. L. Zobrist. Complex preprocessing for pattern recognition. In *Proceedings of the 1971 26^{th} annual conference*, pages 339–348. ACM Press, janvier 1971.