Sélection de modèles pour la classification supervisée avec des SVM (Séparateurs à Vaste Marge). Application en traitement et analyse d'images.

Gilles Lebrun

UNIVERSITE de CAEN/BASSE-NORMANDIE

U.F.R.: Sciences

ECOLE DOCTORALE SIMEM

THESE

Présentée par

Gilles Lebrun

et soutenue

le 24 novembre 2006

en vue de l'obtention du

DOCTORAT de l'UNIVERSITE de CAEN
Spécialité : Informatique

(Arrêté du 7 août 2006)

Sélection de modèles pour la classification supervisée avec des SVM (Séparateurs à Vaste Marge). Application en traitement et analyse d'images.

MEMBRES du JURY

Hubert Cardot, Professeur, Université Francois Rabelais de Tours ..... Directeur de Thèse
Stéphane Canu, Professeur, INSA de Rouen ..... Rapporteur
Héléne Paugam-Moisy, Professeur, Université Lumière Lyon 2 ..... Rapporteur
Olivier Lezoray, Maître de Conférences, UCBN ..... Examinateur
Christophe Charrier, Maître de Conférences, UCBN ..... Examinateur
Marinette Revenu, Professeur, ENSICAEN ..... Examinateur

0.1 Remerciements

Ce travail de recherche à été réalisé dans l'équipe Vision et Analyse d'Images (VAI) du Laboratoire Universitaire des Sciences Appliquées de Cherbourg (LUSAC). Il a été co-financé par l'association Cœur Cancer et le conseil régional de Basse-Normandie.

Je tiens à remercier Monsieur Hubert Cardot, professeur à l'Université François Rabelais de Tours, pour avoir dirigé mon doctorat et guidé mes recherches. Je remercie Messieurs Olivier Lezoray et Christophe Charrier, tous deux maîtres de conférences à l'Université de Caen Basse-Normandie, pour avoir accepté d'être mes co-encadrants. Les nombreuses discussions et conseils que nous avons eux ensemble ont été une aide très précieuse au développement de mon travail de recherche. Je tiens également à remercier, Abderrahim Elmoataz, professeur à l'université de Caen Basse-Normandie et responsable du groupe VAI, pour m'avoir accueilli au sein de son équipe.

Je remercie Monsieur Stéphane Canu, Professeur à INSA de Rouen, et Madame Hélène Paugam-Moisy, Professeur à l'université Lumière de Lyon 2, d'avoir accepté d'être rapporteurs de cette thèse. Je remercie également, Madame Marinette Revenu, Professeur à l'ENSICAEN, d'avoir accepté de faire partie des membres du jury de cette thèse.

Je remercie le Docteur Hubert Elie, chef du service d'anatomie et cytologie pathologiques de l'hôpital Pasteur de Cherbourg, pour m'avoir accueilli dans ses locaux pendant les deux premières années de ma thèse. Je tiens également à remercier Michel Lécluse, biotechnologiste en cytologie clinique, pour sa disponibilité pendant ces deux années.

Je remercie l'ensemble des collègues de l'IUT de Saint-Lô, où j'ai été d'abord vacataire puis ATER à plein temps, pour l'excellente ambiance qui a été propice au bon déroulement de mon travail de thèse. Certains de ces collègues, en acceptant de relire des passages de ma thèse, de jeter un œil averti sur ma présentation de soutenance, de corriger mon anglais ou par leurs discutions éclairées ont également participé à mon travail de thèse. Je tiens donc à remercier tout particulièrement, parmi ces collègues, Virginie Levavasseur, Marie-Pierre Besnard, Jérome Clouet et Patrice Schiller.

Pour finir, je remercie l'ensemble des membres de ma famille et mes camarades qui ont compris mon manque de disponibilité pendant ces quelques années. J'ai tout particulièrement une pensé émue pour mes deux filles Pauline et Justine, ainsi que ma femme Nathalie, car je sais qu'elles ont vécu ces quatre années avec difficulté. La finition de ce document est donc la symbolisation de jours meilleurs pour toutes les trois.

TABLE DES MATIÈRES

CHAPITRE 1

INTRODUCTION

Sommaire

Notre société est de nos jours une société tournée vers l'information. Les systèmes d'information jouent donc un rôle prépondérant dans la production de données. Ces données sont à présent produites en masse et de nombreux besoins sont apparus pour leur traitement et leur analyse car ces masses de données atteignent des complexités telles qu'elles sont difficilement appréhendables par un être humain. La nature de ces traitements et analyses est diverse et variée : acquisition, numérisation, filtrage, compression, archivage, indexation, modélisation, prévision, diagnostic, aide à la décision, etc. Ceux-ci sont exploités intensivement dans de nombreux domaines : médecine, aéronautique, biologie, astronomie, etc. Certains de ces domaines ont la particularité de manipuler des flux de données correspondant à de grandes quantités d'images.

L'imagerie médicale est certainement l'un des domaines de la médecine qui a le plus progressé ces vingt dernières années. Dans ce domaine, l'exploitation d'images numériques est courante depuis de nombreuses années à travers des outils comme les scanners, les radiographes, échographes et microscopes numériques. Ces appareillages exploitent des outils de traitements d'images d'une haute technicité. Plus récemment, nous avons assité à une forte demande de logiciels de traitements des images afin d'aider les praticiens dans leurs diagnostics. Les progrès dans le développement de techniques d'apprentissage artificiel ont permis de répondre à cette attente. L'origine de cette demande est de plus en plus pressante et accroît l'importance de l'analyse d'images dans ce domaine. L'imagerie médicale a également des besoins importants d'archivage d'images par des techniques de compression. Celle-ci doit être réalisée en préservant une haute qualité visuelle des images pour qu'elles puissent être interprétées ultérieurement par le corps médical.

Dans le cadre du pôle *traitement et analyse d'images bas-normand*, les différents acteurs que sont le Laboratoire Universitaire des Sciences Appliquées de Cherbourg (LUSAC), le Groupe de REcherches en Informatique et Instrumentation de Caen (GREYC), le Groupe Régional d'Etudes sur le CANcer (GRECAN) et le Centre Hospitalier Louis Pasteur (CHLP) de Cherbourg sont fortement impliqués dans l'imagerie médicale. Une collaboration a également été initiée au cours de cette thèse avec le laboratoire SIC de Poitiers (plus spécialement sur la thématique d'analyse de la qualité visuelle). L'élaboration de nos travaux se déroule donc dans un environnement pluridisciplinaire. Néanmoins, ces travaux se focalisent plus particulièrement sur des problématiques d'apprentissage ar-

tificiel, tout en tenant compte de certaines spécificités et contraintes liées aux problèmes de traitement et d'analyse d'images considérés.

L'objectif principal de l'apprentissage artificiel est de permettre, à partir d'un ensemble de données, de les généraliser au mieux pour produire un processus décisionnel performant. Dans le cadre de l'apprentissage supervisé, le but est de produire une fonction de décision à partir d'un algorithme d'apprentissage et des données de façon à ce que les prédictions réalisées sur de nouvelles données conduisent à un minimum d'erreurs. Les domaines liés au traitement et à l'analyse d'images produisent de grandes quantités de données et leurs analyses s'inscrivent donc dans une problématique de fouille de données. Le développement de systèmes d'apprentissage supervisé doit alors prendre en compte le fait que les temps d'apprentissage soient exploitables par rapport à la taille de ces données. L'exploitation des images nécessite généralement des contraintes de temps réel, il est donc également nécessaire que les systèmes d'apprentissage artificiel produisent des processus décisionnels de complexités raisonnables.

Parmi les différents types d'algorithmes d'apprentissage définis ces dernières années, on trouve les séparateurs à vaste marge (SVM : *Support Vector Machines*) dont le créateur principal est Vladimir Vapnik. Le nombre de publications concernant cet algorithme (et ces variantes) est très important, aussi bien sur le plan théorique que sur le plan pratique, marquant par là même l'intérêt qu'il suscite dans la communauté scientifique. La raison principale de cet intérêt est que Vladimir Vapnik a développé une théorique statistique de l'apprentissage avec des fondements solides qui est à l'origine des SVM et qui explique les nombreux résultats expérimentaux de qualité obtenus avec cet algorithme.
L'exploitation des SVM n'est pas pour autant triviale pour différentes raisons :

1. Les SVM définissent un cadre général à l'apprentissage et ils nécessitent (comme beaucoup d'autres algorithmes d'apprentissage) de choisir plusieurs paramétrages leur étant liés. Si ce choix n'est pas correctement réalisé leurs capacités de généralisation peuvent être très médiocres. La recherche des bons paramétrages est désignée comme l'étape de sélection de modèles et elle est cruciale pour les séparateurs à vaste marge.
2. Les temps d'entraînement des algorithmes implémentant les SVM ont une croissance quadratique (en première approximation) avec le nombre d'exemples. Cela peut les rendre inexploitables lorsque les bases de données sont de tailles conséquentes et que l'entraînement des SVM est réalisé directement à partir d'elles.
3. Les fonctions de décision produites ont une complexité qui est proportionnelle à la taille de la base d'entraînement, les temps de décision peuvent donc devenir inexploitables lorsque des contraintes de temps réel existent.
4. Ils sont définis pour résoudre, dans leur forme de base, des problèmes à deux classes. Il est donc nécessaire d'utiliser des méthodologies réalisant leur combinaison pour résoudre des problèmes d'apprentissage qui ont, dans la grande majorité des cas, plus de deux classes.

L'objectif de cette thèse [1] est de définir des systèmes d'apprentissage à base de SVM performants. Ces systèmes doivent prendre en compte le fait que les problématiques liées au traitement et à l'analyse d'images puissent rentrer en conflit avec les difficultés d'ex-

1. Cette thèse a été cofinancée par l'association Cœur Cancer et par le conseil régional de Basse Normandie (2 ans) et par le bénéfice d'un poste d'A.T.E.R à plein temps (2 ans).

ploitation des SVM relatées précédemment. Plusieurs de ces problématiques s'inscrivent dans le cadre plus général de la fouille de données, de la définition de processus décisionnels en temps réel, de l'optimisation de problèmes difficiles et de la combinaison d'ensembles de fonctions de décision. Les approches proposées dans cette thèse pour résoudre des problèmes de natures différentes pourront être exploitées dans d'autres domaines où les mêmes problématiques sont rencontrées. Nous nous sommes particulièrement intéressés à:

1. Définir de nouvelles méthodes pour réduire les temps nécessaires à l'estimation des performances d'un SVM à partir de techniques de validation croisée en prenant en compte les particularités d'une implémentation particulière des SVM.
2. Etudier la possibilité de réduire la complexité des fonctions de décision en réduisant le nombre d'exemples nécessaires à l'entraînement d'un SVM afin de réduire les temps de décision. Deux approches sont proposées. La première se concentre sur la définition d'une méthode de sélection d'un sous-ensemble d'exemples les plus pertinents à partir d'un principe de minimisation du risque structurel. La seconde se concentre sur la production d'un ensemble réduit de prototypes représentatifs des données afin d'optimiser un compromis entre complexité de la fonction de décision et pouvoir de généralisation (la sélection d'attributs est également possible avec cette deuxième approche).
3. Définir un schéma hybride de combinaison de classificateurs binaires à partir de schémas de combinaison classiques afin d'améliorer les capacités de généralisation sur des problèmes comportant un nombre important de classes.
4. Proposer une nouvelle méthode basée sur un algorithme évolutionnaire pour optimiser la sélection de l'ensemble des modèles propres à chaque SVM intervenant dans un schéma de combinaison particulier.

Ces méthodes se situent dans le cadre plus général de la sélection de modèles pour des systèmes d'apprentissage à base de SVM (*cf.* section 1.3).

Les méthodes produites ont été utilisées dans un cadre applicatif lié au domaine médical[2] et à la qualité visuelle des images. La première contribution, relative au traitement des images, correspond à la réalisation d'un schéma de segmentation rapide d'images de microscopie médicale. Cette segmentation est basée sur l'exploitation d'un classificateur rapide de pixels produit grâce à notre méthode de réduction de la complexité des processus décisionnels à partir du prototypage de la base d'apprentissage. La seconde contribution, relative à la reconnaissance des formes, a pour but d'améliorer les capacités prédictives d'un système de tri cellulaire destiné à la reconnaissance de cellules cancéreuses. Ces deux contributions sont liées au développement d'une application de traitement et analyse d'images dont la finalité est l'élaboration d'un système de vision en imagerie microscopique médicale. Une autre application, relative à l'analyse de la qualité visuelle des images, a été réalisée. Cette application a pour but de modéliser des *experts informatiques* spécialisés dans la reconnaissance de la qualité visuelle d'images couleur compressées avec la norme JPEG2000. Nous avons défini un système de caractérisation basé sur le système visuel humain pour produire une base d'apprentissage dédiée à ce problème d'expertise. A partir de cette base de données nous avons défini le protocole d'apprentissage relatif à la création de cet expert.

2. A travers une collaboration active avec le service d'Anatomie et Cytologie Pathologie du CHLP de Cherbourg.

1.1 Plan de la thèse

Ce document est décomposé en huit chapitres et trois annexes. Les 8 chapitres peuvent être regroupés en 5 parties distinctes:

Partie 1 : le chapitre 1 expose le cadre de cette thèse, l'organisation du document, l'historique de son déroulement et les éléments principaux d'un système d'apprentissage supervisé.

Partie 2 : les chapitres 2 et 3 rappellent les notions essentielles relatives aux problématiques de l'apprentissage supervisé et de l'optimisation par méta-heuristiques. Les notions les plus en relation avec la partie 3 de la thèse sont les plus développées.

Partie 3 : les chapitres 4 à 6 traitent des différentes approches proposées pour améliorer la constitution d'un système d'apprentissage performant à base de SVM. Pour chacun de ces trois chapitres, la problématique abordée est exposée, un état de l'art relatif à cette problématique est réalisé, les différentes approches proposées sont décrites et des résultats expérimentaux relatifs à ces approches sont présentés. Chacun de ces chapitres propose des perspectives relatives à chaque thématique d'apprentissage abordée.

Partie 4 : le chapitre 7 aborde deux cadres applicatifs où plusieurs des méthodes proposées dans les chapitres 4 à 6 sont mises en œuvre. Plusieurs résultats expérimentaux mettent en évidence les bénéfices de ces méthodes pour les deux applications.

Partie 5 : le chapitre 8 propose une conclusion dans laquelle une synthèse des apports de cette thèse est réalisée et des perspectives plus générales propres à l'élaboration de systèmes d'apprentissage encore plus performants sont proposées.

La **partie 3** de ce document contient l'essentiel des apports de cette thèse relatifs au développement de systèmes d'apprentissage supervisé à base de SVM. Nous avons fait le choix de donner le maximum de détails pour que l'ensemble des procédures expérimentales puissent être facilement reproduites et appliquées à d'autres domaines. Un grand nombre de résultats d'expérimentations sont présentés pour mettre en évidence l'efficacité des méthodes proposées.

La **partie 4** de ce document contient l'essentiel des apports relatifs aux deux cadres applicatifs que sont l'imagerie médicale microscopique et la qualité visuelle des images compressées. Plusieurs systèmes d'apprentissage proposés dans la **partie 3** de ce document sont mis en œuvre pour ces applications.

L'annexe A présente les bases de données utilisées pour cette thèse. L'annexe B propose des informations supplémentaires sur la caractérisation du système visuel humain. L'annexe C donne la liste des publications réalisées pendant le déroulement de cette thèse en tant que premier ou second auteur.

1.2 Un bref historique

La rédaction d'une thèse nécessite de choisir une organisation du document qui ne reflète pas forcément le cheminement des idées, les choix effectués et les motivations poursuivies pendant son déroulement. Le bref historique qui suit permet de mieux appréhender cette chronologie.

Cette thèse a débuté le 1er octobre 2002 et s'inscrit dans le prolongement des thèses d'Olivier Lézoray [LEZORA00] et de Cyril Meurie [MEURIE05A] . La thèse d'Olivier Lézoray a porté sur la segmentation d'images couleur par morphologie mathématique et classification de données par réseaux de neurones. Le domaine d'application correspondait à la définition d'un prototype logiciel permettant la segmentation et la classification de cellules d'images médicales en cytologie des séreuses. La thèse de Cyril Meurie a porté sur la segmentation d'images couleur par classification pixellaire (1ère partie) et sur l'étude des hiérarchies de partitions (2nde partie). Le domaine d'application de la première partie de sa thèse était essentiellement axé sur la définition de différentes stratégies pour améliorer la segmentation des images médicales précédemment citées.

Le Professeur Hubert Cardot m'a proposé de m'intéresser à la réalisation de systèmes d'apprentissage performants à partir de SVM. Ces systèmes devaient également permettre de prendre en compte les problématiques liées au projet de vision et d'analyse d'images microscopiques précédemment cité. Cyril Meurie travaillant sur des problèmes de classification pixellaire supervisée et non supervisée, nous avons alors travaillé ensemble sur la partie supervisée. Je me suis plus particulièrement intéressé aux SVM pour réaliser cette classification pixellaire. Cette collaboration a été l'occasion de me rendre compte que l'apprentissage avec des SVM sur une base de pixels (constituée à partir de plusieurs images) était problématique à cause de sa taille. Pour rendre l'apprentissage possible dans des temps raisonnables, elle a donc été échantillonnée. Les résultats expérimentaux ont montré que les SVM permettaient de définir des classificateurs de pixels performants, mais que les temps de décision étaient trop importants s'ils étaient mis en rapport avec la réduction du taux d'erreur obtenue [MEURIE03A, MEURIE03B], et cela malgré un échantillonage préalable de la base.

Il m'est alors apparu évident que pour exploiter les SVM dans des schémas de segmentation pixellaire avec des temps de décision acceptables, il était nécessaire de réduire le nombre d'exemples pour l'entraînement des SVM car cela aurait une incidence sur la complexité des fonctions de décision produites. A première vue, cela est contradictoire avec le principe d'inférence des SVM car plus il y a d'exemples et plus le risque d'erreur de la fonction de décision produite sera proche du risque d'erreur réelle. Choisir un ensemble de plus en plus réduit d'exemples par tirage aléatoire augmentera la variance et n'est évidemment pas la bonne façon de procéder. Il était alors évident que si l'on doit réduire le nombre d'exemples, il faut que les exemples conservés soient le plus représentatifs possible de la répartition des données initiales. Pendant la phase de recherche dans la littérature des différentes façons de procéder, une discussion avec un des membres du laboratoire, Christophe Charrier[3], nous a conduit à considérer que la problématique de sélectionner des exemples représentatifs était très proche de celle consistant à produire un dictionnaire minimisant la distorsion dans le cadre de la compression des signaux. Nous avons donc utilisé la quantification vectorielle pour produire des ensembles de prototypes qui soient représentatifs des exemples initiaux. Dans un premier temps, pour définir une procédure de sélection des paramètres des SVM qui use de la quantification vectorielle pour réduire les temps d'entraînement globaux nécessaires à la sélection d'un modèle [LEBRUN04A, LEBRUN04B]. L'idée directice est que si l'apprentissage avec une base réduite et représentative de prototypes produit une fonction de décision de mauvaise qualité,

3. Il a étudié intensivement la quantification vectorielle dans des problématiques de compression d'images couleur [CHARRI98].

alors le même modèle avec l'ensemble des exemples produirait une fonction de décision d'aussi mauvaise qualité. Cela permet alors d'éliminer rapidement un mauvais modèle, pour ne garder au final que les meilleurs.

Mais au-delà de cette première mise en œuvre de la quantification vectorielle, l'idée principale visait à ce que le niveau de simplification de la base d'entraînement fasse partie de la sélection de modèle afin qu'un "bon compromis" entre la réduction de la complexité des fonctions de décision et la capacité de généralisation soit réalisé. Ceci afin de répondre entre autre au problème de classification rapide de pixels avec des SVM. Le choix de l'espace couleur a une influence sur les capacités de généralisation d'un classificateur de pixels [MEURIE03A, MEURIE03B]. La production d'un classificateur de pixels performant doit donc être réalisée à partir d'un système d'apprentissage qui prend en compte le fait de devoir sélectionner le meilleur espace couleur. Il y a une grande analogie avec la problématique de sélection d'attributs. Pour réaliser un système d'apprentissage produisant un classificateur de pixels qui répond à ces exigences, un nouveau critère de qualité doit être défini à partir d'un compromis entre la complexité des fonctions de décision produites par les SVM et le taux de reconnaissance de ces fonctions. Une méthode d'optimisation par recherche avec tabous a alors été définie pour choisir le niveau de simplification de la quantification vectorielle, l'espace hybride couleur (*i.e.* sélection des composantes de différents espaces couleur) et les hyper-paramètres des SVM afin d'optimiser ce nouveau critère de qualité [LEBRUN05A, LEBRUN05B, LEBRUN05C].

La problématique liée à la définition d'un classificateur de pixels performant est propre à beaucoup de problèmes d'apprentissage. Elle peut être généralisée sous la forme d'un système d'apprentissage supervisé comme l'illustre la section 1.3. Nous nous sommes alors particulièrement intéressé aux différentes problématiques liées à la définition de systèmes d'apprentissage supervisés efficaces. Il était nécessaire d'améliorer le système d'apprentissage proposé pour la classification rapide de pixels. Il a été montré que ce système d'apprentissage est performant pour produire des fonctions de décision réalisant un compromis généralisation/complexité compétitif [LEBRUN06A].

Travaillant avec des SVM qui sont par définition limités à des problèmes à deux classes, nous nous sommes naturellement intéressé à la problématique de combinaisons de classificateurs binaires pour réaliser des processus décisionnels avec des SVM qui permettent d'aborder des problèmes à plus de deux classes. Il était naturel d'utiliser initialement des schémas de combinaison classiques comme le *un-contre-tous* ou le *un-contre-un*. Nous avions des interrogations relatives à ces deux schémas (et aux autres dans une moindre mesure). Pourquoi chercher à les opposer en cherchant lequel est le meilleur ? Ils avaient chacun des qualités et des défauts et une méthode hybride les exploitants a été définis. De plus, pour cette même problématique, les procédures de sélection des hyper-paramètres pour l'ensemble des SVM intervenant dans des schémas de combinaison n'étaient visiblement pas optimales. Par contre le problème d'optimisation auquel nous pensions avait le désavantage d'être difficile et nous nous sommes intéressé aux algorithmes évolutionnaires pour le résoudre.
Nous nous sommes également intéressé au problème de la définition d'un principe de décodage permettant de prendre en compte un ensemble de décisions pouvant être contradictoires au travers de la classification pixellaire à partir de différents espaces couleur [CHARRI06A].

Les systèmes d'apprentissage considérés à base de SVM nécessitaient tous de multiples phases d'entraînement. Nous nous sommes alors intéressé aux techniques d'*alpha seeding* pour permettre de réduire les temps d'apprentissage nécessaires à de tels systèmes, tout d'abord dans le cadre de la réduction des temps nécessaires à l'estimation de l'erreur pour des procédures de validation croisée [LEBRUN06B], mais avec l'idée qu'il est possible de les appliquer par la suite dans un cadre plus générique.

Dans la dernière partie de cette thèse, nous nous sommes intéressé à un autre domaine d'application qui concerne la qualité visuelle des images. Cela a été l'occasion de mettre en œuvre nos connaissances sur la définition de système d'apprentissage supervisé pour produire des processus décisionnels qui reproduisent avec une grande fidélité le comportement d'un panel d'obervateurs pour évaluer la qualité visuelle des images compressées avec la norme JPEG2000 [CHARRI06B].

Dès le début de cette thèse, une idée avait été émise. Elle correspondait à la considération suivante: Il est préférable de supprimer les exemples aberrants de la base d'entraînement, plutôt que de les pénaliser comme cela est fait avec les SVM. L'hypothèse était que cela permettrait d'améliorer les performances en généralisation. Ce n'est que tardivement, à partir de multiples lectures sur différentes problématiques liées à l'apprentissage, que nous nous sommes intéressé à définir des systèmes d'apprentissage qui prennent en compte le fait que tous les exemples dans une base de données n'ont pas la même pertinence et que cela a des conséquences à la fois sur les capacités de généralisation et sur la complexité des fonctions de décision produites. Nous avons alors défini un système d'apprentissage qui puisse identifier les exemples qui sont les plus aberrants pour les écarter.

L'ensemble des réflexions et travaux réalisés a donc conduit à la définition des éléments essentiels qui doivent composer les systèmes d'apprentissage supervisé. L'ensemble de ces éléments ne sont pas totalement indispensables lorsqu'un problème d'apprentissage particulier est abordé, mais il est nécessaire d'en avoir une vision suffisament générique pour pouvoir aborder aisément de multiples problèmes d'apprentissage.

1.3 Systèmes d'apprentissage supervisé

La réalisation d'un processus décisionnel performant à partir d'un ensemble de données étiquetées doit prendre en compte plusieurs considérations liées à la nature des données, à l'algorithme d'apprentissage supervisé utilisé et à l'objectif à produire. La figure 1.1 illustre les élements essentiels des systèmes d'apprentissage supervisé.

Les données d'apprentissage sont généralement plus ou moins fortement bruitées. Les attributs sont plus ou moins fortement pertinents et ils peuvent être redondants entre eux. Tout comme les attributs, les exemples peuvent être plus ou moins pertinents pour deux raisons : 1) les valeurs des attributs de certains exemples sont plus bruitées que d'autres et/ou 2) l'étiquetage n'a pas été réalisée correctement. Les exemples peuvent également être plus ou moins redondants entre eux. Il est donc important de réaliser la sélection des exemples et des attributs les plus pertinents pour faciliter le processus d'apprentissage.

Les exemples d'un problème d'apprentissage appartiennent à différentes classes. Parmi ces classes, certaines peuvent être plus ou moins difficiles à différencier des autres. Prenons l'exemple de la reconnaissance des lettres manuscrites. Pour certaines écritures il sera

difficile de différencier un *a* d'un *o*, mais pas un *a* d'un *k*. Il est alors courant de décomposer le problème d'apprentissage initial en sous-problèmes plus simples comportant moins de classes. L'algorithme d'apprentissage est alors beaucoup plus performant sur chacun de ces sous-problèmes. L'algorithme d'apprentissage supervisé utilisé peut également induire certains types de décompositions. Il est par-contre nécessaire de pouvoir répondre au problème initial à partir des résultats relatifs à ces sous-problèmes. Ceci nécessite la définition d'une procédure de décodage capable de prendre une décision finale à partir de décisions partielles.

La sélection d'attributs, la sélection d'exemples et la décomposition en sous-problèmes sont liées à la nature des données, mais également à l'objectif à réaliser. Dans le cas de la définition d'un système d'apprentissage supervisé, l'objectif principal est de produire un processus décisionnel qui commette le moins d'erreurs possibles. Cependant, le fait que les temps de décision restent exploitables peut être également aussi important que la réduction du nombre d'erreurs. Il est alors nécessaire de définir un compromis entre ces deux objectifs et nous le désignerons par le terme de *qualité du processus décisionnel*.

Un système d'apprentissage doit réaliser l'optimisation d'un problème qui regroupe la sélection des exemples[4], la sélection des attributs[5], la sélection des paramètres relatifs à l'exploitation d'algorithmes d'apprentissage supervisé, la structure de la décomposition

4. Dans cette désignation générique nous incluons la possibilité de synthétiser de nouveaux exemples à partir des premiers.

5. La possibilité d'extraire de nouveaux attributs doit également être prise en compte.

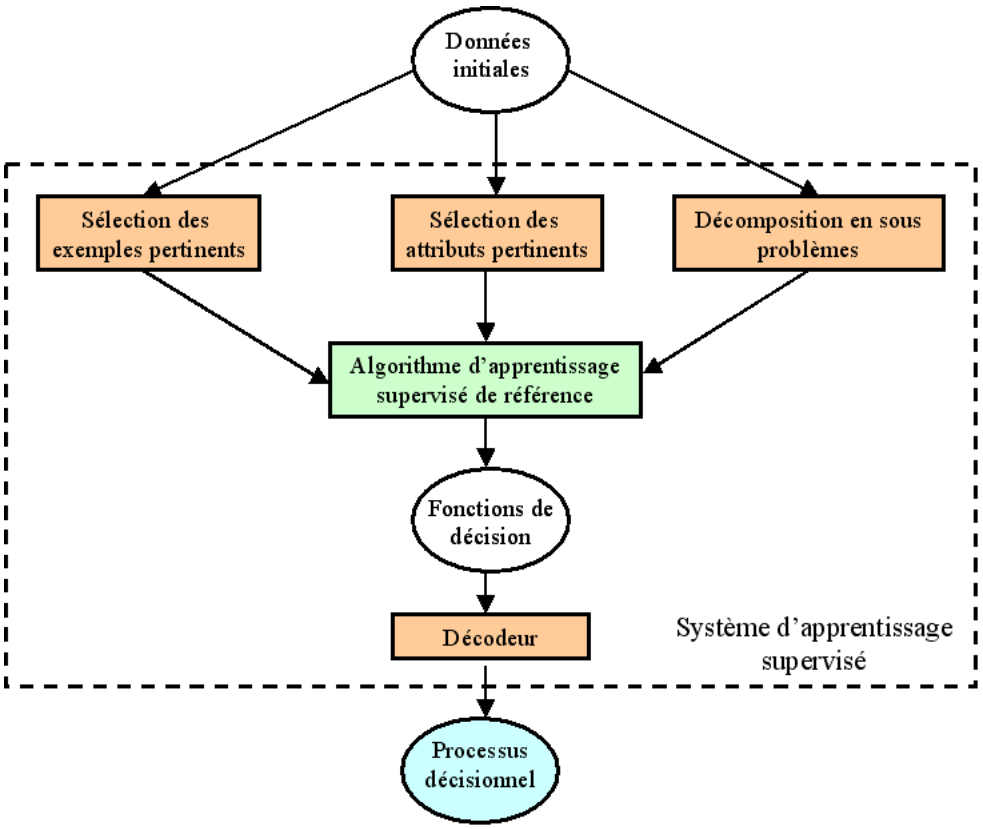


FIG. 1.1 – *Les différents composants d'un système d'apprentissage supervisé.*

en sous-problèmes et le principe de décodage mis en œuvre. Chaque sous-problème peut nécessiter une sélection d'exemples et d'attributs qui lui est propre et donner naissance à une nouvelle décomposition. Nous regroupons l'ensemble de ces sélections sous le terme générique de *sélection de modèles* (ou *sélection multi-modèle* si nous voulons insister sur la pluralité des sélections à réaliser). La sélection d'un modèle optimisant la qualité du processus décisionnel correspond donc à un problème d'optimisation difficile et deux approches sont possibles pour réussir à gérer cette complexité. La première consiste à définir un système d'apprentissage en cascade, où chaque étape ne réalise que la sélection d'un des modèles; l'ensemble de ces étapes réalisant la sélection complète du multi-modèle. Nous sommes alors dans une approche de type *diviser pour régner*. La seconde approche s'appuie sur l'exploitation d'heuristiques ou de méta-heuristiques pour trouver une solution (multi-modèle) proche qui optimise globalement le problème. Ces deux types d'approches (ou une hybridation entre les deux) permettent également de contrôler les temps d'apprentissage nécessaires à la production du processus décisionnel. Lors de la phase d'apprentissage, l'évaluation de la qualité d'un modèle particulier peut demander des ressources en temps de calcul importantes. Il est alors nécessaire de réduire ces temps au maximum pour produire des systèmes d'apprentissage exploitables.

Les approches proposées dans les chapitres 4 à 6 correspondent à l'étude des sous-problèmes relatifs à la constitution de tels systèmes d'apprentissage. *L'algorithme d'apprentissage supervisé de référence* (*cf.* figure 1.1) est majoritairement celui des SVM (et dans une moindre mesure la règle du plus proche voisin). L'ensemble des approches proposées permet d'aider à la réalisation d'un tel système. Bien entendu, ce système d'apprentissage supervisé se veut suffisamment générique pour être adapté à un grand nombre de domaines nécessitant son utilisation. Le chapitre 7 montre que les approches proposées peuvent être adaptées au domaine du traitement et de l'analyse d'images.

CHAPITRE 2

APPRENTISSAGE SUPERVISÉ

Sommaire

2.1 Introduction

Dans notre activité de tous les jours, nous utilisons des processus décisionnels que nous avons appris plus ou moins facilement et régulièrement et nous faisons des prédictions à partir des observations du monde qui nous entoure. A l'aide de ces différents processus décisionnels, l'homme n'a cessé de classer les objets avec lequel il doit composer, que ces objets soient physiques ou abstraits. De génération en génération, l'homme se transmet cette représentation du monde en grande partie par l'éducation : l'apprentissage est alors une phase importante qui peut-être d'une grande complexité. Un médecin observant une radiographie pourra diagnostiquer un problème de santé, alors que pour une personne n'ayant pas appris à lire une radiographie, le diagnostic est pratiquement impossible. Avec l'avènement de l'informatique, s'est posé la question de savoir si une machine pouvait, tout comme les êtres humains, inférer la classe d'appartenance d'un objet à partir de mesures lui étant associées. La réalisation d'algorithmes d'apprentissage artificiel est alors devenu un objectif majeur.

Dans la plupart des cas, la classe d'appartenance d'un objet n'est pas facilement et directement déductible : si c'est le cas il n'est pas utile de réaliser l'apprentissage automatique de la relation d'inférence et nous pourrions coder directement la règle (fonction) de décision. La réalisation d'un apprentissage dans un cadre supervisé nécessite le recours à un processus annexe que nous désignons ici par le terme d'oracle. Cet oracle peut déterminer de façon non infaillible la classe d'un objet et son processus décisionnel n'est pas directement explicable. Notons qu'à ce terme d'oracle d'autres désignations pourront être préférées comme : superviseur, professeur, expert, vérité terrain. Par exemple, en microscopie, un expert humain peut faire office d'oracle et réaliser la labellisation d'objets cellulaires présents sur des images. La notion d'oracle peut également correspondre à une

évaluation a posteriori. Par exemple, le fait qu'une pièce mécanique ait ou non un défaut peut être déterminé *a posteriori* en réalisant un test de résistance à certaines contraintes mécaniques. L'objectif de l'apprenant est alors de déduire l'existence d'un défaut à partir des attributs caractérisant une pièce afin d'éviter par la suite la réalisation de tests mécaniques. Les performances de l'apprenant à réaliser une fonction de décision qui a une bonne qualité prédictive déterminent évidemment la viabilité de la fonction de décision et la confiance placée en l'apprenant. Le but n'est pas de réaliser forcément un processus décisionnel qui soit parfait (*i.e.* qui ne fait jamais d'erreur sur la classe intrinsèque d'un objet), mais d'obtenir qu'il soit le plus éloigné possible d'une décision purement aléatoire. De plus, l'oracle n'utilise pas systématiquement les attributs caractérisant les objets, alors que l'apprenant artificiel n'a accès qu'à ceux-ci. Les performances décisionnelles dépendent donc fortement de la qualité des attributs caractérisant les objets, de la qualité de prédiction de l'oracle associé, de la difficulté intrinsèque au problème de classification étudié, ainsi que des capacités inductives de l'apprenant artificiel.

Quels sont les facteurs qui entrent en jeu dans l'apprentissage et déterminent le résultat final produit par un algorithme d'apprentissage? Comment caractériser que le résultat produit généralise bien les concepts sous-jacents relatifs aux données d'apprentissage ? Quels sont les différents principes inductifs à mettre en oeuvre pour choisir la meilleure hypothèse, en rapport avec ces données, parmi un ensemble d'hypothèses ?

Plusieurs concepts, notations et résultats en rapport avec ces interrogations sont rappelés dans les sections suivantes. La section 2.2 perment d'introduire des notions essentielles comme espace d'hypothèses, risque réel, rique empirique, biais d'apprentissage et principes inductifs. Des notations relatives aux bases de données et à leurs constitutions sont proposées. Dans la section 2.3 les résultats principaux de la théorie de l'apprentissage de Vapnik sont rappelés. En particulier ceux relatifs à la notion de VC-dimension et de son lien avec le principe de minimisation du risque structurel. Le section 2.4 revisite plusieurs algorithmes d'apprentissage tels que les k plus proches voisins, les fenêtres de Parzen, les arbres de décision, les réseaux de neurones et les SVM. Les deux algorithmes les plus détaillés sont les k plus proches voisins et les SVM car ils sont exploités dans cette thèse. Certains détails liés aux algorithmes des SVM ne sont pas abordés dans cette section, car ils le seront dans le chapitre 4. Pour l'ensemble des algorithmes de cette section, nous insistons sur l'importance de contrôler la richesse de l'espace d'hypothèses et sur les façons de réaliser ce contrôle. La section 2.5 insiste sur l'importance de définir des critères de confiance pour mesurer la qualité d'une hypothèse au delà d'une simple estimation de son taux d'erreur. La section 2.6 donne les notions essentielles à l'estimation de l'erreur de généralisation à partir de techniques de validation croisée. Ce chapitre se termine par une brève conclusion.

2.2 Espace d'hypothèses, biais d'apprentissage et principes inductifs

Le pouvoir de généralisation d'un algorithme d'apprentissage artificiel est dépendant du processus inductif qu'il réalise et de l'espace des hypothèses. Cet espace correspond à l'ensemble des fonctions de décision réalisables. Le principe inductif permet de sélec-

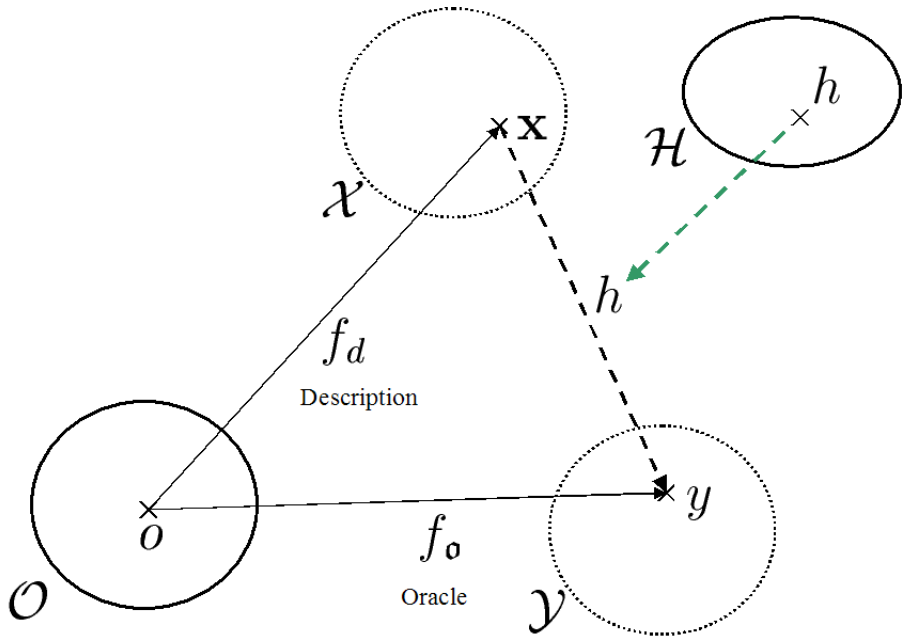


FIG. 2.1 – *Pour un problème d'apprentissage donné, un objet $o \in \mathcal{O}$ représentatif de ce problème est décrit par un vecteur d'attributs $\mathbf{x} \in \mathcal{X}$ et est identifié par l'oracle $\mathfrak{o}$ comme étant de la classe $y \in \mathcal{Y}$. Dans ce schéma f_d et $f_\mathfrak{o}$ représentent respectivement la procédure réalisant la description d'un objet o et le processus décisionnel de l'oracle $\mathfrak{o}$. L'objectif d'un apprenant est de choisir une hypothèse $h \in \mathcal{H}$ dont les prédictions sont le plus proche possible de l'oracle.*

tionner dans l'espace des hypothèses, à partir d'un ensemble de données, celles explicitant le mieux ces données. Ces deux concepts représentent le biais d'apprentissage utilisé par l'apprenant artificiel pour produire une fonction de décision avec les meilleurs capacités de généralisation. Des notations et concepts relatifs aux notions d'espace d'hypothèses, de principe inductif et de biais d'apprentissage sont introduites pour permettre de formaliser les principes et objectifs de l'apprentissage supervisé. Ces notations et concepts sont utilisés ensuite dans les chapitres suivants. Des notations relatives à la constitution de bases d'exemples utilisées dans le cadre d'un apprentissage supervisé sont également introduites.

2.2.1 Objets, exemples et oracle

Pour réaliser un processus d'apprentissage à partir d'un ensemble d'objets, la description et la classe de ces objets doivent être accessibles. La figure 2.1 illustre les différentes relations entre objets, la description des objets et les classes prédites par l'oracle. Soit $\mathcal{O}$ l'espace des objets, $\mathcal{X}$ un espace de description (ou espace des entrées) associé à ces objets et un oracle $\mathfrak{o}$ capable de réaliser une catégorisation des objets issus de $\mathcal{O}$. Dans cette étude, la description d'un objet $o \in \mathcal{O}$ est un vecteur $\mathbf{x} = (x_1, \ldots, x_n)$ à n dimensions. Chaque composante x_i de ce vecteur est désignée par le terme d'attribut ou caractéristique. Soit $f_d : \mathcal{O} \rightarrow \mathcal{X}$, une fonction qui détermine, pour un objet o donné, sa description $\mathbf{x}$ dans

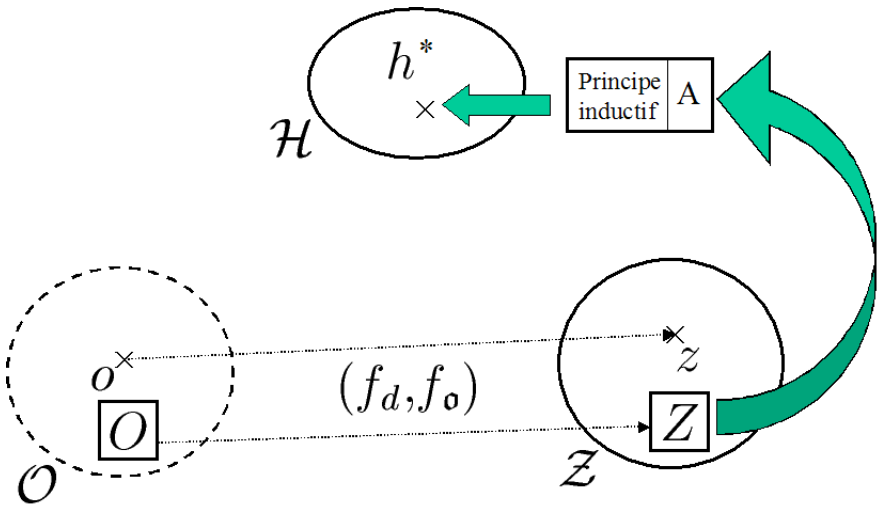


FIG. 2.2 – *Choix de la meilleure hypothèse h^* à partir d'un ensemble fini d'exemples Z et d'un espace d'hypothèses $\mathcal{H}$ lorsqu'un principe inductif (algorithme A d'apprentissage supervisé) est exploité.*

l'espace des entrées. Cette fonction n'est pas forcément déterministe[1], traduisant le fait que le processus de description peut être plus ou moins bruité (capteurs électroniques,...). Soit $f_o : \mathcal{O} \rightarrow \mathcal{Y}$ la fonction d'étiquetage correspondant au processus de catégorisation en n_c classes distinctes de l'oracle avec $\mathcal{Y}$ l'espace des labels des classes $\mathcal{Y} = \{\omega_1, \ldots, \omega_{n_c}\}$ et ω_i la $i^{\text{ème}}$ classe. A partir de $\mathcal{O}$, $\mathcal{Y}$, f_d et f_o il est possible de définir l'espace des exemples $\mathcal{Z}$ (*cf.* figure 2.2). Un exemple $z \in \mathcal{Z}$ correspondant à un objet $o \in \mathcal{O}$ est un couple de données $(\mathbf{x},y)$ tels que $(\mathbf{x},y) = (f_d(o),f_o(o))$. L'oracle qui réalise la labellisation d'un objet n'est pas forcément parfait et il n'a pas forcément un comportement déterministe (*i.e.* il existe des couples d'exemples (x_1,y_1) et (x_2,y_2) dans $\mathcal{Z}$, tel que $x_1 = x_2$ et $y_1 \neq y_2$). Dans ce cas, nous supposerons que pour tout exemple $z = (x,y)$ de l'espace $\mathcal{Z}$, la probabilité conditionnelle $p_{\mathcal{Z}}(y|\mathbf{x})$ est définie relativement à l'espace $\mathcal{Z}$ considéré (*i.e.* le comportement de l'oracle est prévisible en probabilité).

Nous supposons que la classe intrinsèque $\hat{y}$ (ou vraie classe) d'un objet o existe toujours et que la fonction $\hat{f}_o$ réalisant cette labellisation est parfaite. Notons que pour beaucoup de problèmes d'apprentissage, l'oracle parfait représenté $\hat{f}_o$ ne peut être accessible.

2.2.2 Base d'exemples et tirage *i.i.d*

Pour réaliser un apprentissage supervisé, la constitution de bases d'exemples à partir d'un ensemble d'objets est indispensable. Il est important que chaque base d'exemples soit représentative du problème à caractériser, sinon l'inférence ne pourra pas être de qualité. Pour un problème d'apprentissage donné, la fréquence d'apparition ($p_{\mathcal{O}}(o)$) des différents types d'objets n'est pas la même. La probabilité d'observer un vecteur d'attributs $\mathbf{x}$, un exemple z ou une classe y a une certaine distribution qui dépend des objets du problème, de leur caractérisation et de la nature de l'oracle. Notons que ces distribu-

1. Si le fait que la caractérisation d'un objet n'est pas déterministe doit être formalisé, la notation $p_{\mathcal{O}}(\mathbf{x}|o)$ sera utilisée. Elle traduira la probabilité que l'objet o soit caractérisé par le vecteur $\mathbf{x}$.

tions ne correspondent pas toujours au problème naturel associé, mais peuvent avoir été intentionnellement modifiées lors de la collecte des données suivant la nature du problème d'apprentissage que l'on souhaite produire. Nous noterons Z_m une base de données constituée de m exemples tirés aléatoirement suivant la distribution $\mathcal{D}_{\mathcal{Z}}$. Le tirage des exemples est indépendant et identiquement distribué (*i.i.d.*) relativement à cette distribution. O_m désigne les objets utilisés pour la constitution de la base Z_m d'exemples. Dans la suite de ce document, la notion d'objet et d'exemple est souvent non différenciée, bien que d'un point de vue pratique cette différenciation puisse être essentielle. X_m et Y_m sont les notations associées respectivement aux bases de caractéristiques et de classes, lorsque ces informations sont dissociées de la base des exemples Z_m dont elles sont issues. La notation X_m représente également les caractéristiques des objets O_m lorsqu'aucun oracle n'est accessible (dans le cadre d'un apprentissage non supervisé, l'apprenant a seulement accès à X_m). Lorsque la taille de la base n'est pas précisée ou que cette information n'est pas essentielle, les notations Z, O, X ou Y sont utilisées.

2.2.3 Coût relatif à une hypothèse

L'objectif d'un apprenant artificiel A (ou d'un algorithme d'apprentissage supervisé) est d'inférer la meilleur hypothèse h^* à partir d'un ensemble fini d'exemples Z et d'un espace d'hypothèses $\mathcal{H}$ (*cf.* figure 2.2). Si la fonction f_o est dans l'espace d'hypothèse $\mathcal{H}$, l'objectif ultime de l'apprenant est de retrouver cette fonction ($h^* = f_o$). Dans le cas contraire qui est le plus courant, l'objectif de l'apprenant est de sélectionner une hypothèse qui approxime le mieux possible le processus inductif de l'oracle ($h \approx f_o$) ou autrement dit celle dont le coût estimé des erreurs est le plus faible. Dans les deux cas, le problème d'apprentissage correspond à la recherche d'une hypothèse optimale h^*. Un critère de performance doit être défini pour évaluer la qualité de prédiction d'une hypothèse h relativement à une espérance de performance. En premier lieu il est nécessaire de définir une fonction de perte $l(\cdot,\cdot)$ qui caractérise le coût d'avoir prédit $\omega = h(\mathbf{x})$ pour l'exemple $z = (\mathbf{x},y)$, alors que l'oracle a réalisé la prédiction $\omega = y$. La fonction de perte la plus communément utilisée est :

$$l(\omega_1,\omega_2) = \begin{cases} 0 & \text{si } \omega_1 = \omega_2 \\ 1 & \text{sinon} \end{cases} \tag{2.1}$$

et elle permet d'estimer le taux d'erreur. Cette formulation n'est qu'un cas particulier de la formulation générale :

$$l(\omega_1,\omega_2) = \begin{cases} 0 & \text{si } \omega_1 = \omega_2 \\ \texttt{cout}\,(\omega_1,\omega_2) & \text{sinon} \end{cases} \tag{2.2}$$

avec $\forall \omega_1,\omega_2 : \texttt{cout}(\omega_1,\omega_2) \geq 0$. La fonction `cout` correspondant au coût d'avoir prédit la classe ω_1 alors que celle attendue était ω_2 et permet de prendre en compte que tous les couples (ω_1,ω_2) de désaccord ($\omega_1 \neq \omega_2$) ne représentent pas le même risque (prédire un cancer à une personne saine ou la réciproque correspond bien à une erreur dans les deux cas, mais ne représente pas le même risque de mortalité).

L'apprenant a seulement accès à l'espace des exemples $\mathcal{Z}$ à travers une base de données Z. Supposons dans un premier temps qu'il ait accès à l'ensemble de l'espace des exemples

$\mathcal{Z}$ et leurs fréquences d'apparition $\mathcal{D}_{\mathcal{Z}}$. Le risque réel pour l'apprenant, pour une fonction de perte l donnée, correspond à:

$$R_{\text{reel}}(h) = \int_{z=(\mathbf{x},y)\in\mathcal{Z}} l\left(h(\mathbf{x}),y\right) dp_{\mathcal{D}_{\mathcal{Z}}}(\mathbf{x},y) \tag{2.3}$$

L'objectif d'un apprenant artificiel idéal A^* est alors, pour une fonction de coût l et une distribution $\mathcal{D}_{\mathcal{Z}}$ fixée, de choisir l'hypothèse h^* qui minimise le risque correspondant à (2.3), soit:

$$h^* = \arg\min_{h\in\mathcal{H}} R_{\text{reel}}(h) \tag{2.4}$$

Si f_o est compris dans $\mathcal{H}$ alors $h^* = f_o$ et $R(h^*) = 0$. Nous dirons dans ce cas que h^* est cohérente avec l'oracle. Sinon h^* est la fonction qui minimise le risque réel relativement à la fonction de coût choisie. Elle approxime donc au mieux f_o et ainsi le comportement de l'oracle.

$R(h^*)$ représente le risque d'erreur minimale liée à l'hypothèse h^* relativement à l'espace d'hypothèses $\mathcal{H}$ utilisé. Ce risque dépend donc fortement de l'espace d'hypothèses $\mathcal{H}$ utilisé.

L'estimation du risque lié à une hypothèse h à partir d'un ensemble Z_m de m exemples, appelée risque empirique, est égal à :

$$R_{\text{emp}}(h) = \frac{1}{m}\sum_{i=1}^{m} l\left(h\left(\mathbf{x}_i\right),y_i\right) \tag{2.5}$$

avec $(\mathbf{x}_i,y_i) \in Z_m$. La notation $R_{\text{emp}}^{Z_m}(h)$, à la place de $R_{\text{emp}}(h)$, sera utilisée dans ce document lorsqu'il est important d'insister sur le fait que l'erreur empirique est calculée à partir d'un ensemble de données Z_m particulier.
L'hypothèse qui minimise le risque empirique est:

$$h^*_{\text{emp}} = \arg\min_{h\in\mathcal{H}} R_{\text{emp}}(h) \tag{2.6}$$

Elle est généralement différente de h^* et elle peut être très éloignée d'une solution ayant de bonnes capacités en généralisation (*cf.* section 2.2.4). Il y a par contre convergence entre le risque empirique et le risque réel, pour une hypothèse h donnée, lorsque le nombre d'exemples tend vers l'infini.

$$\lim_{m\to\infty} R_{\text{emp}}(h) = R_{\text{reel}}(h) \tag{2.7}$$

Pour un espace d'hypothèses quelconque, il n'y a pas forcément converge entre la sélection de l'hypothèse qui minimise le risque empirique (2.6) et l'hypothèse qui minimise le risque réel (2.4) [CORNUÉ02]. La raison est que lorsque m tend vers l'infini, Z_m sert à évaluer l'ensemble des risques liés aux hypothèses de $\mathcal{H}$ pour sélectionner l'optimal et non comme dans (2.7) à l'évaluation du risque d'une seule hypothèse. Tous les espaces d'hypothèses n'ont pas cette limitation, en particulier lorsque le nombre d'hypothèses de l'espace $\mathcal{H}$ est borné (voir [CORNUÉ02] pour plus de détails).

2.2.4 Compromis biais-variance

L'équation (2.3) donne la solution optimale à un problème d'apprentissage pour une fonction de coût donnée, mais elle suppose que la distribution relative à l'espace des exemples est parfaitement connue. Dit autrement la probabilité $p(\mathbf{x})$ d'observer $\mathbf{x}$ et la probabilité $p(y|\mathbf{x})$ que la classe prédite soit y sachant l'observation $\mathbf{x}$ doivent être parfaitement connues. L'apprenant n'a cependant accès qu'à un nombre restreint d'exemples sous la forme d'une base Z_m de m exemples. Le problème est alors de réaliser un apprentissage efficace à partir de cet ensemble restreint d'objets. Il est évident que l'estimation directe de $p(\mathbf{x})$ et $p(y|\mathbf{x})$ est impossible et ceci d'autant plus que la quantité d'information sera restreinte. Bien entendu, la définition d'attributs fortement pertinents pour d'écrire les objets de la base, ainsi que l'efficacité de l'oracle à labelliser correctement ces objets ont une grande influence sur la constitution de la base d'apprentissage et donc sur l'efficacité de la fonction de décision produite par l'apprenant. Cependant, la difficulté la plus importante est le choix de l'espace d'hypothèses $\mathcal{H}$.

Le première intuition, pour réduire le risque d'erreur, est de chercher à définir un espace $\mathcal{H}$ suffisamment vaste pour permettre de sélectionner une hypothèse la plus proche possible de celle de l'oracle. Mais comment définir cet espace? Peut-on espérer réaliser un apprentissage efficace à partir seulement des données de Z grâce à un espace $\mathcal{H}$ suffisamment vaste? Le compromis biais-variance répond par la négative à cette question [CORNUÉ02, HINNEB02]. Dans les grandes lignes, le compromis biais-variance peut être formulé de la façon suivante : L'erreur de généralisation peut être décomposée en deux termes : un de *biais* et un de *variance*. Ces deux types d'erreur sont liés à l'espace d'hypothèses $\mathcal{H}$ choisi:

– si $\mathcal{H}$ est trop restreint (trop pauvre), aucune solution n'est réellement satisfaisante. La meilleure hypothèse h^* est sélectionnée, bien que sa capacité en généralisation soit faible. L'erreur d'approximation sera due essentiellement au *biais* d'apprentis-

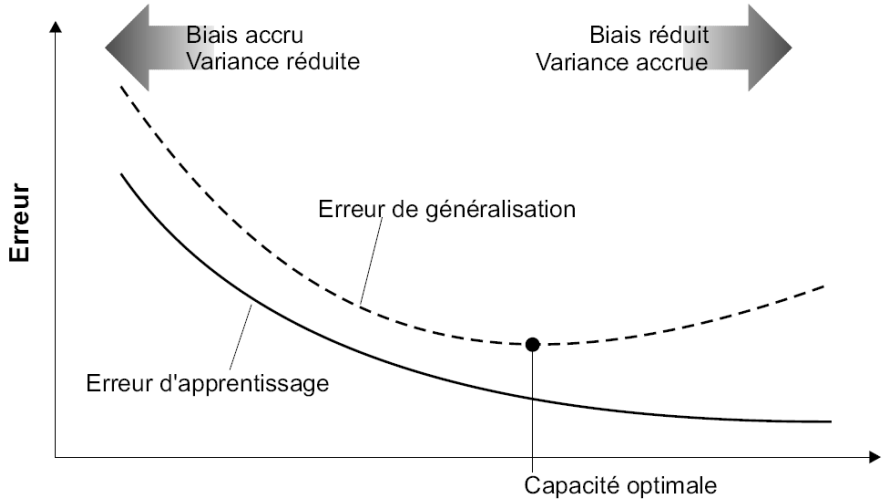


FIG. 2.3 – *Compromis biais-variance pour la sélection d'une hypothèse de capacité de généralisation optimale.*

sage lié au choix de l'espace $\mathcal{H}$. Bien que restreint, si $\mathcal{H}$ est correctement choisi, la solution sélectionnée peut être performante, mais cela nécessite généralement des connaissances importantes sur la nature du problème d'apprentissage alors que l'on souhaiterait réaliser cet apprentissage seulement à partir de la connaissance des valeurs de Z_m,

– si $\mathcal{H}$ est trop vaste (trop riche), il est probable que plusieurs hypothèses aient un comportement adéquat avec le jeu de données Z_m (par exemple, en estimant cette adéquation avec l'équation (2.5)) et l'apprenant n'a d'autre choix que d'en choisir une au hasard. Même si une hypothèse optimale h^* très proche ou identique à f_o existe dans $\mathcal{H}$, l'apprenant a une faible probabilité de la choisir, car il ne peut la caractériser seulement à partir de Z_m. Cette probabilité décroît avec la richesse de $\mathcal{H}$. L'hypothèse choisie peut alors avoir des performances décevantes en généralisation pour de nouveaux exemples alors qu'elle est cohérente avec l'ensemble des données d'apprentissage. Pour différentes données Z_m issues de la même distribution $\mathcal{D}_\mathcal{Z}$, il y a une grande variance entre l'hypothèse sélectionnée et l'hypothèse optimale qui se traduit par un écart important entre le risque réel et le risque empirique. Ce phénomène est désigné par le terme *sur-apprentissage*.

La réalisation d'un algorithme d'apprentissage performant est fortement lié au choix d'un espace $\mathcal{H}$ dont la richesse doit être contrôlée, ce qui correspond à choisir un bon compromis entre biais et variance (voir par exemple [GUIGUE05]). La figure 2.3 en est une illustration. Pratiquement, l'espace des hypothèses exploité par un algorithme d'apprentissage est généralement dépendant de l'algorithme choisi. La définition d'un algorithme d'apprentissage idéal A^*, tel que noté dans la section 2.2.3, n'est pas réellement envisageable, comme le souligne le théorème du No Free Lunch [WOLPER95]. Par contre, l'importance de choisir un biais d'apprentissage approprié à une classe de problème d'apprentissage est indispensable pour obtenir de bonnes capacités en généralisation, comme le souligne Mitchell dans [MITCHE80]. Le choix de $\mathcal{H}$ est un de ces biais d'apprentissage. Un autre point important est de pouvoir estimer les capacités de généralisation d'une hypothèse h avec précision (*cf.* section 2.6), en particulier lorsque l'on sait que la variance peut être importante [TIBSHI96].

2.2.5 Principes inductifs

La définition d'un principe inductif $\mathfrak{I}$ correspond à préciser comment vérifier qu'une hypothèse $h \in \mathcal{H}$ est la meilleure relativement à un ensemble d'exemples Z_m. Généralement, cela correspond à définir une relation d'ordre entre les hypothèses. Le principe inductif ne précise pas les détails algorithmiques pour produire l'hypothèse h^*, ni les problèmes calculatoires qui peuvent en découler, il peut donc exister différents algorithmes qui réalisent le même principe inductif.

Supposons l'existence d'une fonction $R_\mathfrak{I}$ mesurant le pouvoir de généralisation d'une hypothèse donnée h relativement à un principe inductif (par souci d'harmonisation des formulations avec le risque réel (2.4), la valeur de la fonction $R_\mathfrak{I}$ est d'autant plus faible que le pouvoir de généralisation est grand). Soit $\mathfrak{I}$ un principe inductif, la sélection d'une hypothèse optimale $h^*_{\mathfrak{I},Z_m}$ à partir de ce principe peut être formalisée comme :

$$h^*_{\mathfrak{I},Z_m} = \arg\min_{h\in\mathcal{H}} R^{Z_m}_\mathfrak{I}(h) \tag{2.8}$$

avec $R_{\mathfrak{I}}^{Z_m}(h)$ correspondant à l'estimation du pouvoir généralisateur d'une hypothèse donnée h à partir de l'ensemble des données Z_m (et de la fonction de perte l, non mentionnée pour simplifier l'écriture de $R_{\mathfrak{I}}$). Nous noterons $h_{\mathfrak{I}}^*$ à la place de $h_{\mathfrak{I},Z_m}^*$, lorsque la précision de Z_m n'est pas importante.
$R_{\mathfrak{I}}$ est une fonction idéale synthétisant le principe inductif $\mathfrak{I}$ correspondant. L'objectif est de définir un principe inductif $\mathfrak{I}$ tel que la fonction $R_{\mathfrak{I}}$ lui correspondant soit la plus proche possible de R_{reel}. Il n'est pas évident de produire une catégorisation précise des différents principes inductifs, mais trois grandes familles peuvent être mises en évidence [CORNUÉ02]:

- le principe de minimisation du risque empirique,
- le principe de maximum de vraisemblance,
- le principe de compression d'information.

Les 5 principes proposés réalisent un ou plusieurs de ces trois principes de base pour la sélection de la meilleure hypothèse.

2.2.5.1 Sélection de l'hypothèse qui minimise le risque empirique

L'idée sous-jacente est que plus le risque empirique (2.6) est faible et plus le pouvoir généralisateur de l'hypothèse est grand. Cela correspond à considérer que si l'hypothèse h s'accorde aux données fournies, alors elle s'accordera aux données futures. Beaucoup d'algorithmes d'apprentissage utilisent implicitement ce principe : les réseaux de neurones utilisent un espace d'hypothèses vaste qui peut approximer n'importe quelle fonction (celle de l'oracle en l'occurrence).
Le risque de sur-apprentissage fait que ce principe ne peut pas être utilisé directement tel quel pour la majorité des problèmes d'apprentissage. L'évolution des algorithmes à base de réseaux de neurones en est l'illustration [HAYKIN99].

2.2.5.2 Sélection de l'hypothèse la plus probable

L'idée est de sélectionner l'hypothèse la plus vraisemblable. Le principe consiste à définir une distribution de probabilité $p_{\mathcal{H}}(h)$ sur l'espace des hypothèses $\mathcal{H}$ qui correspond à un *a priori* sur les hypothèses. La connaissance des données Z_m modifie la probabilité $p_{\mathcal{H}}(h|Z_m)$ de chaque hypothèse h. La célèbre règle de révision des probabilités est la *règle de Bayes* :

$$p_{\mathcal{H}}(h|Z_m) = \frac{p_{\mathcal{H}}(h) \cdot p_{\mathcal{H}}(Z_m|h)}{p_{\mathcal{H}}(Z_m)} \tag{2.9}$$

Comme $p_{\mathcal{H}}(Z_m)$ est indépendant de h, le principe inductif correspondant au choix de l'hypothèse la plus probable est :

$$h_{\mathfrak{I}}^* = \arg\min_{h \in \mathcal{H}} \left[p_{\mathcal{H}}(h) \cdot p_{\mathcal{H}}(Z_m|h) \right] \tag{2.10}$$

La difficulté est la modélisation de $p_{\mathcal{H}}(h)$ et $p_{\mathcal{H}}(Z_m|h)$. Certaines sont paramétriques et elles supposent, par exemple, que les classes correspondent à des distributions gaussiennes ou des mélanges de distribution gaussiennes. D'autres ne font aucune hypothèse sur un modèle particulier, mais réalisent une approximation locale de la densité comme c'est le cas avec les fenêtres de Parzen [BABICH96] et les k plus proches voisins [COVER67].

2.2.5.3 Sélection de l'hypothèse qui comprime au mieux l'information

L'idée est d'éliminer les redondances présentes dans les données afin d'extraire les régularités sous-jacentes et de produire une hypothèse à la fois simple et proche des données. Ce principe est souvent désigné sous le terme de principe de parcimonie ou rasoir d'Occam. Prenons l'exemple de deux hypothèses qui ont le même risque empirique (ou la même vraisemblance), alors le principe de parcimonie nous dit de choisir l'hypothèse la plus simple. La difficulté est de pouvoir exprimer la complexité d'une hypothèse, relativement à une autre. Le principe de la minimisation du risque structurel [VAPNIK95], le principe de Longueur de description Minimale de Rissanen [RISSAN78] et des principes liés à l'utilisation de la régularisation [BISHOP92, RAKOTO05] permettent de formaliser cette notion de complexité.
Dans la plus part des cas, l'hypothèse optimale correspond à un compromis entre adéquation avec les données et complexité réduite. Le principe inductif correspondant peut être formalisé de la façon suivante :

$$h^*_{\mathfrak{I}} = \arg\min_{h\in\mathcal{H}} \left[R^{Z_m}_{\text{emp}}(h) + \Omega_{\text{complexe}}(h)\right] \tag{2.11}$$

avec $\Omega_{\text{complexe}}(h)$ une fonction traduisant la complexité de l'hypothèse h. Ce qui correspond à pénaliser les hypothèses complexes. La limitation du nombre de neurones cachés dans les réseaux de neurones ou l'élagage des arbres de décision en sont des illustrations.

2.2.5.4 Sélection de l'hypothèse la plus stable

Un apprenant artificiel doit réaliser une induction à partir d'un ensemble de données *i.i.d*. A partir de la même distribution $\mathcal{D}_{\mathcal{Z}}$, Z_m peut avoir des variations plus ou moins importantes pour différents tirages, pourtant c'est le même concept qui doit être appris. L'idée est de choisir une hypothèse qui soit peu sensible à ces variations. Le principe inductif correspondant peut être formalisé comme:

$$h^*_{\mathfrak{I}} = \arg\min_{h\in\mathcal{H}} \left[R^{Z_m}_{\text{emp}}(h) + \Omega_{\text{stable}}(h,Z_m)\right] \tag{2.12}$$

avec $\Omega_{\text{stable}}(h,Z_m)$ une fonction qui traduit la stabilité de l'hypothèse h relativement à des variations de Z_m ($\Omega_{\text{stable}}(h,Z_m)$ est d'autant plus faible que h est stable).

Une solution pour tester la stabilité est de vérifier que l'hypothèse sélectionnée est peu sensible à la suppression de plusieurs exemples de la base Z_m ou à une perturbation légère des valeurs $\mathbf{x}$ des exemples de Z_m [GRANDV97]. Il y a des connexions avec l'induction par compression (en particulier la régularisation) qui peut produire des fonctions plus stables, néanmoins dans ce cas la parcimonie d'une solution ne soit pas un critère impératif. Par exemple, en combinant un ensemble de fonctions de décision, on peut *gommer* les irrégularités particulières de chaque classificateur et produire ainsi une fonction de décision globale plus stable. La combinaison d'un grand nombre de fonctions de décision est donc opposée au principe de parcimonie [KUNCHE04]. L'utilisation de techniques de validation croisée peut être considérée comme un principe de sélection d'hypothèses stables, bien que dans leurs versions classiques il n'y ait pas de pénalisation explicite.

2.2.5.5 Sélection de l'hypothèse qui minimise le risque structurel

Dans les deux principes précédents, la pénalisation est relative à une hypothèse. Dans le cadre de la minimisation du risque structurel, c'est l'exploitation d'un ensemble d'hypothèses de plus en plus riches qui est prise en compte pour la pénalisation.
Soit une suite d'espaces d'hypothèses $\{\mathcal{H}_d\}_d$ telle que :

$$\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots \subset \mathcal{H}_d \subset \cdots \tag{2.13}$$

Le risque empirique minimum correspondant à (2.6) pour chaque espace $\mathcal{H}_d$ décroît avec l'augmentation de l'indice d, car chaque espace est de plus en plus riche et contient les précédents. Le problème est que plus l'indice d est grand et plus le risque de sur-apprentissage est important. Le principe inductif correspondant à la minimisation du risque structurel est une réalisation du compromis biais-variance et il est formalisé de la façon suivante :

$$h^*_{\mathfrak{I}} = \underset{h\in\mathcal{H}_d, d\in\mathbb{N}^+}{\arg\min} \left[R^{Z_m}_{\text{emp}}(h) + \Omega_{\text{structurel}}(\mathcal{H}_d)\right] \tag{2.14}$$

La difficulté réside dans la construction d'une suite d'espaces vérifiant (2.13) et la définition de la fonction de pénalisation $\Omega_{\text{structurel}}$. La théorie de l'apprentissage de Vapnik [VAPNIK98] à travers la notion de VC-dimension (*cf.* section 2.3.2) donne un cadre solide à sa réalisation.

2.2.5.6 Choix d'un principe inductif

Le choix d'un principe inductif et d'un espace d'hypothèse correspond au deux facettes du biais d'apprentissage (souvent aussi nommé biais inductif). Ces deux aspects sont plus ou moins fortement imbriqués lors de la définition d'un algorithme d'apprentissage.

Actuellement, les algorithmes d'apprentissage se focalisent rarement sur un seul de ces principes, mais les combinent pour définir de nouveaux concepts inductifs. Plusieurs des algorithmes d'apprentissage supervisé présentés en section 2.4 en sont des illustrations. Les SVM et la théorie de l'apprentissage de Vapnik associée en sont surement la meilleure illustation.

2.3 Théorie de l'apprentissage de Vapnik

La théorie de l'apprentissage de Vapnik permet une analyse PAC (Probably Approximately Correct) avec des espaces d'hypothèses de taille infinie grâce à la notion de dimension de Vapnik-Chervonenkis. Les notions importantes, liées à cette théorie, sont rappelées dans les sections suivantes.

2.3.1 L'analyse PAC

L'analyse PAC consiste à considérer un principe d'induction comme exploitable (approximativement correct) si l'écart entre le risque réel de l'hypothèse sélectionnée $h^*_{\mathfrak{I}}$ par le principe d'induction et le risque de l'hypothèse idéale h^* est borné en probabilité et que

cette borne est faible. Ce qui correspond à rechercher dans quelles conditions cette relation est vraie:

$$P\left(\left|R_{reel}\left(h_{\mathfrak{I}}^{*}\right)-R_{reel}\left(h^{*}\right)\right|\geq\varepsilon\right)<\delta \tag{2.15}$$

avec δ correspondant à la borne de cette probabilité et ε l'écart sur le risque réel.
Pour un principe inductif $\mathfrak{I}$ fixé, l'évolution de ε est une fonction de la base Z_m, de la richesse de l'espace d'hypothèses $\mathcal{H}$ et de la tolérance δ. En pratique, une analyse dans le pire des cas est réalisée, ce qui permet de s'affranchir de la distribution des exemples. Ceci peut être modélisé par :

$$\forall\mathcal{D}_{\mathcal{Z}}:P\left(\left|R_{reel}\left(h_{\mathfrak{I}}^{*}\right)-R_{reel}\left(h^{*}\right)\right|\geq\varepsilon(m,\mathcal{H},\delta)\right)<\delta \tag{2.16}$$

avec $\varepsilon(m,\mathcal{H},\delta)$ signifiant que ε est dépendante du nombre d'exemples m, du choix de l'espace d'hypothèses $\mathcal{H}$ et de la tolérance δ. Une analyse PAC peut par exemple étudier comment évolue la taille minimum de l'échantillon m si l'on fixe les valeurs de ε et δ. Ce qui correspond à résoudre le problème:

$$\forall\mathcal{D}_{\mathcal{Z}},|Z|>m:P\left(\left|R_{reel}\left(h_{\mathfrak{I}}^{*}\right)-R_{reel}\left(h^{*}\right)\right|\geq\varepsilon\right)<\delta \tag{2.17}$$

Dans [CORNUÉ02], une analyse PAC correspondant à (2.17) est réalisée. Le problème est celui de la recherche d'une fonction discriminante optimale (ou fonction indicatrice) dans un espace d'hypothèses $\mathcal{H}$ de taille finie. Le principe inductif utilisé est la minimisation de l'erreur empirique. On suppose en plus qu'il existe une hypothèse h^* dans $\mathcal{H}$ telle que $R_{\text{reel}}(h^*)=0$. Le résultat de cette analyse est que la taille de la base d'apprentissage doit vérifier:

$$m\geq\frac{1}{\varepsilon}\ln\frac{|\mathcal{H}|}{\delta} \tag{2.18}$$

On peut observer que pour ϵ et δ fixés, la taille de la base qui permet d'avoir cette confiance augmente avec la richesse de l'espace d'hypothèse considéré, ce qui correspond à la difficulté d'évaluer le pouvoir de généralisation d'une hypothèse à partir de peu de données dans un espace d'hypothèses vaste. Un autre résultat intéressant est que pour une taille de Z fixée, il est possible de borner l'erreur en généralisation de l'hypothèse sélectionnée :

$$P\left(\left|R_{reel}\left(h_{\text{emp}}^{*}\right)\right|\geq\frac{1}{m}\ln\frac{|\mathcal{H}|}{\delta}\right)<\delta \tag{2.19}$$

Il est important de noter que cette borne n'est vraie qu'en probabilité, il existe donc des distributions $\mathcal{D}_{\mathcal{Z}}$ pour lesquelles le risque réel est important (*i.e.* le théorème du *no free lunch* n'est pas contredit). Les bases de données correspondent généralement à des distributions sous-jacentes $\mathcal{D}_{\mathcal{Z}}$ plus simples et la borne (2.19) est donc généralement pessimiste. On remarque également que cette étude ne dit rien si l'espace d'hypothèses est de taille infinie.

2.3.2 Dimension de Vapnik-Chervonenkis

L'analyse PAC en section 2.3.1 pose un problème lorsque l'espace des hypothèses est de taille infinie. Pourtant, la richesse d'un espace d'hypothèses ne doit pas être liée qu'au nombre d'hypothèses. Prenons, par exemple pour analogie l'espace des entiers et l'espace des réels. L'ensemble des entiers $\mathbb{N}$ est inclus dans l'ensemble des réels $\mathbb{R}$. Il

serait donc logique d'exprimer que l'ensemble $\mathbb{R}$ est plus riche que $\mathbb{N}$. Le problème est de pouvoir réaliser cette hiérarchisation pour des espaces d'hypothèses dans le cadre de l'apprentissage supervisé.

L'idée de Vapnik et Chervonenkis est d'établir une relation d'ordre entre des espaces d'hypothèses. Soit $G_{\mathcal{H}}(m)$ la *fonction de croissance* qui correspond au nombre maximal de dichotomies que les hypothèses de $\mathcal{H}$ peuvent réaliser sur un ensemble quelconque de m points[2]. Ce nombre est borné : $G_{\mathcal{H}}(m) \leq 2^m$.

Prenons pour exemple l'ensemble $\mathcal{H}$ des hyperplans dans un espace à n dimensions. Chaque hyperplan réalise une dichotomie de l'espace $\mathbb{R}^n$ en le séparant en deux parties. Pour commencer, partons d'un espace simple à une dimension. Soit $X_m = \{a_1, \ldots, a_m\}$ un ensemble de m points dans $\mathbb{R}$. Les hypothèses dans $\mathcal{H}$ sont de la forme $x > b$ ou $x < b$ avec $b \in \mathbb{R}$ et correspondent à l'ensemble des fonctions de seuil. Sans perte de généralité, supposons que $a_1 < a_2 < \ldots < a_m$. Une hypothèse h de $\mathcal{H}$ réalise une dichotomie de X_m en affectant un label $+1$ ou -1 à chaque point suivant que l'hypothèse est vérifiée ou non. Si pour deux points, il est possible de réaliser les 4 dichotomies, pour m points on ne peut en réaliser que $2m$ (avec 3 points, les dichotomies $+-+$ et $-+-$ ne sont pas réalisables) Dans ce cas $G_{\mathcal{H}}(m) = 2m$. Si l'on choisit maintenant comme espace d'hypothèses $\mathcal{H} = \{\text{sign}(\sin bx + c) | b,c \in \mathbb{R}\}$ avec toujours $X_m \subset \mathbb{R}$, alors $G_{\mathcal{H}}(m) = 2^m$. Le deuxième espace d'hypothèses est donc plus riche que le premier en nombre de dichotomies réalisables sur $\mathbb{R}$.

A partir de la fonction de croissance, Vapnik et Chervonenkis ont défini la dimension de Vapnik-Chervonenkis (VC dimension) d'un espace d'hypothèses $\mathcal{H}$, notée $d_{\text{VC}}(\mathcal{H})$, comme:

$$d_{\text{VC}}(\mathcal{H}) = \max_m \{m \in \mathbb{N} : G_{\mathcal{H}}(m) = 2^m\} \tag{2.20}$$

C'est le plus grand nombre de points (exemples) pour lequel l'ensemble des dichotomies sont réalisables à partir des hypothèses de $\mathcal{H}$ (on utilise aussi le terme de *pulvérisables* par $\mathcal{H}$). Dans l'exemple précédent (l'ensemble des fonctions de seuil), l'ensemble des hypothèses a une VC dimension égale à 2, alors que l'ensemble des hypothèses de type sinus à une VC dimension égale à l'infini. Cette mesure permet donc de caractériser la différence de richesse entre deux espaces d'hypothèses. Revenons à l'ensemble $\mathcal{H}$ des hyperplans dans un espace à n dimensions et notons $\mathcal{H}_n^{\text{hyperplan}}$ l'espace de ces hypothèses associés à l'espace $\mathbb{R}^n$. Précédement, nous avons vu que $d_{\text{VC}}(\mathcal{H}_1^{\text{hyperplan}}) = 2$, la figure 2.4 illustre que $\mathcal{H}_2^{\text{hyperplan}}$ pulvérise un ensemble de trois points, mais pas de quatre points (*cf.* figure 2.5), donc $d_{\text{VC}}(\mathcal{H}_2^{\text{hyperplan}}) = 3$. De façon générale, on a $d_{\text{VC}}(\mathcal{H}_n^{\text{hyperplan}}) = n + 1$.

La richesse d'un espace d'hypothèses $\mathcal{H}$ de dimension infinie peut donc être caractérisée grâce à la notion de VC dimension.

2.3.3 Analyse PAC et VC-dimension

Vapnik et al [VAPNIK98] ont réalisé une analyse PAC dans le cadre où l'espace d'hypothèses est de dimension infinie et chaque hypothèse de cet espace est une fonction in-

2. Chaque exemple de la base X_m est un point dans un espace à n dimensions et $G_{\mathcal{H}}(m)$ correspond au nombre maximum de dichotomies réalisables sur un ensemble $X_m \in \mathcal{X}$ quelconque.

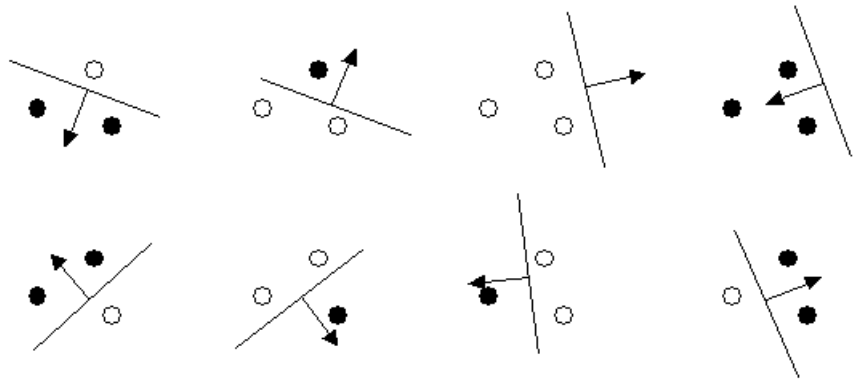


FIG. 2.4 – *Exemple d'ensemble de* pulvérisables *dans* $\mathbb{R}^2$ *de 3 points à partir d'un espace d'hypothèses* $\mathcal{H}_2^{\text{hyperplan}}$.

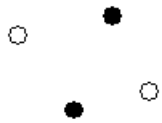


FIG. 2.5 – *Un exemple de l'impossibilité de* pulvériser *4 points dans* $\mathbb{R}^2$ *à partir d'un espace d'hypothèses* $\mathcal{H}_2^{\text{hyperplan}}$.

dicatrice. L'objectif est de caractériser dans quelles conditions l'utilisation du principe de minimisation du risque empirique converge en probabilité vers le risque réel pour cette classe d'espace d'hypothèses et ceci indépendamment de la distribution $\mathcal{D}_\mathcal{Z}$. Nous ne rappelons ici que les résultats les plus importants de leurs analyses, pour une étude plus détaillée des travaux de Vapnik et al, le lecteur peut se reporter à [CORNUÉ02] ou à [CRISTI00], ainsi qu'aux nombreuses références présentes dans ces deux ouvrages.

Premièrement supposons que l'espace $\mathcal{H}$ contient la fonction $f \in \mathcal{F}$ de l'oracle réalisant la labellisation des exemples provenant de $\mathcal{Z}$. Il existe donc $h^* \in \mathcal{H}$ tel que $R_{\text{reel}}(h^*) = 0$ et $R_{\text{emp}}(h^*) = 0$. La question est comment caractériser le fait de choisir une hypothèse h de risque empirique nul alors que son risque réel est supérieur à ε. Cela correspond à déterminer, suivant l'analyse PAC, dans quelles conditions on a :

$$\forall \mathcal{D}_\mathcal{Z} : P\left(\exists h \in \mathcal{H} | Z : R_{\text{emp}}^{Z}(h) = 0 \wedge R_{\text{reel}}(\text{h}) > \varepsilon\right) < \delta \tag{2.21}$$

L'idée consiste à utiliser deux échantillons de m exemples chacun. L'un sert à la sélection d'une hypothèse de risque empirique nul, il est noté Z_m^{app} et correspond à l'ensemble d'apprentissage. L'autre sert à tester la validité d'une hypothèse, il est noté Z_m^{test} et correspond à l'ensemble de test. Notons Z_{2m}, la réunion de Z_m^{app} et Z_m^{test}. L'analyse de (2.21) est remplacée par celle de :

$$\forall \mathcal{D}_\mathcal{Z} : P\left(\exists h \in \mathcal{H} | Z_{2m} : R_{\text{emp}}^{Z_m^{\text{app}}}(h) = 0 \wedge R_{\text{emp}}^{Z_m^{\text{test}}}(\text{h}) > \varepsilon\right) < \delta \tag{2.22}$$

On remarque une certaine analogie avec les procédures de validation croisée. Pour un ensemble Z_{2m} de taille fixée, le nombre de dichotomies est dénombrable et dépend de la fonction de croissance $G_{\mathcal{H}}$. Vapnik et al ont démonté que :

$$\forall \mathcal{D}_{\mathcal{Z}} : P\left(\exists h \in \mathcal{H} | Z_{2m} : R_{\text{emp}}^{Z_m^{\text{app}}}(h) = 0 \wedge R_{\text{emp}}^{Z_m^{\text{test}}}(\text{h}) > \varepsilon\right) < G_{\mathcal{H}}(2m) 2^{-\varepsilon m/2} \tag{2.23}$$

ainsi qu'à partir de (2.21) et de (2.22) :

$$\begin{array}{c} \forall \mathcal{D}_{\mathcal{Z}} : P\left(\exists h \in \mathcal{H} | Z_m^{\text{app}} : R_{\text{emp}}^{Z_m^{\text{app}}}(h) = 0 \wedge R_{\text{reel}}(\text{h}) > \varepsilon\right) < \\ 2P\left(\exists h \in \mathcal{H} | Z_{2m} : R_{\text{emp}}^{Z_m^{\text{app}}}(h) = 0 \wedge R_{\text{emp}}^{Z_m^{\text{test}}}(\text{h}) > \frac{\varepsilon}{2}\right) \end{array} \tag{2.24}$$

et ils ont déduit de (2.23) et (2.24) :

$$\forall \mathcal{D}_{\mathcal{Z}} : P\left(\exists h \in \mathcal{H} | Z_m : R_{\text{emp}}^{Z_m}(h) = 0 \wedge R_{\text{reel}}(\text{h}) > \varepsilon\right) < 2G_{\mathcal{H}}(2m) \cdot 2^{-\varepsilon m/2} \tag{2.25}$$

Si $G_{\mathcal{H}}(2m)$ croît exponentiellement avec m, alors il n'y a pas de convergence en probabilité. Autrement dit, l'utilisation du principe de minimisation du risque empirique avec un espace d'hypothèse trop riche ne devient pas plus pertinent avec l'augmentation du nombre d'exemples. C'est ici que la notion de VC dimension est essentielle. En effet, si elle n'est pas infinie, Vapnik et al ont démontré que :

$$G_{\mathcal{H}}(m) \leq \sum_{i=0}^{d_{\text{VC}}(\mathcal{H})} C_m^i \leq \left(\frac{em}{d_{\text{VC}}(\mathcal{H})}\right)^{d_{\text{VC}}(\mathcal{H})} \tag{2.26}$$

la croissance de $G_{\mathcal{H}}(m)$ est sub-exponentielle lorsque $m < d_{\text{VC}}(\mathcal{H})$ (voir figure 2.6). ,

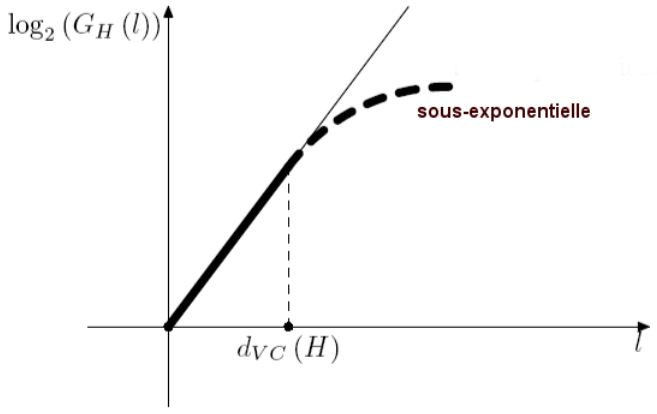


FIG. 2.6 – *Evolution de $G_{\mathcal{H}}$ en fonction du nombre d'exemples (noté l au lieu de m dans cette figure) et de la VC dimension de $\mathcal{H}$ [*CORNUÉ02*].*

en particulier si le nombre d'exemples est suffisant ($m > 2/\varepsilon$) alors une hypothèse h cohérente avec l'échantillon Z_m vérifie :

$$R_{reel}(h) \leq \frac{2}{m}\left(d_{\text{VC}}(\mathcal{H}) \ln \frac{2em}{d_{\text{VC}}(\mathcal{H})} + \ln \frac{2}{\delta}\right) \tag{2.27}$$

avec une probabilité de $1-\delta$.

Dans le cas où l'hypothèse h n'est pas cohérente avec l'ensemble des exemples et réalise m_{err} erreurs de prédiction sur Z_m, Vapnik et al ont déterminé une borne plus générale du risque d'erreur réel [VAPNIK98]:

$$R_{reel}(h) \leq \frac{2m_{\text{err}}}{m} + \frac{4}{m}\left(d_{\text{VC}}(\mathcal{H})\ln\frac{2em}{d_{\text{VC}}(\mathcal{H})} + \ln\frac{4}{\delta}\right) \tag{2.28}$$

Bien que les résultats de cette analyse se limitent aux fonctions indicatrices, les travaux de Vapnik (sur une période de plus de 30 ans) ont permis de lever ces limitations et de produire des résultats plus généraux [VAPNIK95, VAPNIK98]. Ceci conduit, entre autres, à la formulation des SVM basés sur la minimisation du risque structurel [BOSER92].

L'équation (2.28) peut être reformulée de la façon suivante, lorsque le nombre d'exemples m et le taux de confiance $1-\delta$ sont fixés :

$$R_{reel}(h) \leq 2R_{emp}(h) + \Phi\left(d_{\text{VC}}(\mathcal{H})\right) \tag{2.29}$$

La fonction $\Phi(\cdot)$ est forcément strictement croissante. Supposons deux espaces d'hypothèses $\mathcal{H}_1$ et $\mathcal{H}_2$ tel que $d_{\text{VC}}(\mathcal{H}_1) < d_{\text{VC}}(\mathcal{H}_2)$ et $\mathcal{H}_1 \subset \mathcal{H}_2$. Soit h_1 et h_2 respectivement les hypothèses minimisant le risque empirique pour ces deux espaces d'hypothèses. L'augmentation de la richesse d'un espace d'hypothèse ne peut que réduire la valeur de l'erreur empirique ($R_{emp}(h_2) \leq R_{emp}(h_1)$), mais elle ne sera pertinente que si la condition suivante est vérifiée:

$$R_{emp}(h_1) - R_{emp}(h_2) > \left(\Phi\left(d_{\text{VC}}(\mathcal{H}_2)\right) - \Phi\left(d_{\text{VC}}(\mathcal{H}_1)\right)\right)/2 \tag{2.30}$$

Cette constatation est à la base de la définition du principe de minimisation du risque structurel (SRM: Stuctural Risk Minimisation), déjà abordé à la section 2.2.5.5 . L'idée est de choisir une suite d'espaces d'hypothèses qui vérifie la condition (2.13), mais aussi la condition suivante:

$$d_{\text{VC}}(\mathcal{H}_1) \leq d_{\text{VC}}(\mathcal{H}_2) \leq \cdots \leq d_{\text{VC}}(\mathcal{H}_d) \leq \cdots \tag{2.31}$$

Le principe SRM (*cf.* expression (2.14)) peut alors être réécrit à partir de l'équation (2.29) comme :

$$h^*_{\text{SRM}} = \underset{h\in\mathcal{H}_d, d\in\mathbb{N}^+}{\arg\min}\left[R^{Z_m}_{\text{emp}}(h) + \Phi\left(d_{\text{VC}}(\mathcal{H}_d)\right)/2\right] \tag{2.32}$$

La richesse d'un espace d'hypothèse étant caractérisée par sa VC dimension ($\Omega_{\text{structurel}}(\mathcal{H}_d) \equiv \Phi\left(d_{\text{VC}}(\mathcal{H}_d)\right)/2$).

Il est important de noter que, comme les bornes supérieures du risque d'erreur (2.27) et (2.28) sont indépendantes de la distribution $\mathcal{D}_{\mathcal{Z}}$, elles sont très pessimistes. Lorsque qu'un ensemble d'apprentissage Z issu d'une distribution particulière $\mathcal{D}_{\mathcal{Z}}$ est connu, ces bornes supérieures peuvent être réduites. Le principe d'induction à la base des SVM exploite cette possibilité (*cf.* section 4.1.1).

2.4 Algorithmes d'apprentissage supervisés

2.4.1 Les k plus proches voisins

2.4.1.1 Principe général

C'est une des méthodes de classification des plus simples qui donne généralement de bons résultats. Après plus de 35 ans, les classificateurs utilisant la règle des Plus Proches Voisins (PPV) sont encore largement étudiés et utilisés pour résoudre des problèmes de reconnaissance des formes. L'idée de base est d'inférer la classe d'un nouvel exemple en choisissant celle de l'exemple qui lui est le plus proche dans la base d'apprentissage. Ceci correspond à choisir pour un exemple $\mathbf{x}$ la classe $y = \omega_j$ de son plus proche voisin j:

$$D_{\text{PPV}}(\mathbf{x}) = \omega_j |\ j = \arg\min_{1 \leq i \leq m} d\left(\mathbf{x},\mathbf{x}_i\right) \tag{2.33}$$

avec $d\left(\mathbf{x},\mathbf{x}_i\right)$ une métrique permettant de mesurer la distance entre deux exemples.

Pour éviter d'être trop sensible a des données bruitées, il est souvent réalisé la recherche des k plus proches voisins. Notons $N_{\text{voisin}}(\mathbf{x},k,\omega_i,Z_m)$ la fonction retournant le nombre d'exemples de la classe ω_i parmi les k plus proches voisins de $\mathbf{x}$. La classe sélectionnée est celle qui est majoritaire parmi ces k voisins, ce qui correspond à la fonction de décision suivante :

$$D_{k-\text{PPV}}(\mathbf{x}) = \arg\max_{\omega_i \in \mathcal{Y}} N_{\text{voisin}}(\mathbf{x},k,\omega_i,Z_m) \tag{2.34}$$

2.4.1.2 Fondements théoriques

Bien qu'élégante et simple, cette méthode à des fondements théoriques importants liés à une approche probabiliste. Supposons k fixé. Soit un ensemble d'exemples Z_m issu d'une distribution $\mathcal{D}_{\mathcal{Z}}$ correspondant à un ensemble de points dans un espace $\mathbb{R}^n$ qui ont chacun une classe associée. Pour un exemple $\mathbf{x}$, la densité de probabilité $p(\mathbf{x})$ dont il est issu (i.e., $\mathcal{D}_{\mathcal{X}}$) peut être approximée localement comme étant égale à :

$$p(\mathbf{x}) \approx \frac{k}{m \cdot V_R} \tag{2.35}$$

avec V_R le volume de la sphère centrée sur $\mathbf{x}$ contenant les k plus proches exemples de $\mathbf{x}$. Notons k_i le nombre d'exemples de classe ω_i parmi les k plus proches voisins et m_i le nombre d'exemples de classe ω_i dans Z_m. La densité de probabilité $p(\mathbf{x}|\omega_i)$ conditionnellement à la classe ω_i peut être approximée suivant le même principe :

$$p(\mathbf{x}|\omega_i) \approx \frac{k_i}{m_i \cdot V_R} \tag{2.36}$$

En utilisant la règle de Bayes (2.9), il est alors possible d'approximer la probabilité qu'un exemple décrit par le vecteur $\mathbf{x}$ soit de la classe ω_i.

$$P(y = \omega_i|\mathbf{x}) \approx \frac{k_i}{k} \tag{2.37}$$

avec $P(\omega_i) \approx m_i/m$ correspondant à l'approximation de la probabilité d'observer un exemple de classe ω_i. A partir de cette approximation, la classe sélectionnée est celle ayant la plus grande probabilité. L'algorithme des k plus proches voisins est Bayes optimal et si k est fixé à 1, lorsque m tend vers l'infini, l'algorithme du plus proche voisin a un taux d'erreur borné par deux fois l'erreur de Bayes. Bien que ce type d'algorithme ait des propriétés asymptotiques remarquables d'un point de vue théorique, d'un point de vue pratique, il n'est pas exempt de défauts. Le principal est que l'estimation de densité n'est réalisée précisément que si le nombre d'exemples est suffisamment important.

2.4.1.3 Amélioration des capacitées de généralisation

Pour mieux appréhender les problèmes de la règle du plus proche voisin, abordons le comme un problème de recherche d'une hypothèse à partir d'un principe d'inférence de minimisation de l'erreur empirique. La sélection d'une hypothèse est implicite pour la construction du classificateur PPV. Il n'y a pas de phase d'apprentissage à proprement parler lorsque k est fixé. On peut considérer que l'espace des hypothèses est un ensemble de m points dans un espace $\mathbb{R}^n$. Lorsque l'ensemble Z_m est présenté, l'hypothèse sélectionnée est tout simplement Z_m. Si on suppose en plus que k est fixé à 1, l'erreur empirique correspondant à la règle du plus proche voisin est forcément nulle pour cette hypothèse (sauf si des exemples peuvent avoir la même description mais avec des classes différentes). Ceci correspond donc à utiliser un principe inductif de minimisation de l'erreur empirique pour la sélection de l'hypothèse optimale. La possibilité de sélectionner systématiquement une hypothèse d'erreur empirique nulle correspond à utiliser un espace implicite d'hypothèse vaste. La variance est donc grande et la fonction de décision produite est fortement sensible aux variations de Z. Il n'y a donc aucune généralisation à proprement parler avec l'algorithme du plus proche voisin. La fonction de décision produite est donc coûteuse en place mémoire et en temps de décision[3]. Plusieurs méthodes ont alors été proposées pour réduire la variance de la règle du PPV, ainsi que le coût de la fonction de décision.

La méthode la plus simple consiste à utiliser une valeur de k supérieure à un avec la règle *k-ppv* pour atténuer une variance élevée. k agit comme un mécanisme de régularisation en augmentant le biais. Le choix d'une valeur appropriée de k est essentiel et cette valeur est généralement déterminée par validation croisée[4].

Réduction du nombre de prototypes

D'autres méthodes se concentrent sur la réduction du nombre de prototypes (exemples) utilisés par la règle *k-ppv* et réalisent un principe d'inférence basé sur la compression de l'information. L'objectif est d'augmenter le pouvoir de généralisation de la règle *k-ppv* et de réduire ses temps de décision.

La façon la plus simple de produire un ensemble réduit de prototypes est de les sélectionner directement dans Z_m. L'ensemble des méthodes procédant de cette façon sont appelées méthodes d'édition. On peut distinguer deux catégories de méthodes [KUNCHE04] :

3. Bien que ces temps de décision puissent être fortement réduits en utilisant une structure de données comme les *kd-tree* [BENTLE75].

4. Le choix d'une distance particulière pour sélectionner les k plus proches voisins a également une influence non négligable sur la performance du classificateur *k-ppv* [KUNCHE04, GAGNÉ05A].

(1) Celles supprimant les exemples éloignés des frontières de décision et (2) celles supprimant les exemples proches des frontières de décision.
Les méthodes de la première catégorie sont désignées comme des techniques de *condensation* (*condensing techniques*). La méthode proposée par Hart est à l'origine de cette technique [HART68] et de nombreuses variantes ont été proposées par la suite [ANGIUL05, DASARA94]. Dans la plupart de ces méthodes, un exemple est supprimé si l'erreur empirique n'augmente pas en le supprimant. C'est surtout la diminution des temps de décision qui est privilégiée avec ces méthodes, bien que les capacités de généralisation soient influencées par ces suppressions.
Les méthodes de la deuxième catégorie sont désignées comme des méthodes *d'édition de l'erreur* (*error-based editing*). La méthode proposée par Wilson est à l'origine de cette technique [WILSON72] et elle est la base d'une méthode asymptotiquement optimale appelée MULTIEDIT [DEVIJV80] dont plusieurs variantes ont été proposées par la suite [DEVIJV82, DEVI02]. Dans ces méthodes, ce sont les exemples qui présentent le plus d'incertitude que l'on cherche à supprimer. L'objectif est en premier lieu d'augmenter le pouvoir de généralisation et, par la même occasion, de réduire les temps de décision. Elles doivent par contre être utilisées avec précaution, car s'il y a un fort chevauchement entre plusieurs classes, il y a un risque que tous les exemples d'une classe soient supprimés [KUNCHE97A]. Bien que diamétralement opposées, ces deux catégories de méthodes sont généralement complémentaires et sont conjointement utilisées soit en les appliquant séquentiellement, la première suivie de la seconde (ou le contraire) [KUNCHE04], soit en définissant une nouvelle méthode les globalisant [GAGNÉ05A]. La sélection peut également être réalisée en utilisant une technique de validation croisée (*bootstrap editing*) [HAMAMO97].

Une autre possibilité est de déterminer des prototypes qui ne soient pas forcément des exemples présents dans Z_m. Cela est généralement réalisé en utilisant des techniques de classification non supervisées. L'objectif est alors d'identifier des prototypes qui représentent fidèlement un grand nombre d'exemples [XIE93, ODORIC97, BEZDEK98, KOHONE01].

Principe de compression

Quelle que soit la méthode de réduction du nombre de prototypes utilisés par le classificateur *k-ppv*, le problème est alors de choisir un bon compromis entre l'importance de la réduction du nombre de prototypes et l'augmentation de l'erreur empirique. Une adaptation possible du principe de compression (2.11) pour les méthodes *k-ppv* à base de prototypes est [KUNCHE04]:

$$Z_p^* = \underset{Z_p \underset{prototypage}{\leftarrow} Z_m}{\arg\min} \left[R_{emp}^{Z_m}(\text{k-ppv}(Z_p)) + \alpha \frac{p}{m} \right] \tag{2.38}$$

avec $Z_p \underset{prototypage}{\leftarrow} Z_m$ symbolisant le fait que Z_p est un ensemble de p prototypes obtenus à partir de la base Z_m, $R_{emp}^{Z_m}(\text{k-ppv}(Z_p))$ est le risque empirique mesuré à partir de Z_m avec la règle des *k-ppv* lorsque les prototypes utilisés sont ceux de Z_p et α est un coefficient permettant de régler le compromis de réduction. La plupart des méthodes précédemment citées réalisent implicitement cette minimisation sans que le critère à optimiser soit explicitement formulé. Le critère (2.38) (ou des variantes pénalisant la complexité du

modèle) peut être explicitement optimisé avec une méthode de prototypage donnée en utilisant des techniques d'exploration stochastique telles que les algorithmes évolutionnaires [KUNCHE97A, KUNCHE97B, CANO03] ou la recherche avec tabous [CERVER01]. Le choix de α peut traduire une préférence de l'utilisateur, mais il est possible d'utiliser des méthodes métaheuristiques d'optimisation multi-objectifs (*cf.* section 3.2.6) pour éviter un choix arbitraire de la valeur de α [GAGNÉ05A].

2.4.1.4 Principe SRM pour la règle PPV

Nous avons insisté précédemment sur l'importance de réduire la variance de la fonction de décision produite par la règle des PPV en sélectionnant un sous-ensemble réduit de prototypes. Il a été démontré [KARACA03] que la VC-dimension d'un classificateur utilisant la règle du plus proche voisin est égale au nombre d'exemples de référence utilisés par cette règle. Réaliser une induction utilisant le principe de minimisation du risque structurel, dans le cadre de la sélection de prototypes pour la construction d'un classificateur PPV efficace, correspond donc à choisir un compromis efficace entre le nombre de prototypes utilisés et le risque empirique de la règle utilisant ces prototypes. L'équation (2.38) en est une traduction possible. Comme la minimisation de (2.38) (ou une forme équivalente) est un problème de complexité exponentielle avec le nombre d'exemples, Karaçali et al [KARACA03] proposent un algorithme sous-optimal qui permet de rechercher un sous-ensemble réduit de prototypes avec une approche SRM. Pour cela, ils s'inspirent de la manière dont les frontières de décision des SVM sont construites (*cf.* section 2.4.5). Le synopsis de cet algorithme pour un problème de classification binaire est le suivant :

1. Calculer l'ensemble des distances $d(\mathbf{x}_i,\mathbf{x}_j)$ entre les couples d'exemples de classes différentes présents dans la base Z_m et les classer par ordre croissant. Notons i_k et j_k les indices du k ème couple d'exemples relativement à cet ordre.
2. Initialiser $S = \{\mathbf{x}_{i_1},\mathbf{x}_{j_1}\}$ avec les deux exemples les plus proches et initialiser k à 1.
3. Tant que le risque empirique mesuré sur Z_m n'est pas nul pour la règle du PPV établie à partir de S faire: $k = k + 1$ et $S = S \cup \{\mathbf{x}_{i_k},\mathbf{x}_{j_k}\}$ (sans création de doublons!)

Cet algorithme est simple et dans le pire des cas il sélectionne l'ensemble des exemples (la section 5.3.1 montrera comment il peut être amélioré). La comparaison de cette méthode avec les SVM a été réalisée dans [KARACA04] et les résultats présentés montrent son efficacité.

2.4.1.5 Variantes de la règle *k-ppv*

Dans la règle de base des *k-ppv*, la sélection de la classe majoritaire est réalisée par un vote. Chacun des k voisins vote identiquement, quelle que soit la distance qui le sépare de l'exemple concerné. Cette stratégie peut être remise en cause lorsqu'il y a de grandes variations entre la distance du premier et du dernier des k plus proches voisins. Partant de cette remarque, un nouveau concept a été proposé par [DUDANI76] qui a suggéré de donner une plus grande importance au vote du voisin le plus proche de l'exemple $\mathbf{x}$ qu'aux

autres. Il a défini une fonction de pondération, dont la valeur du poids w_j du $j^{\text{ème}}$ voisin diminue avec l'augmentation de la distance :

$$w_j = \frac{d(\mathbf{x}_k,\mathbf{x}) - d(\mathbf{x}_j,\mathbf{x})}{d(\mathbf{x}_k,\mathbf{x}) - d(\mathbf{x}_1,\mathbf{x})} \tag{2.39}$$

avec $\mathbf{x}_1$, $\mathbf{x}_j$, $\mathbf{x}_k$ respectivement les voisins numéro 1, j et k de l'exemple $\mathbf{x}$. Les valeurs de w_j varient de 1 pour le voisin le plus proche à 0 pour le plus éloigné. L'importance du vote $v(\omega_j)$ pour une classe ω_j à partir des poids w_i est :

$$v(\omega_j) = \sum_{i\in\{1,\ldots,k\}\wedge y_i=\omega_j} w_i \tag{2.40}$$

La classe sélectionnée est celle dont le vote est majoritaire. Des variantes ont été proposées par la suite pour déterminer les valeurs des poids w_i [MORIN81, MACLEO87, WANG04]. Une approche floue de la règle des k-ppv a été étudiée et différentes versions proposées [JOZWIK83, KELLER85] : Ces approches permettent de prendre en compte une mesure de confiance concernant chaque classe plausible pour chaque exemple de la base d'apprentissage .

2.4.2 Fenêtres de Parzen

Les fenêtres de Parzen sont une technique simple d'estimation non paramétrique de densités [PARZEN62]. Cette technique approxime une distribution à partir d'un ensemble de données en réalisant une combinaison linéaire de fonctions noyaux centrées sur l'ensemble de ces données. Lorsque la classe y_i de chaque exemple $\mathbf{x}_i$ est connue, la probabilité d'observer x sachant ω_j est estimée à partir de la combinaison linéaire des fonctions noyaux se limitant aux exemples de la classe ω_j considérée:

$$p(x|\omega_j) \approx \frac{1}{m_j \cdot V_j} \sum_{\{i|y_i=\omega_j\}} K(\mathbf{x}_i,\mathbf{x}) \tag{2.41}$$

avec $K(\cdot,\cdot)$ la fonction noyau utilisée, m_j le nombre d'exemples de la classe ω_j et V_j des constantes de normalisation relatives à chaque classe ω_j. Il est alors facile d'estimer $p(\omega_j|x)$ à partir de (2.41) et de $p(\omega_j)$ en utilisant la règle de Bayes (2.9). L'équation (2.41) peut être interprétée comme une moyenne pondérée où chaque exemple $\mathbf{x}_i$ vote avec un

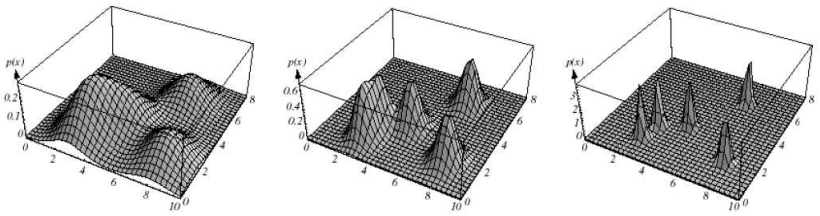


FIG. 2.7 – *Différentes estimations de la densité à partir d'un ensemble de cinq points pour différentes valeurs de la largeur de bande d'un noyau gaussien. [*TAN06*]*

poids qui dépend de sa distance à $\mathbf{x}$. Le choix d'une fonction noyau particulière (et des paramètres associés) est crucial pour les capacités de généralisation du classificateur. Comme la fonction de décision des fenêtres de Parzen utilise l'ensemble des exemples, elle souffre des mêmes défauts que les *k-ppv*. Des améliorations ont alors été proposées pour réduire le nombre d'exemples utilisés par un classificateur de Parzen [FUKUNA89, HOLMST97]. Comme pour les *k-ppv*, des variantes des classificateurs de Parzen utilisant des poids w_i différents pour chaque exemple ont été proposées [HOLMST97, BABICH96]. L'estimation de la densité relativement à une classe devient:

$$p(x|\omega_j) \approx \frac{1}{m_j \cdot V_j} \sum_{\{i|y_i=\omega_j\}} w_i \cdot K\left(\mathbf{x}_i,\mathbf{x}\right) \tag{2.42}$$

La fonction de décision (2.42) a une expression similaire à la fonction de décision produite par un SVM (*cf.* section 2.4.5). Par contre, une grande différence existe dans la méthode algorithmique (*i.e.* principe d'inférence) utilisée pour la détermination des paramètres libres (les poids w_i) et dans le fait que la valeur calculée grâce à la partie droite de (2.42) n'est généralement pas une probabilité.

2.4.3 Arbres de décision

Les arbres de décision correspondent à un ensemble d'algorithmes (CART [BREIMA93], ID3 [QUINLA86], C4.5 [QUINLA93], CHAID [KASS80], etc.) et sont très utilisés depuis de nombreuses années dans le cadre de l'apprentissage supervisé [MITCHE97]. Ils peuvent traiter aussi bien des données représentées par des attributs quantitatifs, des attributs qualitatifs ou des représentations composites (voir figure 2.8). Limitons-nous aux données quantitatives et aux arbres de décision binaires. Ces algorithmes, en plus d'être efficaces pour un grand nombre de problèmes, produisent un processus décisionnel facilement exploitable par un humain. En effet, le résultat est un arbre où chaque nœud est une règle de la forme : si $x_i > a$, alors $d+$ sinon $d-$. $d+$ et $d-$ étant soit un sous-arbre de décision soit une feuille contenant la décision finale à prendre (dans notre cas la classe de l'exemple considéré) et a une valeur de seuil. Un autre avantage est que chaque règle de décision n'exploite qu'un attribut à la fois. L'arbre de décision peut donc n'utiliser qu'un sous ensemble des attributs initiaux et être moins sensible à l'ajout d'attributs non-pertinents. Le problème est alors de définir une méthodologie (pirncipe d'inférence) permettant à chaque étape de la construction de l'arbre de choisir l'attribut le plus pertinent et le seuil de séparation réalisant une des dichotomies. La méthodologie est différente suivant le critère de qualité q utilisé (mesure d'entropie, mesure d'impureté, etc.) pour identifier l'attribut le plus discriminant. Globalement, l'idée de base et que plus on descend dans l'arbre et plus la probabilité qu'un exemple soit d'une classe donnée est accentuée.

Soit un noeud N dans un arbre de décision qui réalise une séparation d'un ensemble d'exemples Z (les données d'apprentissage) en deux ensembles d'exemples Z_{d+} et Z_{d-} à partir d'un seuil a et d'un attribut i. Notons :

$$\delta_N(Z,i,a) = q(Z) - P(x_i > a|Z)q(Z_{d+}) - P(x_i \leq a|Z)q(Z_{d-}) \tag{2.43}$$

la variation de la qualité relativement à cette décision. La sélection de la règle de décision optimale (i^*,a^*) consiste à choisir celle qui maximise (2.43). L'arbre de décision est gé-

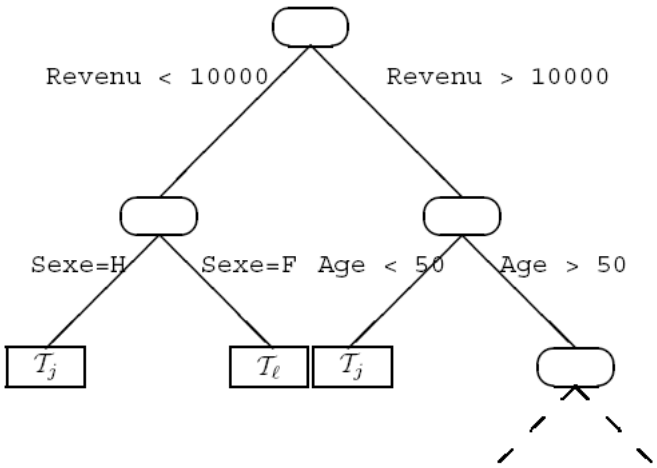


FIG. 2.8 – *Exemple d'arbre de décision exploitant à la fois des données quantitatives et qualitatives.*

néralement construit en appliquant récursivement la règle (2.43) sur les deux sous-arbres produits par la règle précédente.

On peut remarquer qu'en procédant de cette façon, il est toujours possible de définir un arbre de décision tel que le risque empirique sur les données d'apprentissage soit nul (i.e, toutes les données d'apprentissage correspondent à des feuilles de classe unique). Suivant le critère de qualité q utilisé, la nature de l'arbre peut être fortement différente, alors que pour chacun d'eux l'erreur empirique est nulle. De plus, la constuction d'un arbre de décision d'erreur empirique nulle ne garantit pas de bonnes capacités en généralisation. Plusieurs techniques ont été utilisées pour réduire la complexité de l'arbre en appliquant un principe de parcimonie. Une famille de méthodes stoppe prématurément le déploiement de certaines branches de l'arbre si le gain (2.43) n'est pas suffisant (voir dans [KUNCHE04]). Une autre famille de méthodes élague *a posteriori* l'arbre complet construit (*i.e.* d'erreur empirique nulle) si le nœud construit n'est pas suffisamment efficace (voir dans [KUNCHE04]). Récemment ont été proposées des versions évoluées d'algorithmes à base d'arbres de décision exploitant le principe de minimisation du risque structurel et la notion de VC dimension [SCOTT02] afin d'améliorer les capacités de généralisation des arbres de décision produits.

2.4.4 Réseaux de neurones

Les réseaux de neurones artificiels (Artificial Neural Networks) sont des modèles de calcul dont la conception initiale est très schématiquement inspirée du fonctionnement des neurones biologiques [ROSENB58]. Leur objectif initial était de définir un espace d'hypothèses suffisament vaste pour que la modélisation de n'importe quelle fonction de décision soit réalisable. Les sections précédentes ont montré que cette façon de procéder est problématique. Plusieurs directions ont été suivies pour réduire les phénomènes de sur-apprentissage liés à ces méthodes. Actuellement les méthodes connexionnistes sont

très nombreuses et elles partagent des points communs avec les principes d'autres types d'algorithmes d'apprentissage [HAYKIN99, CORNUÉ02, DREYFU04].

2.4.5 Machines à vecteurs de support

Les machines à vecteurs de support (Support Vector Machines) ou Séparateurs à Vastes Marges [5] (SVM) découlent directement des travaux de Vapnik en théorie de l'apprentissage statistique [VAPNIK98, BOSER92]. C'est une méthode de classification supervisée binaire qui a été introduite en 1992. Par la suite, elle a été étendue à des problèmes de régression, d'estimation de densité et de classification non supervisée. Dans un premier temps, nous nous limiterons à la classification supervisée et rappellerons les grandes lignes de la méthode. La section 4.1 décrira de façon plus précise les formulations mathématiques des différents concepts essentiels de cet algorithme d'apprentissage supervisé.

Depuis les années 1995, la recherche à été très prolifique dans l'étude des méthodes à base de SVM [VAPNIK95, PLATT99A, JOACHI00], à la fois dans le domaine pratique et théorique, et de nombreux livres traitant des SVM ont été publiés ces dernières années [VAPNIK98, CRISTI00, HERBRI02, ABE05]. Les raisons d'un tel succès sont les solides fondements théoriques de cette méthode [VAPNIK95, VAPNIK98, HERBRI02], ainsi que la qualité des résultats expérimentaux obtenus [CRISTI00, ABE05].

2.4.5.1 Principe d'induction

Avec les algorithmes d'apprentissage évoqués précédemment, l'objectif initial est de produire une fonction de décision qui minimise l'erreur empirique à travers l'exploitation d'espaces d'hypothèses vastes et de natures différentes. Bien entendu, ils ont été par la suite améliorés pour réduire les risques de sur-apprentissage. Le principe inductif à la base des SVM est celui correspondant à la minimisation du risque structurel.

2.4.5.2 Choix de l'espace d'hypothèses

L'idée principale de Vapnik a été de choisir un espace d'hypothèses particulier qui est l'espace des hyperplans séparateurs. On sait que, pour un ensemble de données à deux classes, si cet ensemble est linéairement séparable, alors il existe un ensemble d'hyperplans les séparant parfaitement. C'est l'ensemble des hypothèses de risque empirique nul. En général il est difficile de caractériser le pouvoir de généralisation de différentes hypothèses de risque empirique nul. Par contre, pour l'espace des hyperplans, il est possible de définir pour ces hypothèses une mesure de leur pouvoir de généralisation. En effet, en plus du taux de reconnaissance sur la base d'apprentissage, il est aussi possible de tenir compte de l'éloignement des exemples de la frontière de séparation correspondant à l'hyperplan. Intuitivement, plus cette distance est grande et plus le risque de mal classer un exemple non-vu est faible. La distance aux exemples les plus proches d'un hyperplan joue donc un rôle important. Cette distance est communément appelée marge géométrique (voir figure 2.9) ou tout simplement marge. La marge joue un rôle analogue à la VC dimension

5. On retrouve cette désignation dans un grand nombre de publications françaises [CORNUÉ02, LOOSLI05]

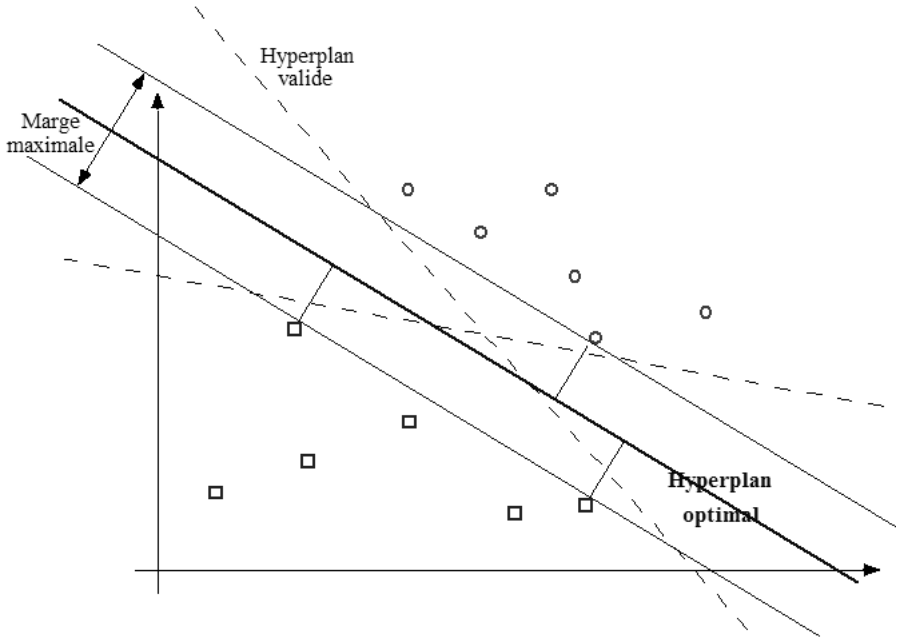


FIG. 2.9 – *Hyperplan optimal et marge géométrique associée.*

pour caractériser la richesse d'un espace d'hypothèses. Notons $\mathcal{H}_\gamma$, l'ensemble des hyperplans séparateurs qui ont une marge géométrique de 2γ relativement à un ensemble de données[6]. Il a été démontré que plus la marge est grande et plus la richesse de l'espace d'hypothèses correspondant est réduite. Soit un ensemble de données Z inscrites dans une sphère de rayon minimale R et un ensemble d'hyperplans $\mathcal{H}_\gamma$ séparant linéairement les données de Z. Il a été démontré que [BURGES98]:

$$d_{\mathrm{VC}}(\mathcal{H}_\gamma) < \min\left(\frac{R^2}{\gamma^2}, n\right) + 1 \tag{2.44}$$

Donc, en respectant le principe de minimisation du risque structurel, à partir d'un ensemble de données pour lequelles on n'a aucune information sur la distribution l'ayant produite, si l'on choisit comme espace d'hypothèses l'espace des hyperplans séparateurs, l'hyperplan de risque empirique nul qui doit être choisi est celui qui a la plus grande marge géométrique γ^*. La solution optimale h^* correspondante est nommée hyperplan optimal.

L'ensemble des remarques précédentes supposent que les données sont linéairement séparables. Que faire si elles ne le sont pas ? Deux idées ont été proposées par Vapnik. La première consiste à projeter les données dans un espace de redescription dans lequel les données sont linéairement séparables. La deuxième est d'accepter qu'un nombre réduit d'exemples puissent être dans la marge, si cela permet de séparer linéairement (dans l'espace initial ou de redescription) les autres exemples .

6. La marge géométrique est celle représentée sur la figure 2.9, dans l'ensemble des formulations suivantes la valeur de γ correspondra à la moitié de cette distance pour rester conforme à la notation utilisée dans plusieurs livres [VAPNIK98, CRISTI00, HERBRI02], même si par abus de langage, nous utilisons dans ce document l'expression : la marge géométrique γ.

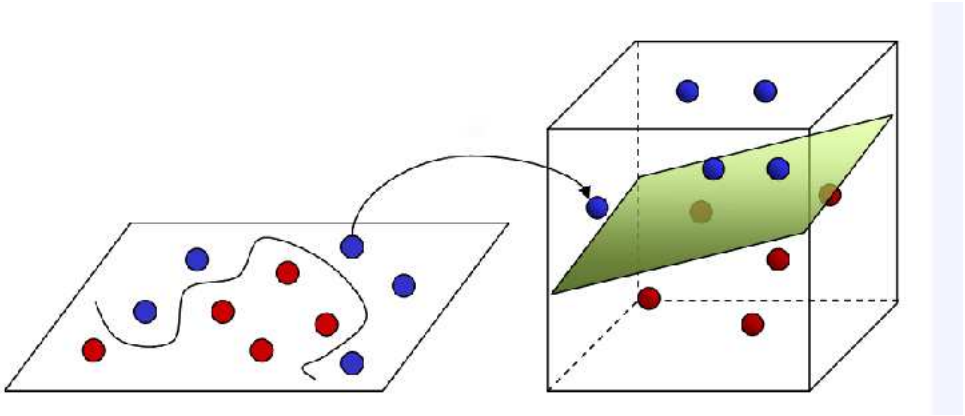


FIG. 2.10 – *Frontière de décision non linéaire dans l'espace initial.*

2.4.5.3 Passage par un espace de redescription

La première idée consiste à choisir une fonction de transformations non linéaire Φ de l'espace d'entrée $\mathcal{X}$ à n dimensions dans l'espace de redescription $\Phi(\mathcal{X})$ à n_Φ dimensions: $\Phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \cdots, \phi_{n_\Phi}(\mathbf{x}))$, puis de rechercher l'hyperplan optimal séparant parfaitement les données dans l'espace $\Phi(\mathcal{X})$. L'avantage est que l'image de l'hyperplan dans l'espace initial des données définit une frontière de séparation qui n'est pas linéaire (voir figure 2.10). Généralement l'espace de redescription est de dimension supérieure à l'espace d'entrée. L'augmentation de la dimension augmente la richesse et donc les possibilités de trouver un hyperplan séparant parfaitement les données. Si $n_\Phi + 1 > m$ l'existence de la séparation linaire de m exemples est garantie (sauf pour un jeu de données dégénéré). Le problème est qu'en augmentant la richesse de l'espace d'hypothèses, la variance augmente et le risque de sur-apprentissage est plus grand. Comme l'algorithme des SVM choisit l'hyperplan de marge géométrique optimale, si la fonction de projection Φ est correctement choisie, l'augmentation du nombre de dimensions ne correspond pas forcément à une augmentation de la VC dimension des hypothèses de marge optimale γ^* (*cf.* 2.44). La difficulté est de choisir une fonction Φ appropriée.

2.4.5.4 Astuce des fonctions noyaux

L'astuce des fonctions noyaux est d'utiliser une fonction qui a la propriété suivante :

$$\forall u,v \in \mathcal{X} : K(u,v) =< \Phi(u),\Phi(v) > \tag{2.45}$$

ou $< \cdot,\cdot >$ est le produit scalaire dans l'espace vectoriel $\mathcal{R}^{n_\Phi}$ avec n_Φ le nombre de dimensions relative à la projection Φ.

Dans ce cas la recherche de l'hyperplan optimal dans l'espace de redistribution peut être réalisé sans avoir à calculer explicitement les projections $\Phi(u)$ [VAPNIK98]. Cela correspond à définir un problème dual d'optimisation quadratique (*cf.* section 4.1) dont le résultat ne dépend que des exemples au contact avec la marge géométrique optimale dans l'espace de redistribution. Ces exemples portent le nom de vecteurs de support (d'où le nom de *Support Vector Machine*). Le nombre d'exemples nécessaires à la détermination de l'hyperplan optimal est généralement beaucoup plus faible que le nombre d'exemples

ayant servis à sa détermination : cela a pour avantage de produire une solution parcimonieuse. La frontière de séparation n'est définie qu'à partir des exemples critiques, définissant par la même la frontière de séparation qui correspond à l'hyperplan optimal. La fonction noyau la plus utilisée dans le cadre de la classification supervisée est le noyau gaussien:

$$K\left(u,v\right) = \exp^{-\frac{\|u-v\|^2}{2\sigma^2}} \tag{2.46}$$

avec σ la largeur de bande du noyau. Elle correspond à une mesure de similarité. D'un point de vue pratique, il est courant de définir une mesure de similarité qui soit adaptée au problème d'apprentissage à réaliser (voir par exemple, le chapitre 8 de [CRISTI00]). Idéalement, si $\forall \mathbf{x}_i,\mathbf{x}_j : K(\mathbf{x}_i,\mathbf{x}_j) = 1$ si $y_i = y_j$ et 0 dans le cas contraire, la définition d'une fonction de décision cohérente devient trivial. Concrètement, on cherche à définir une mesure de similarité s'en rapprochant le plus possible. C'est actuellement un domaine de recherche actif [CRISTI01]. Dans certains cas, la définition de la fonction de similarité est adaptée au problème de classification, mais ne garantit pas que Φ existe pour autant[7]. Plus rigoureusement, il est nécessaire de vérifier que la fonction de noyau utilisée vérifie de bonnes propriétés mathématiques pour que le problème d'optimisation soit convexe (*c.f.* section 4.1.5.1).

Expression de la fonction de décision

La résolution du problème dual d'optimisation quadratique produit un vecteur $\boldsymbol{\alpha}^*$ (*cf.* section 4.1) et à chaque exemple i qui est vecteur de support est associé une variable α_i^* ($\alpha_i > 0$) dont l'ensemble des valeurs définit la position de l'hyperplan optimal h^* dans l'espace de redistribution (une constante b^* est également produite pour que cette localisation soit exacte, *cf.* section 4.1). A partir des valeurs de ces variables, la distance d'un point $\mathbf{x}$ à l'hyperplan est définie par la relation (*cf.* section 4.1):

$$f(\mathbf{x}) = \sum_{i \in SV} y_i \alpha_i^* K\left(\mathbf{x}_i,\mathbf{x}\right) + b^* \tag{2.47}$$

avec SV l'ensemble des indices des vecteurs de support et $f(\mathbf{x})$ est généralement appelée valeur de sortie (*output*) d'un SVM. La fonction de décision correspondante est :

$$D(\mathbf{x}) = \text{sign}\left(f(\mathbf{x})\right) \tag{2.48}$$

La valeur de $|f(\mathbf{x})|$ est utilisée comme une mesure de confiance dans la prédiction et des méthodes ont été proposées pour estimer $P(y = \omega|\mathbf{x})$ à partir des valeurs de $|f(\mathbf{x})|$ sur les données d'apprentissage [PLATT99B, RÜ04].

2.4.5.5 Notion de marge souple

Supposons que, dans une base d'exemples, il y ait des erreurs sur la classe de certains exemples. Choisir un noyau qui permet de définir un espace d'hypothèses suffisamment riche pour être cohérent avec tous les exemples de cette base (par exemple prendre σ très

7. Ces fonctions sont malgré tout utilisées avec l'algorithme des SVM [FAN05]. Les raisons de cette entorse possible au cadre théorique des SVM restent une question ouverte.

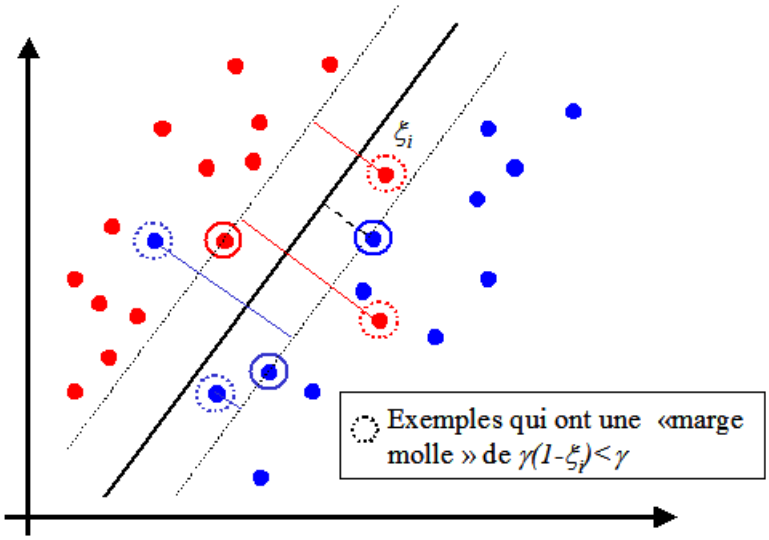


FIG. 2.11 – *Marge souple et slack variable* ξ_i.

faible avec (2.46)) n'est pas un choix judicieux. Le problème est que toute fonction noyau qui permet de trouver une séparation linéaire des exemples de cette base dans l'espace de redistribution n'est pas appropriée. Les contraintes sur la recherche d'un hyperplan optimal doivent être relâchées, si l'on souhaite trouver l'hyperplan permettant de définir une fonction de décision classant correctement les exemples bien étiquetés et incorrectement les mal-étiquetés. Le problème est alors de définir quelles sont les propriétés que doit respecter cet hyperplan par rapport aux données pour identifier celui qui est optimal. Le principe SRM permet de répondre à cette question, sans donner la façon précise de le réaliser. Il faut à la fois minimiser le nombre d'exemples mal-classés (le risque empirique) et maximiser la marge géométrique (*i.e.* diminuer la richesse de l'espace d'hypothèse). Un bon compromis doit donc être réalisé, mais rien ne nous dit comment régler celui-ci.

Vapnik a introduit la notion de marge souple (soft margin) qui correspond toujours à la recherche d'un hyperplan de marge optimale, mais avec une règle d'exception qui autorise que quelques exemples soient à une distance plus faible de l'hyperplan que la marge correspondante. Soit $\xi_i = \max(1-y_i \cdot f(x_i),0)$ (voir figure 2.11), un indice mesurant l'importance de la pénétration de l'exemple i dans la zone définie par l'hyperplan h de marge géométrique γ. Cette variable est appelée variables d'écart (slack variable) (si $\xi_i > 1$, l'exemple n'est pas du bon coté de l'hyperplan relativement à sa classe). L'idée de la marge molle est de rechercher l'hyperplan de marge optimale pénalisée par l'importance des variables ressort. Le terme de marge molle vient du fait que l'on peut considérer que les exemples pour les quels $\xi_i > 0$, ont une marge géométrique réduite de $\gamma(1 - \xi_i)$. Le terme de pénalisation est de la forme $C \sum \xi_i$ avec C une constante qui permet de définir l'importance de la pénalisation. La section 4.1.1.2 donne les détails du problème dual correspondant, mais globalement le problème d'optimisation à résoudre (ainsi que la fonction de décision produite) est de même nature (voir la section 4.1.1.2 pour plus de détails).

2.4.5.6 Les bornes sur l'erreur de généralisation

Un des avantages essentiels des SVM est que la solution produite à partir d'un ensemble d'apprentissage fournit des informations sur l'erreur en généralisation en plus de la définition classique d'une fonction de décision. L'importance de la marge géométrique par rapport au rayon de l'hyper-sphère englobant les données est un indicateur performant de la capacité de généralisation. Le nombre de vecteurs de support utilisés par la fonction de décision par rapport au nombre total d'exemples dans la base d'entraînement est aussi un indicateur performant de la capacité de généralisation.

Bornes qui utilisent la marge géométrique

Si la fonction de décision produite est cohérente avec l'ensemble d'apprentissage, le risque d'erreur peut être borné en probabilité à partir de l'expression de la VC dimension (2.44) dans l'équation (2.27).

Si la fonction de décision produite n'est pas cohérente avec l'ensemble des exemples d'apprentissage, alors le risque d'erreur est borné par l'utilisation de la borne suivante (toujours avec une probabilité de $1-\delta$, c est une constante, voir le chapitre 4 de [CRISTI00] pour plus de détails) :

$$e_{reel} \leq \frac{c}{m}\left(\left(\frac{R}{\gamma'}\right)^2 \log^2 m + \frac{1}{\delta}\right) \tag{2.49}$$

On peut remarquer une certaine similarité avec l'équation (2.27). La quasi-marge γ' est déduite de la marge γ, les valeurs des variables ressorts $\boldsymbol{\xi}$ caractérisant l'importance de pénétration dans la marge des exemples considérés et du type de pénalité p appliqué ($p=1$ linéaire et $p=2$ quadratique) pour les exemples pénétrant dans cette marge. L'expression de γ' est la suivante :

$$(\gamma')^2 = \frac{R^2 + \|\boldsymbol{\xi}\|_p^2 \log\left(1/\gamma\right)^{2-p}}{R^2\gamma^2} \tag{2.50}$$

Plus le nombre d'exemples avec une variable ressort non nulle est grand, plus la quasi-marge γ' diminue. L'utilisation de la notion de marge souple permet donc de trouver une solution à un problème non linéairement séparable, mais l'augmentation artificielle de la marge géométrique γ, en diminuant l'importance de la pénalisation C, se traduit par une diminution de la *marge de confiance* γ'.

Dans [CRISTI00, CHAPEL04], d'autres bornes exploitant la notion de marge géométrique sont également proposées.

Bornes qui utilisent le nombre de vecteurs de support

Si la fonction de décision produite par un SVM utilise un nombre restreint n_{VS} de vecteurs de support pour être cohérente avec l'ensemble des données d'apprentissage, elle agit comme un schéma de compression efficace des données. Elle généralise correctement l'information car elle permet d'obtenir une représentation à la fois parcimonieuse et co-

hérente [LITTLE86]. La borne supérieure de l'erreur en généralisation suivante est alors vérifiée [CRISTI00]:

$$e_{reel} \leq \frac{1}{m - n_{VS}} \left(n_{VS} \log_2 \frac{em}{n_{VS}} + \frac{m}{\delta} \right) \tag{2.51}$$

avec une probabilité de $(1 - \delta)$. Il est important de noter que ce résultat est vrai pour toute fonction de décision produite n'exploitant qu'un nombre restreint d'exemples pour produire une fonction de décision cohérente.

Lorsque la fonction de décision produite n'est pas cohérente avec l'ensemble d'apprentissage, la borne précédente ne peut pas être utilisée, il existe par-contre une borne maximum de l'erreur réelle qui n'a pas cette contrainte[8] :

$$E\left(e_{reel}\right) \leq E\left(\frac{n_{VS}}{m-1}\right) \tag{2.52}$$

Pour estimer cette valeur de façon précise, il est nécessaire de réaliser un grand nombre d'apprentissages (ceci peut être réalisé avec différentes techniques de validation croisée). Dans le cadre de l'exploitation d'un estimateur basé sur la procédure de *leave-one-out (loo)*, il est possible de borner l'erreur loo à partir d'un seul apprentissage (*cf.* section 4.2.1.1):

$$e_{loo} \leq \frac{n_{VS}}{m} \tag{2.53}$$

2.4.5.7 Avantages et désavantages des SVM

L'avantage de la création d'une fonction de décision avec l'algorithme des SVM est que la solution produite correspond à l'optimum d'une fonction convexe. Elle ne possède donc pas plusieurs optima locaux comme pour les réseaux de neurones (dans leur formulation classique), mais un optimum global. Cet optimum correspond à la minimisation du risque structurel et donc à la recherche d'une hypothèse possédant de bonnes capacités de généralisation à partir d'un espace d'hypothèses donné. L'espace d'hypothèses dépend du choix de la fonction noyau, ce qui est à la fois un avantage et un inconvénient. Un avantage car l'algorithme d'apprentissage reste identique. Le problème est que la fonction noyau a souvent des paramètres libres, comme la largeur de bande avec un noyau gaussien, et que la recherche de la valeur optimale de ces paramètres libres ne correspond plus à la recherche du minimum d'une fonction convexe.

La recherche des valeurs idéales de ces paramètres, appelée communément *sélection de modèles*, est critique. Illustrons cet état critique avec l'utilisation d'un noyau gaussien et une largeur de bande σ très faible. La fonction de décision produite, pour ces conditions, sera cohérente avec les données et elle aura autant de vecteurs de support que d'exemples. L'hyperplan sélectionné, bien que de marge géométrique maximum, a une marge faible. Les bornes supérieures vues précédemment sont très élevées et elles ne garantissent plus rien sur le pouvoir de généralisation. Donc la sélection d'une mauvaise fonction noyau (ou de ces paramètres) peut conduire à produire un effet de sur-apprentissage et c'est le principal désavantage des SVM.

8. $E(\cdot)$ désigne ici l'espérance.

Un autre désavantage des SVM est la durée importante de la phase d'entraînement, en particulier si une sélection de modèles doit être réalisée, bien que dans les dix dernières années plusieurs optimisations de l'algorithme initial aient été proposées [PLATT99A, KEERTH01, FAN05].

Un dernier désavantage est la complexité de la fonction de décision produite lorsque la base d'apprentissage est de taille importante, même si, à la différence de la règle PPV, l'ensemble des exemples n'est pas utilisé grâce à la notion de vecteur de support.

2.5 Importance de la notion de *marge de confiance*

Les sections précédentes montrent que pour qu'un algorithme d'apprentissage soit performant il est nécessaire d'estimer la flexibilité de l'espace d'hypothèses et la notion de VC-dimension est un outil essentiel pour cette mesure. Dans le cas des SVM la notion de marge géométrique est importante car elle donne des informations sur la VC-dimension. La marge géométrique est un critère de confiance de la qualité de la fonction de décision produite.

La marge géométrique des SVM n'est pas le seul critère de confiance exploité pour caractériser le pouvoir de généralisation d'un algorithme d'apprentissage [SCHAPI97]. D'autres critères de confiance peuvent également guider un algorithme dans la construction ou la sélection d'une hypothèse [CRAMME02, GILAD-04].

2.5.1 AdaBoost et marge d'hypothèse

Les travaux de Schapire et al [SCHAPI97] ont montré que la notion de distance des exemples à la frontière de décision n'était pas le seul critère à prendre en compte pour mesurer la robustesse d'une hypothèse parmi un ensemble d'autres. AdaBoost est un algorithme qui a la particularité de combiner un ensemble de classificateurs *faibles* pour produire une décision robuste. Le terme *faible* est utilisé dans le sens où les classificateurs utilisés réalisent des prédictions légèrement supérieures au simple hasard. Dû à son rôle particulier de combiner des fonctions de décision produites par d'autres algorithmes d'apprentissage pour construire une fonction de décision finale plus performante, l'algorithme AdaBoost a un statut à part par rapport à tous les autres algorithmes d'apprentissage abordés précédemment.

Le synopsis d'AdaBoost est le suivant[9] :

1. Initialiser les poids w_i d'importance des exemples z_i ($t = 0$),
2. Réaliser un apprentissage avec un algorithme A en tenant compte de l'importance des exemples[10] pour produire une fonction de décision D_t,
3. Attribuer un poids α_t à la fonction D_t d'autant plus élevé que la fonction réalise peu d'erreurs sur les exemples z_i de poids w_i importants,

9. Pour les détails techniques sur la détermination exacte des poids, voir [SCHAPI97]

10. Une manière de procéder est de réaliser un tirage aléatoire sur la base d'exemples pour constituer la base d'entraînement. La probabilité d'appartenance d'un exemple z_i à la base d'entraînement est définie à partir des poids w_i.

4. Mettre à jour les poids w_i des exemples en diminuant le poids des bien classés et en augmentant ceux des autres,
5. Répéter les étapes 2 à 4 ($t = t + 1$), jusqu'à ce qu'un critère d'arrêt soit vérifié. ($t > t_{\max}$ par exemple).

La fonction de décision finale est :

$$D_T(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t D_t(x)\right) \tag{2.54}$$

avec $\forall x : D_t(x) \in [-1, +1]$.

Une autre particularité observée expérimentalement est qu'augmenter le nombre d'itérations de l'algorithme AdaBoost ne produit pas un phénomène de sur-apprentissage de la base comme on pourrait s'y attendre. En effet, une première borne sur l'erreur de généralisation établie est [SCHAPI97]:

$$R_{reel}(D_T) \leq R_{emp}(D_T) + O\left(\sqrt{\frac{T \cdot d_{\text{VC}}}{m}}\right) \tag{2.55}$$

L'interprétation de ce résultat est que la confiance sur l'estimation du risque réel diminue avec le nombre d'itérations. Au contraire, alors que l'algorithme AdaBoost produit une solution cohérente avec la base d'entraînement à une itération t_1, des résultats expérimentaux montrent qu'il continue à améliorer ses performances de généralisation (mesurée a partir d'une base de validation) pour les itérations ultérieures à t_1, tout en conservant la cohérence avec la base initiale. Pour expliquer ce phénomène, Shapire et al introduisent une autre notion de marge de *confiance* que la marge géométrique des SVM. Elle est basée sur le calcul du nombre de votes corrects et incorrects réalisés par l'ensemble des classificateurs pour chaque exemple de la base d'apprentissage. Ils définissent la notion de marge d'hypothèse γ_h pour un exemple de la base d'apprentissage par :

$$\gamma_h(x,y) = \frac{y \sum_{t=1}^{T} \alpha_t D_t(x)}{\sum_{t=1}^{T} \alpha_t} \tag{2.56}$$

Ce nombre est dans l'intervalle $[-1,1]$ et est positif seulement si D_T classifie correctement l'exemple. La marge γ_h peut être interprétée comme une mesure de confiance sur les prédictions. Notons :

$$c(\hat{\gamma}_h) = 1 - E(\gamma_h(x,y) < \hat{\gamma}_h) \tag{2.57}$$

la mesure de confiance relativement à une marge seuil de $\hat{\gamma}_h$ avec $E(\gamma_h(x,y) < \hat{\gamma}_h)$ l'espérance de $\gamma_h(x,y) < \hat{\gamma}_h$ estimée à partir de la base d'apprentissage. Shapire et al [SCHAPI97] ont prouvé que l'erreur en généralisation peut être bornée à partir de la notion de marge d'hypothèse, du seuil maximum sur cette marge et de la mesure de confiance.

$$R_{reel}(D_T) \leq 1 - c(\hat{\gamma}_h) + O\left(\sqrt{\frac{d_{\text{VC}}}{m \cdot \hat{\gamma}_h^2}}\right) \tag{2.58}$$

On note que dans cette formulation le nombre d'itérations n'intervient plus. Seules la valeur de seuil sur la marge et la mesure de confiance sur cette valeur sont utiles. Il a également été montré expérimentalement [SCHAPI97] que l'algorithme AdaBoost produit

une augmentation de la marge d'hypothèse γ_h des exemples difficiles à classer au cours de ses itérations (bien que la diminution de la borne (2.58) n'ait pas été prouvée). Notons

$$\hat{\gamma}_h^*(Z) = \min_{(x,y)\in Z} \gamma_h\,(x,y) \tag{2.59}$$

la marge minimum mesurée sur la base d'apprentissage. L'expression (2.58) devient :

$$R_{reel}\,(D_T) \leq O\left(\sqrt{\frac{d_{\mathrm{VC}}}{m\cdot\left(\hat{\gamma}_h^*\right)^2}}\right) \tag{2.60}$$

A partir de cette formulation, on comprend que l'algorithme AdaBoost puisse continuer à optimiser ses performances en généralisation alors que l'erreur empirique est nulle lorsque les itérations supplémentaires augmentent la valeur de (2.59).

2.5.2 Marge d'hypothèse à partir d'un ensemble de prototypes

La notion de plus proche voisin joue un rôle essentiel dans le domaine de l'apprentissage supervisé, comme nous l'avons vu en section 2.4.1. Soit l'hypothèse Z_k correspondant à k exemples d'une base d'exemples Z_m ($k < m$) choisis comme des prototypes représentatifs de la base initiale. L'idée est de définir une mesure qui correspond à une marge à partir de la règle PPV pour un exemple donné. Elle est définie de la façon suivante :

$$\gamma_h(x,y) = \frac{1}{2}\left(\|x_i - x\| - \|x_j - x\|\right) \tag{2.61}$$

avec x_j et x_i qui sont respectivement les exemples les plus proches de x de la même classe que y et d'une classe différente de y. Elle est utilisée dans la définition de l'algorithme LVQ [CRAMME02][11].

2.5.3 Marge géométrique et marge d'hypothèse

Revenons sur la notion de marge géométrique et de marge d'hypothèse pour en donner une interprétation générale et intuitive dans le cadre de la classification supervisée (voir figure 2.12).

La marge géométrique (ou *sample margin*) est une mesure de confiance qui est déterminée à partir de la distance minimale que doit parcourir un exemple pour qu'il traverse la frontière de décision propre à l'hypothèse, dans le cas des SVM cela correspond à la distance à l'hyperplan. Plus cette distance est grande et plus le risque de réaliser une erreur de prédiction est faible. A partir de l'estimation de la marge géométrique pour l'ensemble des exemples, il est possible de déterminer une mesure de confiance globale. Dans le cas des SVM, cette distance minimale à hyperplan qui est utilisée comme mesure de confiance globale sur l'ensemble de la base.

La marge d'hypothèse (ou *hypothesis margin*) mesure (à partir des exemples) de combien l'hypothèse doit être modifiée pour que les décisions induitent par l'hypothèse changent. Avec AdaBoost c'est une mesure qui dépend des fonctions de décision combinées et de l'importance accordée individuellement à chacune de ces fonctions.

11. Elle est également utilisée pour définir un algorithme qui permet de réaliser une sélection d'attributs [GILAD-04].

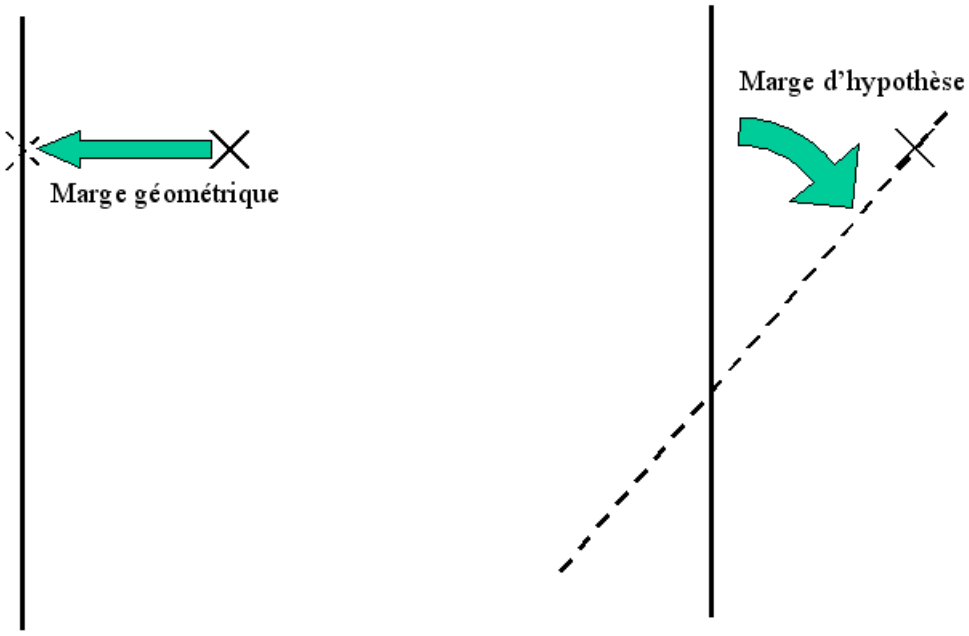


FIG. 2.12 – *Illustration des notions de marge géométrique de marge d'hypothèse*

L'essentiel est donc que l'on puisse contrôler la richesse des hypothèses à partir de mesures dans la phase de sélection (construction) d'une hypothèse (fonction de décision) sur un ensemble de données afin de garantir une forte probabilité pour que la fonction de décision produite ait des capacités de généralisation importantes. La validation de ces marges de confiance peut être réalisée par la définition de bornes sur l'erreur de généralisation et/ou des mesures expérimentales sur les capacités de généralisation. AdaBoost est l'exemple d'une première acceptation expérimentale qui a été ensuite validée par la démonstration de bornes sur l'erreur en généralisation. Les SVM ont suivi par contre le chemin inverse.

Ces notions de marge (ou critère) de confiance sont donc des outils indispensables pour définir des heuristiques puissantes lors de la recherche et de la construction d'hypothèses.

2.6 Estimation de l'erreur de généralisation

Il est important de pouvoir mesurer les performances en généralisation d'une fonction de décision produite par un algorithme d'apprentissage et de pouvoir comparer différentes fonctions de décision entre elles. Souvent les algorithmes d'apprentissage ont différents paramètres, variantes ou conditions d'initialisation. Comment sélectionner la bonne fonction de décision parmi un ensemble de fonctions de décision ? L'erreur empirique sur l'ensemble d'apprentissage n'est généralement pas un bon indicateur, car le risque plus ou moins important de sur-apprentissage existe. Pour éviter ce problème, plusieurs méthodes ont été définies. Elles sont basées sur le fait que les données utilisées pendant la phase d'apprentissage sont différentes de celles utilisées pour l'évaluation du résultat de cet apprentissage.

La base de données (Z) est divisée en trois bases différentes (Z^E, Z^T, Z^V avec $Z = Z^E \cup Z^T \cup Z^V$) lorsqu'une estimation précise des capacités de généralisation doit être réalisée. La première est la base d'entraînement (Z^E) et elle est directement utilisée pendant la phase d'entraînement de l'algorithme d'apprentissage supervisé pour la production de la fonction de décision (D). La suivante, la base de test (Z^T), est utilisée pour tester le pouvoir de généralisation de la fonction de décision produite et peut servir à sélectionner les bons paramètres (sélection de modèle) d'un algorithme d'apprentissage. La dernière, la base de validation[12] (Z^V), permet au final de valider la fonction de décision sélectionnée à partir d'un ensemble de données non exploité pendant le processus complet d'apprentissage (avec sélection de modèle). La réunion de la base d'entraînement Z^E et de test Z^T est généralement désignée comme la base d'apprentissage ($Z^A = Z^E \cup Z^T$). Si aucune sélection de modèle ne doit être réalisée, les bases de test et de validation sont généralement confondues. Nous notons D^{Z^E} la fonction de décision produite à partir de la base d'entrainement E et $e_V(D^E)$, le taux d'erreur de la fonction D^E estimée à partir de la base de validation V.

Si le nombre d'exemples présents dans Z^E et Z^T est insuffisant, la mesure $e_T(D^E)$ ou $e_V(D^E)$ n'est pas représentative des capacités de généralisation de l'algorithme d'apprentissage qui a produit cette fonction de décision. Pour éviter ce problème, plusieurs techniques statistiques ont été définies.

2.6.1 Validation croisée en k parties

La Validation Croisée (VC) en k parties est la procédure la plus utilisée dans le cadre de l'apprentissage supervisé. La base d'exemples Z est divisée en k sous-ensembles $Z = Z_1 \cup 1 \cdots \cup Z_k$ de tailles identiques (ou quasi-identiques, si le nombre d'exemples n'est pas multiple de k). Il y a k phases d'entraînement où les bases d'entraînement Z_i^E et les bases de test Z_i^T sont définies à partir des Z_i avec respectivement $Z_i^E = Z/Z_i$ et $Z_i^T = Z_i$.

L'estimation de l'erreur réelle est réalisée à partir de la moyenne des erreurs réalisées sur les k bases de test :

$$e_{\text{réelle}} \approx e_{k\text{VC}} = \frac{1}{k} \sum_{i=1}^{k} e_{Z_i^T}\left(D^{Z_i^E}\right) \tag{2.62}$$

En pratique la valeur de k choisie est généralement comprise entre 3 et 10 et $k = 5$ est un choix très courant. La limitation de k à des valeurs peu élevées est principalement motivée par la limitation des durées nécessaires à ces multiples phases d'entraînement. Au final, un apprentissage est réalisé sur la base compléte Z pour produire la fonction de décision dont l'erreur est estimée par la procédure de validation croisée.

12. Dans la littérature les notions relatives aux deux termes : *base de test* et de *base de validation*, sont parfois inversées par rapport à celles utilisées dans ce document. Nous avons adopté celles utilisées par Cornuéjols et al [CORNUÉ02] dans leur livre car elles nous ont semblé plus logiques si l'on considère que la sélection d'un modèle est réalisée en testant ses capacités à partir de Z^T et que la procédure complète d'apprentissage est validée à partir de Z^V.

2.6.2 Leave-one-out

La procédure de leave-one-out est un cas particulier de la procédure de validation croisée avec $k = m$. Il y a donc m phases d'entraînement : une pour chaque exemple de la base Z. Chaque base d'entraînement correspond à la base d'exemples initiale à laquelle un exemple a été enlevé $Z_i^E = Z/\{z_i\}$. La base de test ne contient qu'un exemple $Z_i^T = \{z_i\}$, celui enlevé à Z. L'équation 2.62 est utilisée pour déterminer l'erreur de *leave-one-out*, notée e_{loo}. Cette estimation de l'erreur réelle est considérée comme l'estimateur le moins biaisé. L'avantage de cette méthode est que chaque phase d'entraînement a un maximum d'exemples, mais le nombre important de phases d'apprentissage devient prohibitif dès que le nombre d'exemples est de l'ordre de quelques centaines.

2.6.3 Bootstrap

La procédure de bootstrap [EFRON82], technique statistique adaptée au problème de l'estimation du risque réel dans le cadre de l'apprentissage supervisé, réalise un ensemble de plusieurs apprentissages à partir de différents ensembles d'entraînement produits à partir des données initiales. Elle diffère cependant des deux méthodes précédentes, car la constitution d'une base d'entraînement est réalisée à partir d'un tirage aléatoire avec remise des m exemples de la base Z. L'idée est d'approcher par simulation la distribution du risque réel avec un ensemble d'entraînement de taille m, lorsque l'on sait que sa variance peut être importante. Pour que la simulation soit efficace, il est nécessaire que le nombre d'entraînements soit important (plusieurs centaines). L'estimateur de boostrap est particulièrement adapté aux bases d'apprentissage de taille réduite. Nous décrivons par la suite trois procédures pour estimer l'espérance moyenne de l'erreur. Supposons que b bases d'entraînements soit générées et notons Z_i^E la base d'entraînement numéro i ($1 \leq i \leq b$) avec m exemples sélectionnés par un tirage aléatoire avec remise et B_j l'ensemble des indices i des bases d'entraînement Z_i^E qui ne contiennent pas l'exemple j. Le premier estimateur nommé naïf (ou *plug-in*) estime l'erreur empirique directement à partir des bases d'entraînement Z_i^E. Il est donc sensible au problème de sur-apprentissage, d'où le nom d'estimateur naïf :

$$e_{\text{naif}} = \frac{1}{b} \sum_{i=1}^{b} e_{Z_i^E} \left(D^{Z_i^E} \right) \tag{2.63}$$

Le deuxième estimateur *out-of-bag* (oob) est produit à partir du comportement des fonctions de décision pour les exemples non utilisés par chaque apprentissage (*i.e.*, les B_j) :

$$e_{\text{oob}} = \frac{1}{m} \sum_{j=1}^{m} \frac{1}{|B_j|} \sum_{i \in B_j} I \left(D^{Z_i^E}(x_j) = y_j \right) \tag{2.64}$$

avec $I(v)$ une fonction qui vaut 1 si v est vrai et 0 sinon. La probabilité qu'un exemple soit sélectionné dans une base d'entraînement de m exemples est de $1 - (1 - 1/m)^m \approx 1 - 1/e \approx 0{,}632$. Le nombre d'exemples non vus lors d'un entraînement est donc en moyenne de 37%. Bien que moins optimiste que l'estimateur précédent, un biais est introduit par la réduction de la taille de la base de test. Efron et Tibshirani ont proposé un nouvel

estimateur [EFRON97] à partir des deux précédents pour réduire l'importance des défauts qui concernent ces deux estimateurs. Le premier étant optimiste et le second pessimiste, la somme pondérée suivante, nommée .632-bootstrap, est un judicieux compromis :

$$e_{\text{réelle}} \approx e_{.632} = 0{,}368 \times e_{\text{naif}} + 0{,}632 \times e_{\text{oob}} \tag{2.65}$$

2.7 Conclusion

Dans ce premier chapitre nous nous sommes intéressé aux objectifs principaux de l'apprentissage et de la définition d'algorithmes capables de réaliser ces apprentissages. Nous avons rappelé que :

- l'apprentissage doit être réalisé à partir d'une représentation imparfaite du monde représenté sous la forme de bases de données numériques,
- il est nécessaire de choisir un principe inductif pour sélectionner, dans un espace d'hypothèses également à définir, l'hypothèse qui généralise au mieux ces données,
- la sélection de la meilleure hypothèse est problématique à cause de la difficulté de choisir un bon compromis biais-variance,
- le contrôle de la richesse de l'espace d'hypothèses est indispensable pour éviter une augmentation incontrôlée de la variance.

Nous avons ensuite rappelé les résultats principaux de la théorie de l'apprentissage de Vapnik et en particulier les rôles essentiels de la notion de VC-dimension et du principe de minimisation du risque structurel. Plusieurs algorithmes d'apprentissage artificiel sont décrits en s'attardant plus particulièrement sur les SVM et sur ceux exploitant la règle *k-ppv*. Nous insistons entre autres sur la façon de réduire la complexité des processus décisionnels, créés par ces algorithmes, pour contrôler la richesse de l'espace d'hypothèses sous-jacent, en particulier dans la définition de principes d'inférence basés sur la minimisation du risque structurel. Nous insistons également sur l'importance de l'utilisation de critères de confiance, autres que le taux d'erreur, pour aider la création de processus décisionnels performants. Pour finir, nous rappelons les différentes techniques de validation croisée pour réaliser une estimation de l'erreur de généralisation.

Ce chapitre a permis de mettre en lumière les problématiques liées à la définition et à l'exploitation d'algorithmes d'apprentissage. Les concepts et notations introduits seront utilisés dans les chapitres 4 à 6.

CHAPITRE 3

OPTIMISATION AVEC MÉTA-HEURISTIQUES

Sommaire

Dans le chapitre 2, nous avons vu que les problèmes d'apprentissage correspondent à la recherche d'une fonction de décision h^* de qualité optimale, suivant un critère q, l'espace des fonctions de décision admissibles étant défini par un ensemble de paramètres ou variables $\boldsymbol{\theta}$ libres, généralement contraint dans un ensemble de valeurs possibles Θ. La recherche de la fonction de décision optimale correspond donc à un problème d'optimisation[1]:

$$\theta^* = \arg\max_{\theta \in \Theta} q\left(h_\theta\right) \tag{3.1}$$

Lorsque l'on ne se limite pas au cadre de l'apprentissage, une formulation générique d'un problème d'optimisation est :

$$\theta^* = \arg\max_{\theta \in \Theta} f(\theta) \tag{3.2}$$

avec θ^* la solution de l'espace des solutions Θ qui optimise la fonction objectif f ($f \equiv q\left(h_\theta\right)$) qui représente la qualité d'une solution θ pour un problème donné. Dans la suite de ce chapitre, tous les problèmes seront considérés comme des problèmes de maximisation, sachant que la transformation d'un problème de minimisation en un problème de maximisation est triviale et peut être simplement réalisée en changeant le signe de la fonction f à optimiser.

Les méthodes permettant de réaliser cette optimisation seront très variables suivant la nature du problème. Par exemple dans le cas des *k-ppv*, il suffit d'évaluer le taux d'erreur pour un ensemble réduit de valeurs de k ($k \in \{1, \cdots, k_{\max}\}$) et de choisir la meilleure. Dans le cas des SVM, lorsque la fonction noyau (ainsi que son ou ses paramètres) et la constante de régularisation C sont fixées, la recherche de la valeur des $\boldsymbol{\alpha}$ correspond à l'optimisation d'une fonction convexe quadratique avec des contraintes linéaires. C'est un problème mathématique bien défini (*cf.* chapitre 4) et il existe des méthodes classiques

1. Si les variables du modèle $\boldsymbol{\theta}$ doivent respecter des contraintes explicites, pour garder une formulation la plus générale possible, elles sont implicitement représentées par les limitations intrinsèques de l'espace Θ.

pour le résoudre (bien que des algorithmes spécifiques au cas des SVM aient été développés pour tenir compte de problèmes d'espace mémoire et de temps de calcul). Dans le cas plus général où seule la fonction noyau a été fixée (un noyau gaussien par exemple), mais pas les paramètres du noyau (σ pour un noyau gaussien) et la constante de régularisation C, le choix des valeurs optimales de σ et C correspond à la recherche d'un modèle θ optimal ($\theta \equiv (C,\sigma)$). Le choix de $\boldsymbol{\alpha}^*$ étant considéré comme implicite à un modèle $\boldsymbol{\theta}$ particulier, même si le calcul effectif de $\boldsymbol{\alpha}^*$ n'est pas trivial. Le problème de la recherche du modèle $\boldsymbol{\theta}$ correspond à un problème qui n'est plus convexe (*cf.* chapitre 4), il y a donc en général plusieurs minima locaux. Parmi les méthodes proposées pour résoudre ce type de problème, un sous-ensemble d'entre elles utilise des méthodes désignées sous le terme de méta-heuristiques.

Les méthodes méta-heuristiques ont été fortement développées et utilisées dans les années 1980 pour résoudre initialement des problèmes d'optimisation difficiles issus de la recherche opérationnelle. Un exemple d'application industrielle est le placement automatique des composants électroniques afin que la minimisation de la longueur des pistes dans l'implantation des circuits intégrés soit proche de l'optimal. Ce problème a comme illustre parent celui du voyageur de commerce. C'est un problème combinatoire pour lequel les méta-heuristiques sont connues pour produire des solutions très efficaces même lorsqu'il est de grande taille. Les méta-heuristiques ont été ensuite appliquées à de nombreux domaines dont l'apprentissage artificiel. La problématique qui correspond à la sélection d'un sous-ensemble réduit d'attributs pertinents ou d'exemples pertinents est une des applications majeures des méta-heurisques dans le cadre de l'apprentissage artificiel [KUNCHE99]. La liste des applications industrielles actuelles dans lesquelles elles sont mises en œuvre est importante. Le lecteur intéressé peut trouver de nombreux exemples dans [DRÉO03]. Pour le cas de l'apprentissage artificiel, le chapitre 8 de [CORNUÉ02] explore la problématique de l'apprentissage à travers l'utilisation d'algorithmes évolutionnaires. Tout problème d'apprentissage confronté à des problèmes d'explosion combinatoire ou d'espace de recherche avec des minima locaux peut tirer profit des méthodes méta-heuristiques.

L'avantage des méthodes méta-heuristiques est qu'elles sont suffisamment génériques pour permettre d'optimiser une large gamme de problèmes différents, sans nécessiter de changements profonds dans le principe algorithmique employé. Pour un problème particulier, l'adéquation d'un algorithme de type méta-heuristique consiste à définir des briques élémentaires adaptées à ce problème. Ces briques élémentaires sont alors directement utilisées par l'ossature générale de la méthode méta-heuristique. Un des avantages de ces méthodes est que lorsque deux problèmes sont de nature similaire, l'effort d'adaptation est réduit pour passer de l'optimisation de l'un à l'autre, car certaines briques de base peuvent être réutilisées ou nécessiter des modifications mineures.

Dans ce chapitre, nous nous proposons de rappeler les fondements essentiels des méthodes méta-heuristiques en insistant sur les méthodes désignées sous les termes de recherche avec tabous et d'algorithmes évolutionnaires. Les autres catégories de méthodes méta-heuristiques connues que sont le recuit simulé et les fourmis artificielles ne sont que brièvement évoquées ici, bien que d'un point de vue général elles soient également importantes.

3.1 Recherche avec tabous

La recherche avec tabous est une méthode méta-heuristique pour l'optimisation difficile qui a été développée dans les années 80 par Fred Glover 80 [GLOVER89A, GLOVER89B]. Elle s'est révélée particulièrement efficace et a été appliquée avec succès à de nombreux problèmes [GLOVER97, HAO99]. Elle est basée sur une recherche itérative qui choisit dans un voisinage restreint la meilleure solution même si elle est plus mauvaise que celle de l'itération précédente. Une mémoire à court terme est utilisée pour éviter tout cycle visitant périodiquement le même optimum local. A partir des derniers mouvements mémorisés, un ensemble de solutions est considéré comme tabou et un mouvement n'est réalisable que vers l'une des solutions voisines qui n'est pas taboue.
Le choix des solutions taboues peut être tout simplement les dernières solutions visitées, mais en général le critère qui détermine les solutions taboues à un moment précis de la recherche dépend du problème à optimiser et de la représentation d'une solution. Lorsque des caractéristiques de modification définissent l'appartenance d'une solution à l'ensemble tabou, l'ensemble des solutions interdites à chaque itération peut contenir des solutions meilleures que toutes celles déjà visitées. Un mécanisme particulier, appelé l'aspiration, permet de pallier l'effet de ne pas pouvoir les sélectionner et de lever ainsi leur statut de tabou.

Des mécanismes nommés diversification et intensification permettent de doter cette méthode de comportements correspondant à des effets caractéristiques d'une mémoire à long terme, la recherche taboue oscillant entre ces deux états pendant son processus de recherche. L'objectif est de permettre à la fois d'atteindre des régions prometteuses par un parcours rapide de l'ensemble de l'espace de recherche et de pouvoir localiser le plus précisément possible une solution proche de l'optimal par un parcours agressif de la région prometteuse la contenant. L'ensemble de ces propriétés fait que le méthode de recherche avec tabous peut être adaptée à une grande classe de problèmes d'optimisation difficiles.

Les démonstrations de convergence pour la recherche avec tabous existent, mais supposent des conditions strictes, rarement présentes en pratique [FAIGLE37]. Glover c'est inspiré de l'utilisation que les humains font de leur mémoire lorsqu'ils cherchent par *tâtonnement* pour améliorer une solution à un problème. De l'avis même de l'auteur [GLOVER97] de cette méthode, la modélisation de la mémoire introduit de multiples degrés de liberté qui s'opposent à toute analyse mathématique rigoureuse de la recherche avec tabous. Cependant ce point de vue de Glover n'a pas découragé certains chercheurs à continuer une étude théorique de cette méthode [PUANGD04], dont l'auteur de cette remarque [GLOVER02].

3.1.1 Synopsis de la méthode

La méthode a pour objectif de produire une solution la plus proche possible de celle correspondant à l'optimal du problème (3.2). Un algorithme de type recherche avec tabous commence avec une solution initiale $\boldsymbol{\theta}_0 \in \Theta$ à l'itération $it = 0$. La solution $\boldsymbol{\theta}_0$ est soit choisie au hasard parmi celles de Θ, soit déterminée à partir d'informations extérieures caractérisant sa qualité comme solution de départ. La recherche est axée sur une exploration itérative de l'ensemble des solutions en utilisant la notion de voisinage et elle garde trace, à chaque itération, de la meilleure solution $\boldsymbol{\theta}^*$ découverte dans le passé. Formelle-

ment, on définit pour toute solution $\boldsymbol{\theta}$ un ensemble $V(\boldsymbol{\theta}) \in \Theta$ qui correspond à l'ensemble des solutions voisines de $\boldsymbol{\theta}$. Pour certains problèmes d'optimisation, lorsque les variables correspondant au modèle $\boldsymbol{\theta}$ sont continues, il est commun de quantifier l'espace correspondant à l'ensemble des solutions possibles. Dans ce cas, le voisinage d'une solution $V(\boldsymbol{\theta})$ correspond à un ensemble de solutions énumérables. Ceci n'est envisageable que si la valeur f des solutions non atteignables de l'espace initial, due à la quantification, est très proche de la valeur des solutions atteignables dans le voisinage immédiat. L'objectif est de choisir le meilleur mouvement possible dans le voisinage en tenant compte des informations déductibles des mouvements et solutions visitées dans le passé, ceci afin de pouvoir se diriger de manière efficace vers la solution optimale. L'information minimale utilisée par une recherche avec tabous à partir des mouvements passés est le risque de formation d'un cycle et c'est la notion de solution taboue à travers l'utilisation d'une mémoire à court terme qui permet d'éviter la formation de cycles. Formellement, un ensemble de solutions taboues, noté $\bar{\Theta}_{it}$, est défini pour chaque itération. Une solution est éligible pour la prochaine itération si elle est dans le voisinage de la solution initiale et ne fait pas partie des solutions taboues à l'itération considérée. Il existe une exception qui consiste à accepter une solution taboue comme valide si elle permet d'atteindre une solution de qualité meilleure que la plus performante rencontrée ($\boldsymbol{\theta}^*$). Ce mécanisme est nommé *aspiration*. Sans connaissance particulière sur le problème d'optimisation, la règle minimale de l'aspiration est d'accepter de lever le statut tabou d'une solution $\boldsymbol{\theta}$ si elle produit une solution meilleure que la meilleure déjà rencontrée. Formellement, les solutions valides dans un voisinage sont définies de la manière suivante :

$$\tilde{V}(\boldsymbol{\theta}_{it}) = \{\boldsymbol{\theta} | \boldsymbol{\theta} \in V(\boldsymbol{\theta}_{it}) \wedge (\boldsymbol{\theta} \notin \bar{\Theta}_{it} \vee f(\boldsymbol{\theta}) > f(\boldsymbol{\theta}^*_{it}))\} \tag{3.3}$$

Les solutions taboues pour la prochaine itération dépendent des solutions taboues actuelles, du mouvement choisi représenté par le couple $(\boldsymbol{\theta}_{it},\boldsymbol{\theta}_{it+1})$, ainsi que la meilleure solution actuelle $\boldsymbol{\theta}^*_{it}$. En toute généralité, il est nécessaire de tenir compte de la possibilité d'élaguer l'espace de recherche à partir de la connaissance d'une bonne solution. Comme l'ensemble des solutions taboues correspond à une mémoire de court terme, il est nécessaire de rappeler systématiquement quelle est la meilleure solution déjà rencontrée si on ne veut pas perdre trace de cette possibilité d'élagage. Formellement, la fonction de mise à jour des solutions taboues est notée g. Cette fonction est trop fortement dépendante du problème d'optimisation à traiter pour en donner une description plus précise.

Le Synopsis de la recherche avec tabous est le suivant:

1. choisir une solution initiale $\boldsymbol{\theta}_0 \in \Theta$ ($\boldsymbol{\theta}^* = \boldsymbol{\theta}_0$, $\bar{\Theta} = \emptyset$, $it = 0$),
2. sélectionner comme prochaine solution $\boldsymbol{\theta}_{it+1}$ la meilleur solution non taboue (*i.e.* $\boldsymbol{\theta}_{it+1} = \underset{\boldsymbol{\theta} \in \tilde{V}(\boldsymbol{\theta}_{it})}{\arg\max}\, f(\boldsymbol{\theta})$) dans le voisinage de $\boldsymbol{\theta}_{it}$,
3. mettre à jour la meilleure solution actuelle $\boldsymbol{\theta}^*$ si $f(\boldsymbol{\theta}_{it+1}) > f(\boldsymbol{\theta}^*)$,
4. mettre à jour la liste des solutions taboues: $\bar{\Theta}_{it+1} = g(\bar{\Theta}_{it},(\boldsymbol{\theta}_{it},\boldsymbol{\theta}_{it+1}),\boldsymbol{\theta}^*)$,
5. répéter les étapes 2 à 5 ($it = it + 1$) jusqu'à ce que le critère d'arrêt soit vrai,
6. retourner $\boldsymbol{\theta}^*$.

Le choix d'un bon critère d'arrêt est également fortement dépendant du problème à optimiser.

3.1.1.1 Représentation d'une solution

Une solution $\boldsymbol{\theta}$ est généralement représentée par un vecteur de n variables $(\theta_1, \cdots, \theta_n)$, où chaque variable a un domaine de définition $\mathcal{D}(\theta_i)$ généralement représenté par deux valeurs $\theta_i^{\min}$ et $\theta_i^{\max}$ qui correspondent aux deux valeurs extrêmes possibles que peut prendre la variable θ_i. Ces variables peuvent être de type réel, entier ou booléen. Un vecteur $\boldsymbol{\theta}$ peut contenir des variables toutes d'un même type ou être composé de variables de types hétérogènes.

3.1.1.2 Mouvements et voisinage

La notion de voisinage est définie à partir de celle de mouvement possible. Le voisinage d'un point (solution) correspond à l'ensemble des solutions que l'on peut atteindre en réalisant à partir de ce point un mouvement autorisé. Formellement, nous représenterons le statut *possible* d'un mouvement par une fonction mv à deux arguments retournant une valeur booléenne. $mv(\boldsymbol{\theta}_o,\boldsymbol{\theta}_d)$ désignera un mouvement d'un point d'origine $\boldsymbol{\theta}_o$ vers un point de destination $\boldsymbol{\theta}_d$ de l'espace Θ. Le voisinage d'un point $\boldsymbol{\theta}_o$ correspond à une définition en extension de l'ensemble des points atteignables par un mouvement autorisé :

$$V(\boldsymbol{\theta}_o) \equiv \{\boldsymbol{\theta} | \boldsymbol{\theta} \in \Theta, mv(\boldsymbol{\theta}_o,\boldsymbol{\theta})\} \tag{3.4}$$

La définition d'un mouvement peut être de nature multiple. Pour cette définition, il est important de vérifier qu'il est toujours possible de relier deux points quelconques de l'espace des solutions à partir d'un ensemble énumérable de mouvements (en ignorant dans ce cas la notion de solution taboue). Prenons le cas où l'espace des solutions autorisées correspond au produit cartésien de l'ensemble des domaines de définition des variables impliquées dans la représentation vectorielle d'une solution. Un mouvement autorisé peut correspondre à ajouter une quantité $\pm\delta$ à une de ces variables. Ce qui peut être formalisé par:

$$mv_\delta(\boldsymbol{\theta},\acute{\boldsymbol{\theta}}) \equiv \boldsymbol{\theta},\acute{\boldsymbol{\theta}} \in \Theta, \exists i : (|\acute{\theta}_i - \theta_i| = \delta, \forall j \neq i : \acute{\theta}_j = \theta_j) \tag{3.5}$$

Dans ce cas, il est évident qu'il existe toujours un chemin (ensemble de mouvements) permettant de relier deux points de l'espace Θ.

Le choix de la valeur de δ lorsque les variables sont de type réel est un problème essentiel et correspond à une problématique de même nature que celle concernant le choix du pas pour les méthodes de type descente de gradient. La valeur de δ dans ce cas est considérée comme un paramètre à règler pendant les itérations de la méthode (voir par exemple [CAWLEY01]). Lorsque les valeurs des variables sont de type entier, la valeur de δ est généralement 1 (tout du moins dans la phase d'intensification). Lorsque les variables sont de type booléen, on considère qu'elles sont codées 0 ou 1 (δ =1 dans ce cas). Il est souvent utile de pouvoir faire référence à la nature de la modification qui permet de passer d'une solution $\boldsymbol{\theta}$ à une solution $\acute{\boldsymbol{\theta}}$, qualifiant ainsi la nature du mouvement réalisé. Nous noterons par Δ la représentation de la modification tel que : $\acute{\boldsymbol{\theta}} = \boldsymbol{\theta} + \Delta$. Pour la définition (3.5), le couple (i,s) avec $s \in \{+,-\}$ peut être une représentation compacte et facilement manipulable de Δ si la valeur du pas δ est fixe, dans le cas contraire le triplet (i,s,δ) est une représentation envisageable.

La précédente définition du mouvement n'est qu'un cas parmi tant d'autres[2]. Elle dépend de la nature de l'espace de recherche qui est directement lié à la nature du problème à optimiser. Les a priori sur un problème permettent généralement de choisir une définition appropriée au problème et cette définition peut permettre un convergence plus rapide vers une solution proche de l'optimum [DRÉO03].

Le choix d'une définition des *mouvements possibles* par rapport à une autre peut être motivé par le fait que la variation de la fonction objectif f est directement calculable à partir du mouvement réalisé : $f(\acute{\boldsymbol{\theta}}) - f(\boldsymbol{\theta}) = \tilde{f}(\delta)$. Dans le cas où $\tilde{f}(\delta)$ est beaucoup plus rapide à calculer que $f(\acute{\boldsymbol{\theta}})$, il est judicieux de choisir ce type de mouvement. Si en plus, le mouvement est sémantiquement significatif pour le problème d'optimisation à résoudre, il est encore plus judicieux de le choisir. Le nombre de mouvements à réaliser pour atteindre deux solutions éloignées peut également intervenir dans le choix de la définition de la notion de mouvement.

Pour obtenir plus d'informations sur les différentes formulations envisagables pour la définition de voisinage et de mouvement, se reporter aux livres traitant de ce sujet [GLOVER97, DRÉO03].

3.1.1.3 Mémoire à court terme

Nous avons évoqué précédemment que la mémoire à court terme a pour premier rôle d'éviter la formation de cycle. Revenons sur cette notion importante. Le fait de choisir le meilleur mouvement dans un voisinage restreint, et ceci même si cela conduit à choisir une solution moins performante, est un mécanisme essentiel pour pouvoir quitter un optimum local, mais il n'est pas suffisant. En effet, le prochain mouvement peut conduire à la précédente solution, si aucune mémoire ne permet de se rappeler que celle-ci vient juste d'être explorée. Une mémoire à court terme permet de restreindre les solutions à visiter à partir de des expériences passées en désignant certaines solutions comme taboues. Si formellement il est facile de donner une formulation générale de la notion de solutions taboues, pratiquement, l'implémentation réelle de g est fortement dépendante de la nature du problème à considérer. Malgré tout, il est possible de dégager des principes généraux concernant la mémoire à court terme qui sont:

- le type de représentation manipulée pour la détermination de l'état tabou,
- la durée de la mémoire à court terme et donc du statut tabou d'une solution (ou mouvement),
- la structure de données utilisée pour déterminer rapidement les solutions (ou mouvements) interdites.

La mémoire à court terme peut exploiter deux représentations : les solutions et les mouvements. Par exemple pour éviter de former un cycle, la première possibilité est de mémoriser les solutions déjà explorées dans une liste *taboue* de longueur donnée. La longueur de la liste représentant l'effet de mémoire à court terme. Une autre solution est de mémoriser le mouvement inverse à celui que l'on vient de réaliser comme tabou, évitant ainsi de revenir en arrière lors de la traversée d'une vallée, avec le risque d'interdire de visiter une solution qui n'a jamais encore été explorée. Dans l'exemple de la représentation (3.5) d'un mouvement, cela consiste à considérer que si la valeur d'une variable

2. Elle nous sera par contre utile dans le chapitre 5

vient d'être augmentée, ce choix est assumé; il est donc interdit de la diminuer pendant un certain nombre d'itérations (effet de la mémoire à court terme). Les deux représentations peuvent être combinées. Dans ce cas cela correspond à autoriser de repasser par une solution s'il existe encore des chemins le traversant qui n'ont pas été empruntés. Ces mécanismes ne sont pas suffisants pour empêcher de rester bloqué dans un voisinage restreint d'un optimum local, en particulier si le nombre de solutions différentes dans un voisinage proche d'une solution (*i.e.* atteignables en quelques mouvements) est important. La mémoire locale peut également servir à contraindre la direction d'exploration en rendant tabous des mouvements (ou solutions) correspondant à un changement trop important de direction de recherche. Un ensemble de mouvements successifs est alors utilisé pour définir cette notion de direction. Cet effet a également la possibilité d'empêcher la formation de cycles courts.

La durée d'une interdiction, pour une solution ou un mouvement, est un paramètre important. Trop faible, la procédure de recherche est quasi-amnésique et peut facilement produire des cycles. Trop importante, elle peut conduire à restreindre la recherche dans une zone limitée de l'espace de recherche, lui interdisant par la même l'occasion d'en ressortir suffisamment rapidement. Plusieurs expérimentations [GLOVER97, DRÉO03] montrent que ce n'est pas un paramètre qui nécessite un réglage fin, mais seulement un dimensionnement correct par rapport à celui du problème, ainsi que la prise en compte de la stratégie utilisée pour la définition des états tabous. Une variante permet de rendre l'effet de la mémoire à court terme plus performant pour un grand nombre de problème. Elle consiste à ne pas choisir une durée fixe d'interdiction mais à choisir le nombre d'itérations d'interdiction aléatoirement entre un seuil bas et un seuil haut [TAILLA95] pour chaque nouvelle solution ou mouvement tabou.

L'efficacité de la recherche et les capacités nécessaires pour le stockage des informations concernant la détermination d'un état tabou, nécessitent généralement la définition de structures de données appropriées. De plus, la mise à jour de ces données pour éliminer les interdictions dont la durée est dépassée doit aussi être efficace. L'utilisation de structures de données basées sur des ensembles de tables de hachages organisées de manière hiérarchique correspond généralement à un cadre approprié pour la réalisation de mémoires à court terme (voir par exemple [WOODRU93]).

3.1.1.4 Aspiration

L'utilisation d'une mémoire à court terme pour définir quelles solutions sont taboues peut être plus ou moins agressive. La possibilité d'interdire l'accès à une solution voisine jamais visitée (ou conduisant à une solution jamais visitée) peut avoir pour effet qu'une solution optimale relativement à toutes celles déjà visitées sera forcement ignorée. Pour réduire l'importance de ce phénomène, un mécanisme nommé aspiration permet d'en réduire les conséquences. L'idée est d'évaluer (plus ou moins précisément) le potentiel de toute solution voisine pour déterminer si l'une d'entre elles est objectivement meilleure (ou conduisant à une solution meilleure), et la sélectionner malgré son statut de tabou. On dit alors que le processus de recherche a été aspiré par la solution. Les critères d'aspiration autres que la sélection d'une solution meilleure que toutes celles visitées (nous le nommons critère simple d'aspiration) nécessitent forcément une connaissance précise pour pouvoir anticiper qu'une solution conduira le processus de recherche vers une so-

lution meilleure (ou tout du moins avec une forte probabilité) pour accepter d'annuler l'interdiction la concernant. On peut trouver de nombreux critères d'aspirations dédiés à des classes particulières de problèmes dans [GLOVER97].

3.1.2 Intensification et Diversification

Bien qu'en général la recherche avec tabous atteigne rapidement de bonnes solutions, elle peut rencontrer des difficultés à atteindre les meilleures solutions malgré une bonne exploitation de la mémoire à court terme. Dans le cas d'un espace de recherche de grande dimension, le nombre d'itérations pour atteindre les meilleures solutions peut rapidement devenir prohibitif. Cet effet est accentué lorsque le coût d'estimation de la fonction objectif pour une solution donnée reste important et qu'aucune estimation rapide à partir du mouvement l'ayant produit n'est facile à définir. Un autre effet à prendre en compte est que la mémoire à court terme, par définition, ne permet pas de prendre en compte le comportement de la recherche sur un grand nombre d'itérations, ce qui la rend myope sur le long terme. Indépendamment du nombre d'itérations laissées à la phase de recherche avec tabous, elle peut ne plus pouvoir faire progresser les solutions construites au-delà d'un certain seuil, faute de vision globale sur sa stratégie de recherche. La réalisation des mécanismes désignés par les termes de diversification et d'intensification permettent généralement d'améliorer les performances de la recherche avec tabous dans de tels cas. Globalement cela correspond à ajouter une mémoire à long terme qui va changer le comportement de la méthode au cours du temps. Cette mémoire permettra, entre autres, d'alterner entre des phases d'intensification et de diversification pendant l'exploration de l'espace de recherche. L'intensification a pour objectif de réaliser une recherche exhaustive dans les régions prometteuses et faiblement explorées. Cela peut également correspondre à une spécialisation de la méthode de recherche, soit en se concentrant sur un certain type de mouvement, soit en se limitant à la modification d'un ensemble réduit de variables de la solution. La diversification a pour objectif des modifications significatives de la solution pour permettre de quitter rapidement une région peu efficace ou déjà fortement explorée par une précédente phase d'intensification et de découvrir ainsi une nouvelle région prometteuse. La diversification peut également concerner un changement de stratégie de recherche ou simplement obliger à choisir des actions peu utilisées dernièrement. La mémoire à long terme enregistre alors des informations statistiques qui permettent de résumer l'effet de sa propre recherche sur un nombre important d'itérations. Ces statistiques servent au pilotage des phases de transition entre des stratégies d'intensification et de diversification.

3.2 Algorithmes évolutionnaires

Le biologiste britannique Charles Darwin a mis en avant le mécanisme de la sélection naturelle pour expliquer comment des espèces nouvelles pouvaient être issues d'autres espèces. Dans son ouvrage majeur, *De l'origine des espèces par voie de sélection naturelle* [DARWIN59], Darwin indique que les espèces vivantes subissent des variations aléatoires et que seules celles qui se révèlent favorables à la survie dans leur environnement particulier sont conservées et transmises, celles qui se révèlent défavorables étant éliminées. C'est ainsi qu'opère la sélection naturelle, qui est, en quelque sorte, la sanc-

tion de l'environnement sur l'aptitude des êtres à survivre et à se reproduire. La sélection naturelle peut alors être vue comme la résolution d'un problème d'optimisation où la solution (l'espèce) recherchée doit être optimale ou plus exactement proche de l'optimale par rapport au problème (adéquation avec le monde environnant). La génétique a par la suite permis de mieux comprendre ce qui se cachait derrière le terme *aléatoire* utilisé par Darwin, en particulier, l'importance du matériel génétique intrinsèquement lié à chaque espèce et qui est transmis au moment de la reproduction. Le caractère aléatoire peut alors être décomposé en plusieurs briques qui sont :

- la taille de la population, ainsi que sa répartition géographique,
- la possibilité de se reproduire qui est plus ou moins importante suivant que l'individu est plus ou moins adapté,
- le croisement entre les gènes des deux individus lors de la reproduction, qui va définir les caractéristiques génétiques de l'enfant et son adéquation avec le monde dans lequel il devra vivre.
- les mutations du matériel génétique. Ces mutations sont rares, mais peuvent être la cause, sur une autre échelle de temps, de la naissance de nouvelles espèces,
- la durée de vie plus ou moins longue selon les individus.

Ceci laisse entrevoir que, dans le monde du vivant, il existe d'autres voies pour l'établissement de processus stochastiques performants qui permettent d'optimiser des structures complexes sans que les caractéristiques du problème soient codées de manière explicite.
Des chercheurs ont alors eu l'idée de s'inspirer des mécanismes de sélection naturelle dont nous sommes issus, pour développer une classe d'algorithmes dits évolutionnaires, afin de résoudre des problèmes d'optimisation dits difficiles. Dans les années 70, les premiers travaux sur l'évolution artificielle ont concerné les algorithmes génétiques, les stratégies d'évolution, la programmation évolutive et la programmation génétique. Les *algorithmes génétiques* (*Genetic Algorithms*, GA) ont été présentés en 1975 par Holland [HOLLAN75]. Les fondations de la *programmation évolutionnaire* (*Evolutionary Programming*: EP) sont plus anciennes (années 1960) et ont commencé avec les travaux de Fogel et al [FOGEL66]. Parallèlement, les stratégies d'évolution (*Evolution Strategies*: ES)) ont été initiées par les travaux de Rechenberg et al [RECHEN65]. La programmation génétique (*Genetic Programming*: GP) est exprimée formellement par Koza [KOZA92]. Elle permet la programmation automatique d'ordinateurs par application des mêmes principes que les algorithmes génétiques. Ces quatres types d'algorithmes ont utilisé des principes globalement communs qui sont inspirés de la théorie de l'évolution de Darwin [DARWIN59]: utilisation d'une population d'individus, évaluation des individus par une fonction, sélection des meilleurs et génération d'une nouvelle population avec des opérateurs de croisement et de mutation.

Dans la section suivante nous nous intéresserons particulièrement aux algorithmes de type stratégie d'évolution ou programmation évolutionnaire. Les distinctions particulières avec les algorithmes génétiques seront évoquées. Le cas particulier de la programmation génétique, bien que passionnante, ne sera pas plus approfondi. Nous laissons le soin au lecteur désirant plus d'informations de consulter l'un des livres suivants [KOZA92, BANZHA98, DRÉO03]. Globalement nous désignerons ces algorithmes par le terme générique de *évolutionnaires*.

3.2.1 Synopsis des méthodes évolutionnaires

Les algorithmes génétiques sont maintenant de plus en plus considérés comme un cas particulier des algorithmes évolutionnaires [DRÉO03]. Les différences les plus notables sont que:

– Les algorithmes génétiques s'inspirent fortement de la transcription génotype - phénotype de la génétique naturelle. Un génotype est une chaîne de symboles, le plus souvent binaires, qui permet de réaliser directement et facilement les opérations de croisement et mutation, sans tenir compte de la nature de la solution. Cette chaîne n'est généralement pas une représentation directe d'une solution, mais demande une transcription pour devenir une solution (le phénotype) dont on puisse évaluer la performance à un problème donné. Un individu d'un algorithme génétique peut donc être vu sous deux formes différentes suivant le type d'information que l'on manipule. Dans leur exploitation la plus fine, les algorithmes génétiques utilisent des notions encore plus ancrées dans la génétique naturelle à travers des notions de diploïdie (paires de chromosomes) et dominance (une relation importante entre génotype et phénotype).
– Les algorithmes évolutionnaires ne cherchent pas à être tellement semblables à la génétique naturelle. Dans ce cas, un individu de la population est généralement un codage explicite de la solution d'un problème qui n'a pas lieu d'être forcement binaire. Les notions de génotype, phénotype, diploïdie et dominance ne sont utilisées par les algorithmes évolutionnaires que si elles permettent de façon évidente d'améliorer l'optimisation d'un problème.

Sous ces considérations nous donnons le synopsis commun à une méthode de type évolutionnaire.

1. initialisation d'une population d'individus,
2. évaluation des performances de chaque individu,
3. sélection d'un ensemble d'individus *parents* parmi les plus performants pour la reproduction,
4. génération des individus *enfants* par croisement entre les *parents*,
5. mutation éventuelle de certains individus (*enfants* et *parents*),
6. évaluation des performances des *enfants* (et des *parents* qui ont muté),
7. regroupement de tous les individus: *parents* et *enfants*, puis sélection d'un sous-ensemble d'individus performants pour la prochaine itération,
8. Tant qu'une condition d'arrêt n'est pas vérifiée, répéter les étapes 3 à 8,
9. Retourner le meilleur individu trouvé pendant l'ensemble de ces itérations.

3.2.2 La population

Pour garder des notations communes avec la recherche avec tabous, nous conservons la notation θ pour une solution (individu). Nous noterons $\boldsymbol{\theta}_j(k)$, l'individu j de la $k^{\text{ème}}$ population. La performance d'un individu évolutionnaire est notée $f(\theta_j)$. Les individus d'une population pendant la phase d'initialisation sont généralement produits par tirage aléatoire. Dans le cas où une heuristique permettrait de produire des solutions de bonnes

qualités, il est courant de ne pas créer toute la population à travers cette heuristique, mais d'ajouter des solutions aléatoires, ceci afin d'augmenter la diversité de la population initiale. Le nombre d'individus a une importance sur les performances des algorithmes évolutionnaires: son augmentation favorise la diversité, mais accroît le coût de son utilisation. Le nombre d'individus n'est donc pas un paramètre facile à régler. Ce n'est pas le choix d'une valeur exacte qui est important, mais celui d'un ordre de grandeur convenable. La dimension de l'espace de recherche est par-contre un indicateur utile à son dimensionnement. L'efficacité des opérateurs de croisement et de mutation a également une influence sur la taille optimale de la population. D'un point de vue pratique, il est courant de réaliser des essais avec différentes tailles de population pour déterminer un bon compromis entre efficacité et coût de l'exploration.

Dans les algorithmes évolutionnaires les plus couramment utilisés, la taille de la population entre deux itérations est constante [DRÉO03, GAGNÉ05B]. Notons μ la taille de la population initiale et λ le nombre d'enfants créés à l'étape de reproduction. Dans la plupart de ces algorithmes, la reproduction est réalisée à partir du croisement de deux individus. L'étape 3 doit donc sélectionner λ couples d'individus pour la reproduction à partir d'une population de μ individus. L'étape 7 ne doit conserver que μ individus à partir d'une population de $\mu + \lambda$ individus. Il est courant de fixer le nombre d'enfants produits à chaque itération comme égal à la moité du nombre de parents ($\lambda = \mu$ est également courant).

3.2.3 Représentation d'une solution

Comme évoqué précédemment, il est usuel de coder les solutions comme des chaînes binaires, même si la solution d'un problème ne s'exprime pas naturellement sous forme binaire. Dans le cas où une solution ne se représente pas naturellement sous forme de chaîne binaire, il est alors nécessaire de définir une méthode de transformation d'une représentation à l'autre. Une raison de l'attrait pour des représentations binaires est qu'initialement les schémas de Holland définissaient un cadre théorique qui justifiait l'efficacité des opérations avec des chaînes binaires [HOLLAN75, HOLLAN92]. Une autre raison est qu'il existe un grand nombre d'opérateurs de croisement adaptés à cette représentation [GOLDBE94]. De plus, l'opérateur de mutation est simple à définir (*cf.* section 3.2.4.3). Dans le domaine de l'apprentissage artificiel, la représentation binaire est utilisée dans des algorithmes de sélection d'attributs, de catégorisation et d'édition de bases d'apprentissage [KUNCHE97A, SUN02, RÖ04]. La raison principale est que le codage de la solution est direct (génotype et phénotype sont confondus). En effet, un bit représente :

– la sélection ou rejet d'un attribut ou d'un exemple.

– l'appartenance ou non à un même groupe (classe).

La représentation binaire n'est pas toujours la plus adaptée à une optimisation par méthode évolutionnaire d'un problème donné. Il est de plus en plus courant de représenter une solution Θ comme un vecteur de variables dont le domaine de définition appartient à l'espace des entiers ou des réels [DRÉO03, GAGNÉ05B]. Les chaînes binaires sont alors un cas particulier où le domaine de définition des variables du vecteur correspondant est $\{0,1\}$ et chaque variable peut être considérée comme un entier. Comme dans le cas de la recherche avec tabous, les algorithmes évolutionnaires peuvent manipuler des structures plus complexes comme des arbres. La programmation évolutionnaire correspond à cette

catégorie. Pour cette présentation des méthodes évolutionnaires, nous nous limiterons au cas où les solutions ont une représentation vectorielle.

Nous noterons θ_i la $i^{\text{ème}}$ variable dans la solution et $\mathcal{D}(\theta_i)$ le domaine de définition de cette variable.

3.2.4 Opérateurs de base

Dans les méthodes évolutionnaires, les trois opérateurs de base sont ceux de sélection, de croisement et de mutation. Nous donnons ici une brève description de ces trois types d'opérateurs en n'abordant que les versions les plus courantes.

3.2.4.1 Opérateurs de sélection

Dans le synopsis précédent, l'opérateur de sélection est utilisé aux étapes 3 et 7. C'est un opérateur qui prend en entrée une population et produit en sortie la liste des individus sélectionnés. Dans la majorité des cas, c'est le même type d'opérateur qui est utilisé à ces deux étapes, même si d'un point de vue sémantique ces deux opérations ont une signification différente. A l'étape 3, la phase de sélection détermine quels sont les individus qui participeront aux reproductions. A l'étape 7, la phase de sélection détermine quels sont les individus qui vont survivre. Par rapport aux tailles de population généralement utilisées (*cf.* section 3.2.2), l'étape 3 doit sélectionner 2λ individus à partir de μ individus et l'étape 7 doit sélectionner μ individus parmi $\mu + \lambda$, un même individu pouvant être sélectionné plus d'une fois. L'existence de clones d'un même individu dans une population n'est problématique que si le nombre de clones d'un même individu est élevé par rapport à la taille de la population. Nous verrons dans la suite de cette section comment éviter ce problème. Par simplification, nous considérons que nous sommes dans le cas de l'étape 3 pour la suite des explications sur la sélection. La transposition à l'étape 7 de ce qui suit ne pose aucun problème.

Pression de sélection et dérive génétique

Il existe différents types d'opérateur de sélection, mais globalement ils sont tous basés sur le même principe. La probabilité qu'a un individu d'être sélectionné est d'autant plus grande qu'il est meilleur. La qualité d'un individu dépend de sa performance, mais généralement ce n'est pas la valeur de sa performance qui est directement utilisée pour mesurer son aptitude à la sélection. La raison est qu'un écart trop important entre les valeurs mesurant l'aptitude à la sélection des individus se traduit par le fait que le plus "fort" de ces individus domine l'ensemble des autres. En quelques itérations la population ne contient que des clones quasi-identiques à cet individu dominant. La conséquence est une diminution importante de la diversité de la population et par la même un risque important d'être piégé dans un minimum local. Ce phénomène dépend d'un facteur appelé *pression de sélection* qui mesure la prédominance d'un individu par rapport aux autres. Il est d'autant plus marqué que la pression de sélection est importante. Une pression de sélection trop faible est également problématique. En effet, aucun individu ne se distingue suffisamment des autres et le meilleur individu a quasiment autant de chances d'être sélectionné

que le pire individu. La conséquence est que globalement la performance moyenne de la population n'évolue plus dans le temps. Cet effet est appelé *dérive génétique* et il doit être autant combattu qu'une pression de sélection trop importante. La dérive génétique n'est pas seulement produite par une pression de sélection trop faible. Elle peut dépendre d'une méthode de sélection avec une trop grande variance, ainsi que de l'utilisation d'opérations de croisement ou de mutation non appropriés. En tenant compte de ces remarques, l'aptitude à la sélection d'un individu est déterminée à partir de $f(\boldsymbol{\theta})$ et de statistiques sur la population des parents et enfants de la génération actuelle. Des statistiques sur les générations précédentes peuvent aussi être prises en compte pour mesurer le risque de dérive génétique. Notons $\tilde{f}(\boldsymbol{\theta})$ l'aptitude d'un individu à être sélectionné vis-à-vis de la population à laquelle il appartient. Nous donnons la définition la plus couramment utilisée de la pression de sélection:

$$p_s = \frac{\hat{f}}{\bar{f}} \tag{3.6}$$

avec $\bar{f}$ la valeur moyenne des performance des individus et $\hat{f}$ la valeur de performance du meilleur individu. Une façon de contrôler la pression de sélection est d'obliger cette valeur à être constante indépendamment de la constitution de la population ($p_s = 2$ est une valeur couramment utilisée). Donnons trois solutions pour que la pression de sélection p_s ait la valeur désirée:

1. Une première solution consiste à réaliser un simple décalage $\tilde{f}(\boldsymbol{\theta}) = f(\boldsymbol{\theta}) - a$. La valeur de a dépend de la pression de sélection choisie:

$$a = \frac{p_s\bar{f} - \hat{f}}{p_s - 1} \tag{3.7}$$

   Le fait que des individus aient des valeurs négatives de $\tilde{f}\boldsymbol{\theta})$ pose problème. Pour l'éviter, il est courant d'utiliser la relation : $\tilde{f}(\boldsymbol{\theta}) = \max(f(\boldsymbol{\theta}) - a, \epsilon)$ avec ϵ une faible valeur. Ceci a pour effet de réduire la pression de sélection. Si cette réduction est trop importante, il peut être nécessaire de diminuer la valeur de a pour compenser l'effet du seuillage.
2. Une autre solution consiste à élever la fonction de performance à une puissance k adéquate pour obtenir la pression sélective désirée: $\tilde{f}(\boldsymbol{\theta}) = f(\boldsymbol{\theta})^k$. La valeur de k est par-contre plus difficile à déterminer. La méthode de Newton peut par exemple être utilisée pour déterminer la valeur de k correspondant à une pression de sélection p_s choisie.
3. Une dernière solution consiste à réaliser au préalable un classement, par ordre décroissant, de chaque individu à partir de sa performance. Notons r_i le rang d'un individu i dans une population de μ individus. L'aptitude d'un individu est déterminés par la relation suivante:

$$\tilde{f}(\boldsymbol{\theta}_i) = \left(1 - \frac{r_i}{\mu}\right)^{p_s - 1} \tag{3.8}$$

   Cette solution à l'avantage de produire une différence notable entre les meilleurs individus même si leurs performances intitiales étaient déjà très proches.

Règles de proportionnalité

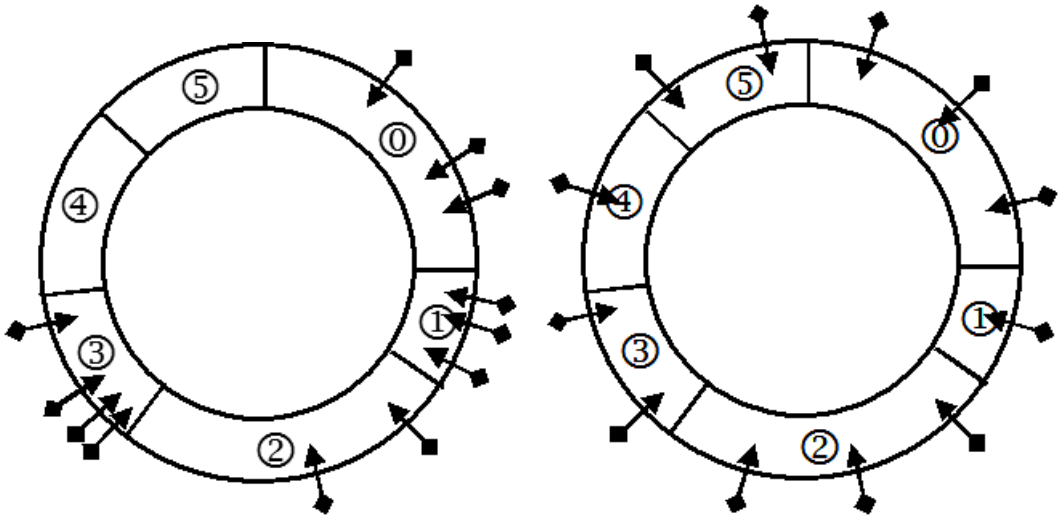
Deux grandes catégories d'opérateurs de sélection existent. Ceux qui réalisent la sélection en appliquant une règle de proportionnalité et ceux qui réalisent la sélection par tournois. Nous n'aborderons que le cas de la sélection à partir d'une règle de proportionnalité, pour la sélection par tournois, se reporter à [DRÉO03]. Nous considérons que 2λ individus doivent être sélectionnés pour réaliser λ croisements de couples d'individus.

Pour ce type de règle, le nombre espéré μ_i de sélections d'un individu i peut être déterminé par la règle de proportionnalité suivante :

$$\mu_i = \frac{2\lambda}{\sum_{i=1}^{\mu} \tilde{f}(\boldsymbol{\theta}_i)} \tilde{f}(\boldsymbol{\theta}_i) \tag{3.9}$$

A partir des valeurs μ_i, deux stratégies de répartition sont applicables. Elles sont désignées par les acronymes RWS (*Roulette Wheel Selection*) et SUS (*Stochastic Universal Sampling*).

La stratégie RWS est basée sur le principe de la roulette de casino. Il y a autant de cases sur cette roulette que d'individus et chaque case a une taille égale à μ_i. La boule est lancée autant de fois qu'il y a d'individus à sélectionner. La figure 3.1(a) illustre cette stratégie. Le problème majeur de cette première stratégie est qu'elle a une grande variance. Dans le pire des cas, seuls les individus les plus mauvais sont sélectionnés. La dérive génétique peut alors être importante.



(a) RWS (b) SUS

FIG. 3.1 – *Illusration de la stratégie RWS (a) ou SUS (b). La population initiale comporte 6 individus et 12 individus sont sélectionnés pour la phase de reproduction.*

La stratégie SUS garde l'idée générale de la méthode RWS, mais les tirages des exemples sélectionnés ne sont pas indépendants les uns des autres pour diminuer la variance et par la même le risque de dérive génétique. L'idée générale est qu'un individu soit sélectionné au minimum $\lfloor \mu_i \rfloor$ fois et au maximum $\lceil \mu_i \rceil$ fois, tout en conservant la même

espérance μ_i pour chaque individu i. Pour obtenir ce résultat, on définit pour chaque individu un intervalle $I_i = [v_i,V_i[$ avec $v_i = V_{i-1}$, $V_i = v_i + \mu_i$ et $v_0 = 0$. Une valeur $x_0 \in [0,2\lambda[$ est ensuite déterminée par tirage aléatoire uniforme. Cette première valeur désigne le premier exemple sélectionné à partir des intervalles I_i. Pour la première sélection rien ne change par rapport à la méthode RWS. Par contre, pour les $2\lambda - 1$ autres, les valeurs x_j sont déterminées par la règle suivante : $x_j = (x_0 + j) \mod 2\lambda$. Soit s_j l'indice de $j^{\text{ème}}$ individu sélectionné. s_j est alors défini comme : $s_j = i : x_j \in I_i$. La figure 3.1(b) illustre cette stratégie. Cette méthode est généralement préférée à RWS car en plus d'avoir une variance plus faible, elle sélectionne forcément au moins une fois tous les individus les plus performants (*i.e.* $\mu_i > 1$).

3.2.4.2 Opérateurs de croisement

L'idée principale est d'utiliser plusieurs parents (généralement 2) pour produire un (ou deux) descendants de meilleure performance en tirant profit des points positifs de chaque parent et en gommant leurs effets négatifs. Pour garder la possibilité d'explorer efficacement l'espace de recherche, le fait qu'un enfant soit de meilleure performance que ces parents ne doit pas être systématique, mais la probabilité que ce fait se produise doit être plus importante que l'effet contraire. Dans le cas où cet effet est trop important, il y a risque de convergence trop rapide vers un minimum local, bien que l'opérateur de mutation puisse réduire l'importance de ce risque. L'opérateur de croisement peut également produire des individus avec une très faible performance, dans ce cas, c'est l'opérateur de sélection qui évitera qu'ils ne perdurent dans la population.

Il reste à définir comment favoriser cet effet. Partons de la considération que toute solution $\boldsymbol{\theta}$ représentée par un individu est un vecteur correspondant à un ensemble de variables dont on recherche globalement les valeurs optimales. L'idée est que chaque solution (individu), même si elle n'est pas optimale, doit contenir une partie des valeurs de la solution optimale (ou proche de l'optimal) recherchée. L'opérateur de croisement à donc pour objectif d'extraire de chaque individu les bonnes bribes d'informations qu'il contient pour les transmettre aux enfants. La figure 3.2 illustre ce principe. Pour ce cas, deux enfants sont produits à partir du croisement deux parents. Il existe une grande variété d'opérateurs de croisement, nous n'aborderons ici que les plus communs [GOLDBE94, DRÉO03].

Plaçons-nous d'abord dans le cas où les solutions utilisent une représentation binaire. La règle qui consiste à produire deux enfants pour chaque croisement entre deux parents est la plus utilisée dans les approches génétiques. Pour les algorithmes évolutionnaires, il est courant de ne produire qu'un seul enfant pour chaque croisement[3]. Initialement, il était courant de ne choisir qu'un point de coupure dont la position était choisie par tirage aléatoire, ce qui consiste à échanger deux sous chaînes binaires. D'un point de vue plus général, le nombre de points de coupure peut être plus important. Ce nombre peut être fixé ou déterminé aléatoirement pour chaque croisement. La position des points de coupure peut être contrainte, les limitant à certaines régions. La représentation sous forme de chaîne peut être considérée comme cyclique (dans ce cas il faut un minimum de 2 points de coupure) pour limiter les effets de bord. A l'extrême, il y a autant de point de coupure que le nombre d'éléments binaires moins 1. Un tirage aléatoire détermine alors quel bit de l'un des deux parents est sélectionné pour construire la représentation de l'enfant.

3. Par rapport à l'exemple de la figure 3.2, cela consiste à n'utiliser qu'un des deux croisements.

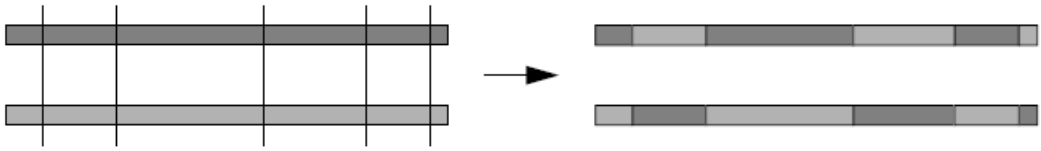


FIG. 3.2 – *Croisement entre deux parents avec 5 points de coupure pour la production de deux enfants par effet symétrique.*

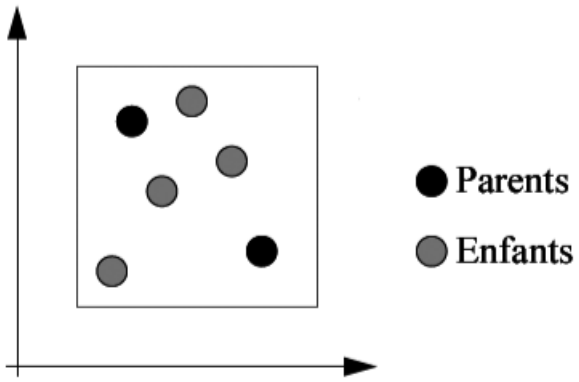


FIG. 3.3 – *Exemples d'enfants potentiels produit pour la méthode* BLX-α volumique *à partir de deux parents lorsque seulement deux variables sont utilisées. Le rectangle représente le domaine des solutions possibles lorsque* $\alpha = 0{,}25$.

Lorsque les variables manipulées ne sont pas de type binaire, mais sont de type réel, l'opération de croisement ne peut plus être définie comme une simple copie de parties des solutions correspondant à chaque parent, sinon l'espace des solutions explorées est réduit à un sous-espace des solutions possibles Θ. Pour ne pas être limité à ce sous-espace, une règle de croisement appelée *BLX-α volumique* [ESHELM97] est généralement utilisée. Supposons que l'on souhaite réaliser le croisement entre deux parents $\boldsymbol{\theta}_{j_1}$ et $\boldsymbol{\theta}_{j_2}$ pour un sous-ensemble de variables $\mathbf{S}$ dont la sélection dépend des points de coupure. La méthode produit, pour chaque variable i prise en considération, une valeur β_i, à partir d'un tirage aléatoire tiré uniformément dans l'intervalle $[-\alpha{,}1+\alpha]$. Les valeurs θ_i de l'enfant sont alors déterminées par la relation suivante:

$$\forall i \in S : \theta_i = (\theta_{i,j_1} - \theta_{i,j_2})\,\beta_i + \theta_{i,j_2} \tag{3.10}$$

Cette opération est réalisée pour tous les croisements à produire. La figure 3.3 est une illustration 2D de l'application de cet opérateur. La valeur choisie pour α a une importance significative. Dans le cas où $\alpha < (\sqrt{3}-1)/2$ le croisement est contractant, ce qui signifie que son application itérée aboutit à la concentration de la population en son centre de gravité. Pour éviter ce problème, il est courant de prendre $\alpha = 0{,}5$ même si l'opérateur de mutation réduit ce risque. Un autre inconvénient de cet opérateur est qu'il a tendance à réduire les éventuelles corrélations entre les variables. Si l'existence de corrélations est connue, il est préférable d'utiliser une variante de cet opérateur nommée *BLX-α linéaire*.

Les opérations de croisement peuvent produire des solutions qui ne font plus partie de l'espace des solutions possibles Θ, en particulier quand le problème à optimiser doit

respecter un certain nombre de contraintes. Il est alors parfois nécessaire de corriger la solution produite pour qu'elle reste valide. Par exemple l'application directe de l'opérateur BLX peut produire des valeurs en dehors du domaine de définition $\mathcal{D}(\theta_i)$ des variables et il est alors nécessaire de contraindre ces valeurs produites au domaine de définition.

3.2.4.3 Opérateur de mutation

L'opérateur de mutation a pour objectif de maintenir la diversité de la population à un niveau suffisamment important. En effet, les opérateurs de sélection et de croisement peuvent produire une dérive génétique, réduisant par là même l'espace des solutions qui pourront être explorées. L'opérateur de mutation a la possibilité de changer les éléments d'une solution de manière totalement aléatoire indépendamment des performances de ces individus. Pour éviter de réduire à néant les effets positifs des opérateurs de sélection et de croisement, il est nécessaire que ces mutations aient une fréquence relativement faible. Dans le cas d'une représentation binaire, l'opérateur de mutations réalise le changement d'état d'un bit d'une solution avec une probabilité faible ($< 1\%$). Lorsque la représentation des individus n'est pas binaire, la mutation gaussienne est la plus utilisée [DRÉO03]; elle ajoute à un individu θ une variable aléatoire gaussienne de moyenne nulle et d'écart type $\boldsymbol{\sigma}$. Comme le support n'est pas borné, il est théoriquement possible d'atteindre tout point de l'espace des solutions. En pratique le choix de la valeur de $\boldsymbol{\sigma}$ est important, la valeur peut être fixée à partir de la connaissance de la dimension de l'espace de recherche. Une autre solution est d'adapter la valeur de $\boldsymbol{\sigma}$ pendant le déroulement de l'algorithme évolutionnaire afin de contrer les effets de la dérive génétique. Dans les algorithmes évolutionnaires les plus évolués, le taux de mutation n'est pas une valeur constante, mais dépend du comportement global de l'algorithme au cours des itérations. Lorsque la performance moyenne d'une population évolue positivement au cours des itérations, il est préférable de réduire la fréquence des mutations pour ne pas perturber cet effet positif. Dans le cas contraire, où la performance moyenne de la population stagne, il est alors préférable d'augmenter cette fréquence.

3.2.5 Elitisme

Dû à leur aspect stochastique, les méthodes évolutionnaires peuvent produire une solution optimale à une itération donnée qui ne survivra pas aux actions des opérateurs de sélection, croisement et mutation. Pour éviter de perdre trace d'une telle solution, il est obligatoire de sauvegarder à tout moment la meilleure solution trouvée. Une solution pour s'assurer que le meilleur individu n'est pas perdu est d'obliger qu'il soit toujours sélectionné à chaque itération et d'interdire la possibilité de le modifier par mutation. Dans ce cas on parle d'algorithme évolutionnaire élitiste, car il conserve systématiquement le meilleur individu dans la population. Ce principe n'est pas systématique utilisé, car il peut être la cause d'une dérive génétique importante en obligeant les autres individus à rester dans le voisinage de la *solution élite*.

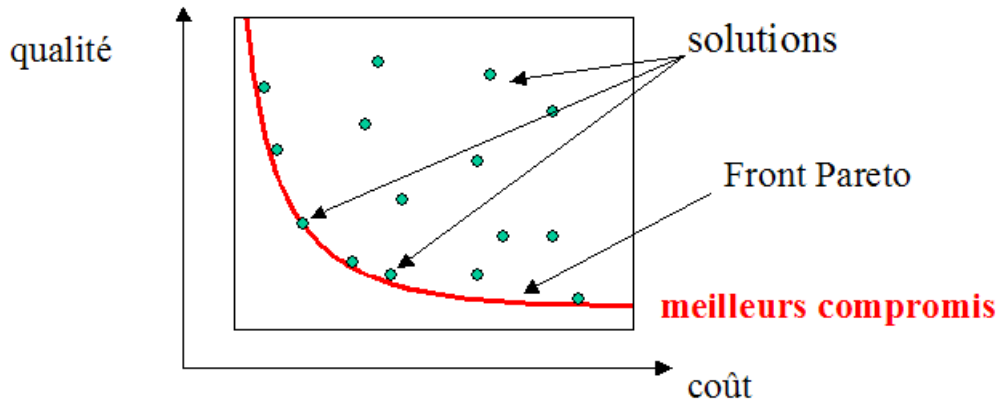


FIG. 3.4 – *Exemple de front de Pareto pour un problème multi-objectif regroupant l'optimisation de la qualité et du coût.*

3.2.6 Optimisation multi-objectif

Il est courant de devoir optimiser plusieurs objectifs en même temps. Le problème le plus courant dans notre société est de trouver des solutions à des problèmes qui soit à la fois de la meilleure qualité et du coût le plus réduit. Ce qui correspond généralement à choisir un bon compromis entre les deux. La solution la plus couramment utilisée pour réaliser cette optimisation est de définir une nouvelle fonction objectif à partir de l'ensemble des autres. Par exemple, par une combinaison linéaire de ces objectifs. Les valeurs des coefficients de cette combinaison correspondent alors à la caractérisation du compromis choisi. Avec les algorithmes génétiques, il existe une autre possibilité qui est liée à la notion de *front de Pareto*[4]. L'idée est de ne pas fixer la nature du compromis avant de réaliser l'optimistion, mais de produire des solutions correspondant à différents compromis, le choix du meilleur compromis étant reporté à une étape ultérieure. La figure 3.4 illustre une représentation du front de Pareto avec les deux objectifs classiques précédemment évoqués. Dans ce cas, l'algorithme génétique *enregistre* les solutions trouvées qui sont le long de ce front. Il serait aussi possible de réaliser cette opération avec d'autres méta-heurisitiques comme la recherche avec tabous. L'avantage des méthodes évolutionnaires est qu'il est possible de définir des mécanismes particuliers qui obligent les solutions représentées par la population à être suffisamment diversifiées pour qu'elles ne se concentrent pas toutes dans une même région, mais qu'elles soient également réparties le long de ce front [COELLO02]. Ce mécanisme permet une optimisation plus agressive de la frontière de ce front, car à travers la population, c'est directement le front qui est optimisé. Pour plus de détails sur l'optimisation multi-objectif avec des algorithmes évolutionnaires se reporter à [COELLO02, GAGNÉ05B, COLLET05].

4. Une solution appartient au front de Pareto, s'il n'existe pas de solutions connues qui la domine sur tous ces objectifs.

3.3 Autres familles de méta-heuristiques

Il existe de nombreuses méthodes méta-heuristiques. Parmi les plus connues qui n'ont pas été décrites précédemment, on peut citer les méthodes de type recuit simulé et les nombreuses méthodes désignées sous le terme de colonies de fourmis. Elles ont été appliquées au domaine de l'apprentissage pour la sélection d'attributs, la sélection d'exemples et la catégorisation [DEBUSE97, AZZAG04, AL-ANI05]. Nous dressons ici un bref aperçu de ces deux familles de méta-heuristiques. A partir de ces quatre grandes familles, de nombreuses méthodes hybrides ont été développées avec pour objectif de tirer profit des avantages de chacune de ces méthodes suivant la classe du problème à résoudre.

3.3.1 Recuit simulé

Le recuit simulé est une méta-heuristique inspirée d'un processus utilisé en métallurgie. Ce processus alterne des cycles de refroidissement lent et de réchauffage (recuit) qui tendent à minimiser l'énergie du matériau par rapport à un refroidissement constant de matériau. Le principe du recuit simulé exige que l'on accepte, occasionnellement et sous le contrôle de la *température de recuit*, une augmentation de l'énergie courante pour ne pas être piégé dans un optimum local. Par rapport à la formulation (3.2) une solution a un niveau d'énergie d'autant plus faible que sa valeur $f(\theta)$ est grande. L'algorithme du recuit simulé réalise une marche aléatoire dans l'espace des solutions. La notion de *pas*, mouvement ou transformation élémentaire de la solution est dépendante de la nature du problème. Chaque *pas* est accepté ou refusé à partir d'une règle qui s'appuie sur des résultats de la physique statistique. Si le *pas* permet de réduire le niveau d'énergie, il est forcement accepté. S'il augmente le niveau d'énergie, le *pas* est accepté avec une probabilité p qui dépend de l'augmentation du niveau d'énergie E de la température du système. Cette probabilité est déterminée par la relation suivante :

$$p = \exp\left(\frac{-\Delta E}{T}\right) \tag{3.11}$$

avec $\Delta E = f(\theta) - f(\theta + \Delta\theta)$ représentant la variation d'énergie, T la température du système et $\Delta\theta$ le *pas* réalisé. Au cours des itérations la température du système diminue globalement pour devenir nulle. Cela correspond généralement à un critère d'arrêt de l'algorithme de recuit simulé. La diminution de la température est réalisée par paliers toutes les n itérations. Pour plus de détails sur cette méthode méta-heuristique se reporter à [DRÉO03], ainsi qu'à l'ensemble des références présentes dans ce livre.

3.3.2 Colonies de fourmis

Cette approche de Colorni et al [COLORN92] s'efforce de simuler la capacité collective de résolution de certains problèmes difficiles par une colonie de fourmis, alors que chaque fourmi prise séparément a des capacités très limitées. Ces algorithmes simulent le mode de communication indirecte des fourmis via l'environnement désigné par la terme de *stigmergie* à travers l'utilisation de *phéromone*. Les algorithmes de colonies de fourmis possèdent plusieurs caractéristiques importantes qui sont:

– un parallélisme intrinsèque élevé,

– une grande flexibilité pour s'adapter à des modifications de la nature du problème,
– une robustesse importante quant à la perte d'une partie de sa population,
– la possibilité de s'auto-organiser malgré la décentralisation de l'information.

Le développement de ces algorithmes est récent et la transposition à chaque problème d'optimisation n'est pas toujours évidente. Le lecteur interessé peut faire référence aux ouvrages [BONABE99, DRÉO03] pour avoir plus d'information sur ce type d'algorithmes. Hölldober et Wilson recensent aussi une somme impressionnante de connaissances sur la biologie des fourmis dans [HÖLLDO90].

3.3.3 Méthodes hybrides et autres méta-heuristiques

Les métaheuristiques forment une classe d'algorithmes foisonnante, où beaucoup de principes sont communs à l'ensemble de ces algorithmes. Les différences entre ces catégories de méta-heuristiques deviennent floues lorsque les nombreuses variantes dans chaque catégorie sont prises en compte. Une tentative d'unification incluant la recherche avec tabous, les algorithmes évolutionnaires et les colonies de fourmis est réalisée dans [TAILLA98]. Elle est désignée sous le terme de "programmation à mémoire adaptative" et elle insite sur l'utilisation d'une forme de mémoire dans ces algorithmes, ainsi que sur l'importance des phases d'intensifications et de diversifications. Des méthodes qui réalisent l'hybridation de la recherche avec tabous et du recuit simulé ont aussi été proposées (voir par exemple [WOODRU94]). Dans le chapitre 5 de [DRÉO03] d'autres méthodes méta-heuristiques sont également disponibles, ainsi que différentes hybridations de ces méthodes.

3.4 Conclusion

Nous avons présenté dans cette section différentes méthodes méta-heuristiques. Nous avons indiqué qu'un grand nombre de problèmes d'apprentissage correspondaient à l'optimisation d'un problème difficile. A travers quelques références, nous montrons que des méthodes à base de méta-heuristiques ont déjà été appliquées avec succès à des problèmes d'apprentissage. Nous détaillons plus particulièrement dans ce chapitre les méta-heuristiques correspondant à la recherche avec tabous et aux algorithmes évolutionnaires. Les descriptions et notations utilisées pour ces deux familles seront réutilisées dans les chapitres 5 et 6.

CHAPITRE 4

ENTRAÎNEMENT D'ENSEMBLES DE SVM POUR LA VALIDATION CROISÉE

Sommaire

Dans ce chapitre nous présentons une méthode permettant d'évaluer plus rapidement l'erreur en généralisation d'un SVM à partir de l'évaluation de l'erreur de *leave one out*, d'une procédure de validation croisée en k parties ou d'une technique de *boostrap*. D'un point de vue global, nous considérons que ces 3 cas font partie de méthodes de validation croisée, bien que d'un point de vue pratique, l'accélération de leurs estimations demande des aménagements différents. L'idée directrice est basée sur le fait que lorsqu'un ensemble d'entraînements doit être réalisé avec des jeux de données qui sont corrélés entre eux, chaque entraînement avec un de ces jeux de données peut fournir des informations essentielles à la réalisation des entraînements suivants. L'exploitation cohérente de ces informations doit alors permettre de réduire, de façon significative, les temps nécessaires à la réalisation de ces entraînements, sans pour autant dégrader de manière significative la solution produite par chaque phase d'apprentissage.

La sélection d'un modèle efficace pour obtenir une fonction de décision performante en généralisation à partir de l'algorithme des SVM est crucial (*cf.* section 2.4.5.7). Il est alors indispensable de pouvoir estimer avec précision les capacités en généralisation de fonctions de décision produites par les SVM quel que soit le modèle exploité. Un modèle pour les SVM correspond[1] à l'utilisation d'une fonction noyau, au choix des valeurs des paramètres libres de la fonction noyau, de la valeur de la constante de régularisation C et du type de pénalité appliquée lorsque la notion de marge souple est exploitée. Il peut être nécessaire, en plus, de comparer des fonctions de décision suivant différentes techniques de pré-traitement des données brutes (sélection d'attributs par exemple). Dans ce cas, la technique de pré-traitement fait partie des paramètres libres du modèle à sélectionner. Les bornes supérieures sur l'erreur en généralisation ou la marge de l'hyperplan optimal ne sont pas forcément les meilleurs critères pour la sélection du modèle optimal [DUAN03]. L'estimation de l'erreur en généralisation à partir de techniques de validation croisée est généralement plus appropriée. Ces techniques permettent de comparer le pouvoir de généralisation avec d'autres méthodes d'apprentissage, lorsqu'aucune base de validation de taille suffisante n'existe. Le problème est que ces techniques ont un coût calculatoire important car elles nécessitent de nombreuses phases d'apprentissage. Ces techniques de-

1. Dans sa formulation la plus classique.

viennent alors rapidement inexploitables avec quelques centaines d'exemples lorsque le nombre de phases d'entraînement à réaliser est important. On remarque, par contre, que ce type d'estimation correspond au cadre de la production d'un ensemble de fonctions de décision, donc d'un ensemble de phases d'apprentissage, à partir de données dont la corrélation est évidente.

Nous proposons une nouvelle méthode permettant d'obtenir des résultats similaires à ces multiples entraînements, mais à partir d'un nombre fortement réduit d'opérations, comparativement à l'application directe des techniques de validation croisée (*c.f.* section 2.6). La méthode proposée s'articule autour de deux points essentiels : (1) Choisir une solution proche de l'optimal comme valeur d'initialisation d'un algorithme itératif d'entraînement des SVM, (2) définir un critère d'arrêt prématuré efficace pour cet algorithme itératif (efficace dans le sens où la solution produite a des capacités en généralisation très proches de l'optimal). Nous montrerons que ces deux actions conjointes permettent de réduire grandement le nombre d'itérations des algorithmes d'entraînement de type SMO (Sequential Minimal Optimization[2]) pour chaque phase supplémentaire d'apprentissage à réaliser.

Dans la première partie de cette étude, la formulation mathématique de la recherche d'un hyperplan optimal propre à l'algorithme des SVM est rappelée, ainsi que le critère permettant de déterminer si une solution (un hyperplan) est optimale. Les possibilités d'adapter le type de fonctions de décision produites par les SVM en fonction de la nature des données d'apprentissage utilisées est brièvement décrite. Tous ces rappels sont utiles à une description de l'algorithme SMO et des notations qui lui sont liées. Cette description insiste sur les points essentiels de la méthode SMO et sur les implications pour des méthodes qui réalisent un ensemble d'apprentissage à partir de cet algorithme.
Dans une deuxième partie, un état de l'art des techniques d'initialisation et d'arrêt prématuré de l'algorithme SMO est réalisé. Les descriptions relatives à nos méthodes sont ensuite proposées, leurs avantages par rapport aux précédentes méthodes sont exposés et des résultats d'expérimentations sont produits pour montrer leur efficacité.

4.1 Entraînement d'un SVM

Dans la section 2.4.5, nous avons insisté sur les idées et fondements théoriques de l'apprentissage avec des SVM. Dans cette section, nous donnons la formulation mathématique des problèmes d'optimisation à résoudre pour déterminer l'hyperplan optimal, tout d'abord dans le cas où les données sont linéairement séparables (marge dure), puis dans le cas où elles ne sont plus linéairement séparables (marge souple). Les deux méthodes de pénalisation pour la recherche d'une marge souple optimale sont ensuite décrites en donnant succinctement leurs avantages et inconvénients.

4.1.1 Recherche de l'hyperplan optimal

L'objectif est de déterminer l'hyperplan $h \in \mathcal{H}$ qui maximise la marge géométrique avec $\mathcal{H}$ l'ensemble des hyperplans possibles sur un espace $\mathbb{R}^n$. Cet hyperplan optimal h^*

2. Initialement proposé par Platt [PLATT99A].

définit la fonction de discrimination qui généralise au mieux (*cf.* section 2.4.5) une base d'apprentissage $Z_m = \{(\mathbf{x}_i,y_i)\}$ avec $y_i \in \{-1,+1\}$ et $\mathbf{x}_i \in \mathbb{R}^n$. Nous allons voir que ce problème peut être formulé comme un problème d'optimisation quadratique sous contraintes et qu'il existe deux formulations : (1) primaire et (2) duale. La formulation duale a pour avantage principal d'être adaptée à l'utilisation de fonctions noyaux. Considérons en premier le cas de la recherche d'un hyperplan séparant parfaitement les données, désigné comme le problème de recherche de la *marge dure optimale*.

4.1.1.1 Cas des données linéairement séparables

Soit un couple $(\mathbf{w},b)$ caractérisant un hyperplan h et le produit scalaire $\langle u,v\rangle = \sum_{i=1}^{n} u_i v_i$ dans l'espace vectoriel $\mathbb{R}^n$. On peut montrer que si:

$$\forall(\mathbf{x}_i,y_i) : y_i\left(\langle \mathbf{w},\mathbf{x}_i\rangle + b\right) \geq 0 \tag{4.1}$$

alors l'hyperplan h correspondant sépare linéairement les données d'apprentissage Z_m. Soit la fonction f qui réalise la séparation de ces données:

$$f(\mathbf{x}) = \langle \mathbf{w},\mathbf{x}\rangle + b \tag{4.2}$$

Le signe de (4.2) permet de déterminer la classe d'un exemple $\mathbf{x}$ relativement à la représentation $(\mathbf{w},b)$ d'un hyperplan séparateur h.

Un hyperplan h n'est pas défini par un seul couple unique $(\mathbf{w},b)$, mais par un ensemble de couples équivalents à partir de ces valeurs:

$$h \equiv \forall a \in \mathbb{R}^+ : (a\cdot\mathbf{w},a\cdot b). \tag{4.3}$$

La distance euclidienne d'un point $\mathbf{x}$ à un hyperplan h est égale à :

$$d(\mathbf{x},h) = \frac{|\langle a\cdot\mathbf{w},\mathbf{x}\rangle + ab|}{\|a\cdot\mathbf{w}\|} \tag{4.4}$$

avec $\|\mathbf{w}\| = \sqrt{\langle \mathbf{w},\mathbf{w}\rangle}$. Elle est évidemment indépendante de la constante a. La marge géométrique γ d'un hyperplan h qui sépare linéairement les données Z_m est définie à partir de (4.4):

$$\gamma = \min_{1\leq i\leq m} d\left(x_i,h\right) \tag{4.5}$$

Notons $\gamma(h)$, la fonction déterminant la valeur correspondant à (4.5) pour un hyperplan donné h. Déterminer l'hyperplan h^* de marge géométrique optimale γ^* correspond à résoudre le problème suivant :

$$h^* = \arg\max_{h\in\mathcal{H}} \gamma(h) \tag{4.6}$$

en respectant les contraintes (4.1). La formulation (4.6) est vérifiée pour tout couple $(\mathbf{w}^*,b^*)$ $\equiv h^*$, mais elle n'explique pas comment identifier un des couples $(\mathbf{w}^*,b^*)$ admissibles correspondant à h^*. Soit $I_{\mathrm{SV}}(h)$ l'ensemble des indices des exemples à une distance $\gamma(h)$ de l'hyperplan h:

$$I_{\mathrm{SV}}(h) = \{i | d(\mathbf{x}_i,h) = \gamma(h), 1\leq i\leq m\} \tag{4.7}$$

Ce sont les exemples définissant la marge géométrique d'un hyperplan h relativement à un ensemble Z_m linéairement séparé par h. Dans le cas de l'hyperplan optimal h^*, $I_{\mathrm{SV}}(h^*)$ identifie les exemples désignés par le terme *vecteurs de support*.
On peut remarquer qu'à partir de l'équation (4.4), on peut démontrer que $\|\gamma \cdot \mathbf{w}\| = \mathrm{cst}$. Il suffit de poser $a = \gamma$ dans (4.4) pour les exemples appartenant à $I_{\mathrm{SV}}(h)$.

On remarque que maximiser la marge géométrique est équivalent à minimiser la norme de $\mathbf{w}$. Sans perte de généralité, imposer une représentation d'un hyperplan h par un couple $(\mathbf{w},b)$ unique peut être réalisé en imposant que $\|\gamma \cdot \mathbf{w}\| = 1$. Dans ces conditions, rechercher l'hyperplan h^* de marge optimale est équivalent à minimiser le problème suivant :

$$h^* \equiv \begin{cases} (\mathbf{w}^*,b^*) = \arg\min(\|w\|) \\ \forall i \in \{1,...,m\} : y_i\left(\langle \mathbf{w},\mathbf{x}_i\rangle + b\right) \geq 1 \end{cases} \tag{4.8}$$

Pour correspondre à un problème d'optimisation quadratique, il est plus généralement formulé comme la minimisation de :

$$Q(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 = \frac{1}{2}\langle \mathbf{w},\mathbf{w}\rangle \tag{4.9}$$

sous les contraintes:

$$\forall i \in \{1,...,m\} : y_i\left(\langle \mathbf{w},\mathbf{x}_i\rangle + b\right) \geq 1 \tag{4.10}$$

Ce problème est désigné comme la *version primale* du problème des SVM. Dans le cas où l'on cherche $(\mathbf{w}^*,b^*)$ dans l'espace de redescription $\Phi(\mathcal{X})$, les contraintes (4.10) deviennent $\forall i \in \{1,...,m\} : y_i\left(\langle \mathbf{w},\Phi(\mathbf{x}_i)\rangle + b\right) \geq 1$ et $\mathbf{w}$ est un vecteur à n_Φ dimensions. Lorsque la dimension de $\mathbf{w}$ est faible, la résolution du problème primal peut être réalisée par des techniques quadratiques. Mais lorsque cette dimension est grande (ce qui se produit avec un espace de redescription de grande dimension) la résolution du problème primal devient problématique. Pour certaines fonctions Φ, elle est impossible, car n_Φ est infini. De plus, pour des valeurs raisonnables de n_Φ, le calcul effectif d'un vecteur $\Phi(\mathbf{x}_i) = (\phi_1(\mathbf{x}_i),\cdots,\phi_{n_\Phi}(\mathbf{x}_i))$ peut être lourd en temps de calcul.

L'idée est alors de convertir le problème primaire en un problème dual où le nombre de variables inconnues est égal au nombre d'exemples dans la base d'entraînement. Pour cela, le problème contraint correspondant à (4.9) et (4.10) est transformé en un problème primal non contraint :

$$Q(\mathbf{w},b,\boldsymbol{\alpha}) = \frac{1}{2}\langle \mathbf{w},\mathbf{w}\rangle - \sum_{i=1}^{m} \alpha_i \left\{ y_i\left(\langle \mathbf{w},\Phi\left(\mathbf{x}_i\right)\rangle + b\right) - 1\right\} \tag{4.11}$$

avec le vecteur $\boldsymbol{\alpha} = (\alpha_1,\cdots,\alpha_m)$ correspondant aux multiplicateurs de Lagrange. La solution optimale du problème primal correspond au point selle de la la fonction (4.11). Elle est caractérisée (en partie) par les conditions de Karush-Kuhn-Tucker (KKT) (voir par exemple le chapitre 5 de [CRISTI00] pour plus d'informations sur la théorie lagrangienne correspondante). On en déduit les conditions d'optimalité dans le dual suivantes :

$$\frac{\partial Q(\mathbf{w},b,\boldsymbol{\alpha})}{\partial \mathbf{w}} = 0 \tag{4.12}$$

$$\frac{\partial Q(\mathbf{w},b,\boldsymbol{\alpha})}{\partial b} = 0 \tag{4.13}$$

$$\forall i : \alpha_i \left\{ y_i \left(\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle + b \right) - 1 \right\} = 0 \tag{4.14}$$

$$\forall i : \alpha_i > 0 \tag{4.15}$$

La relation entre les contraintes d'inégalité (4.10) et d'égalité (4.14) sur les multiplicateurs de Lagrange associés est appelée *conditions de complémentarité KKT*. De (4.14) on déduit que soit $\alpha_i = 0$, soit $\alpha_i \neq 0$ et $y_i \left(\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle + b \right) = 1$. Les exemples qui sont vecteurs de support tel que définis par (4.7) vérifient donc que: $\forall i \in I_{\text{SV}} : \alpha_i > 0$. Ce qui illustre une autre relation entre la version primale et duale.

Utilisant (4.11), les conditions (4.12) et (4.13) au point selle peuvent être reformulées respectivement en :

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \Phi(\mathbf{x}_i) \tag{4.16}$$

et

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \tag{4.17}$$

En substituant (4.16) et (4.17) dans (4.11), on obtient le problème dual de maximisation de :

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle \tag{4.18}$$

Avec les contraintes (4.15) et (4.17) sur les multiplicateurs de Lagrange. On peut remarquer qu'en plus de limiter le nombre de variables au nombre d'exemples, la vérification de toutes les contraintes du problème dual pour une solution $\boldsymbol{\alpha}$ est plus facile.

Si une fonction noyau $K(\cdot,\cdot)$ appropriée est utilisée (*cf.* section 4.1.5.1), l'équation (4.18) utilisant cette fonction noyau peut être reformulée comme :

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{4.19}$$

Si les données sont linéairement séparables dans l'espace de redescription les valeurs optimales de (4.9) et (4.19), correspondant respectivement aux problèmes d'optimisation quadratique primale et duale, sont identiques ($Q(\mathbf{w}^*, b^*) = Q(\boldsymbol{\alpha}^*)$). On parle alors de *saut de dualité nul* (voir [CRISTI00] pour plus de détails). A partir de l'équation (4.16), la valeur de la marge géométrique peut être reformulée à partir des valeurs des variables du problème dual :

$$\frac{1}{\gamma^2} = \langle \mathbf{w}^*, \mathbf{w}^* \rangle = \sum_{i,j=1}^{m} \alpha_i^* \alpha_j^* y_i y_j \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle = \sum_{i,j=1}^{m} \alpha_i^* \alpha_j^* y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{4.20}$$

Dû au fait que le *saut de dualité* est nul, on déduit des équations (4.20), (4.18) et (4.9) que la marge optimale peut être exprimée comme :

$$\frac{1}{\gamma^2} = \sum_{i=1}^{m} \alpha_i^* \tag{4.21}$$

En utilisant (4.16), l'expression de (4.2) utile à la définition de la fonction de décision $D(\mathbf{x}) = \text{sign}(f(\mathbf{x}))$ des SVM est reformulée de la façon suivante :

$$f(\mathbf{x}) = \langle \mathbf{w}^*, \Phi(\mathbf{x}) \rangle + b^* = \sum_{i \in I_{\text{SV}}} \alpha_i^* y_i K(\mathbf{x}, \mathbf{x}_i) + b^* \tag{4.22}$$

La valeur de b^* est déterminée à partir d'un des vecteurs de support. En effet, il est établi que : $\forall i \in I_{\text{SV}} : f(\mathbf{x}_i) = 1$. On en déduit que[3]:

$$\forall i \in I_{\text{SV}} : b = 1 - \sum_{j \in I_{\text{SV}}} \alpha_j^* y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{4.23}$$

4.1.1.2 Cas des données non-linéairement séparables

Dans le cas où les données ne sont pas linéairement séparables dans l'espace de redescription utilisé, le problème de la recherche d'une *marge dure* optimale est insoluble. Comme évoqué précédemment, l'utilisation de la notion de marge souple permet d'autoriser certains exemples à ne pas respecter la contrainte (4.10). Cela est réalisé par l'introduction de variables d'écart ξ_i positives ou nulles ($\xi_i > 0$). Il y a autant de variables d'écart que de nombre d'exemples. La contrainte (4.10) est relâchée de la façon suivante :

$$\forall i \in \{1, \cdots, m\} : y_i \left(\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle + b \right) \geq 1 - \xi_i \tag{4.24}$$

On peut remarquer que si, pour un exemple i, on a $\xi_i > 1$, alors on autorise un exemple à pouvoir être mal classé. L'objectif est maintenant de trouver l'hyperplan optimal de marge optimale γ^* qui réalise le minimum d'erreurs. Cette optimisation directe n'est pas possible, car elle correspond à un problème d'optimisation combinatoire trop difficile [VAPNIK98, CRISTI00]. A la place on cherche à résoudre le problème de minimisation de :

$$Q(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2 + \frac{C}{p} \sum_{i=1}^{m} \xi_i^p \tag{4.25}$$

sous les contraintes (4.24). La variable p correspond au type de pénalité appliquée pour les exemples dont les variables d'écart ne sont pas nulles ($p \in \{1,2\}$) et C est une constante permettant de régler l'importance de cette pénalité par rapport à la marge que l'on cherche à optimiser. Si C tend vers l'infini, on retrouve le problème d'optimisation de marge dure. Il est évident que sous cette formulation, tous les problèmes sont solubles. Par contre la formulation du problème dual dépend du type de pénalité appliquée. Nous rappelons ici leurs formulations. Leurs démonstrations sont dans le principe similaires à celles de la section 4.1.1.1, pour plus de détails se reporter par exemple à [VAPNIK98, CRISTI00, ABE05].

Problème dual avec $p = 1$

Maximiser:

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

3. En pratique, on prend pour b^* la moyenne de tout les b_i^* déterminés à partir de chaque vecteur de support i. La raison est que numériquement, la relation $f(\mathbf{x}_i) = 1$ n'est pas exactement vérifiée ($f(\mathbf{x}_i) \approx 1$). Dans [LOOSLI06A] page 39, une autre méthode est proposée pour déterminer la valeur de b.

sous les contraintes:

$$\sum_{i=1}^{m} \alpha_i y_i = 0$$
$$\forall i \in \{1, \cdots, m\} : 0 \leq \alpha_i \leq C \tag{4.26}$$

Problème dual avec avec $p = 2$

Maximiser:

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \left(K(\mathbf{x}_i, \mathbf{x}_j) + \frac{\delta_{i,j}}{C} \right) \tag{4.27}$$

sous les contraintes:

$$\sum_{i=1}^{m} \alpha_i y_i = 0$$
$$\forall i \in \{1, \cdots, m\} : 0 \leq \alpha_i$$

avec $\delta_{i,j}$ la fonction de Kronecker égale à 1 si $i = j$ et à 0 sinon.

On peut vérifier que lorsque C tend vers l'infini, on retrouve la formulation du problème dual de la marge dure pour les deux types de pénalisation.

La pénalisation se traduit, dans le cas où $p = 1$ dans l'expression duale, par une limitation des valeurs maximum des multiplicateurs de Lagrange. Dans le cas où $p = 2$, cela correspond à modifier la fonction noyau: $\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j) + \delta_{i,j}/C$, ainsi que la fonction de projection (voir [CRISTI00] pour plus de détails) :

$$\tilde{\Phi}(\mathbf{x}_i) = \begin{pmatrix} \Phi(\mathbf{x}_i) \\ \frac{y_i}{\sqrt{C}} \mathbf{e}_i \end{pmatrix} \tag{4.28}$$

avec $\mathbf{e}_i$ un vecteur de dimension m dont tous les éléments sont à zéro sauf le $i^{\text{ème}}$. On peut remarquer que dans cet espace de redescription, les données seront forcément linéairement séparables car chaque exemple a un attribut qui code sa classe.

Pour la détermination de la valeur de la marge optimale, seule l'expression (4.14) peut être utilisée dans le cas où $p = 1$. Dans le cas où $p = 2$, en plus de l'expression (4.14), il existe une formulation ne dépendant que des valeurs des multiplicateurs de Lagrange :

$$\frac{1}{\gamma^2} = \left(\sum_{i=1}^{m} \alpha_i^* \right) - \frac{1}{C} \langle \boldsymbol{\alpha}^*, \boldsymbol{\alpha}^* \rangle \tag{4.29}$$

La fonction de décision a toujours la même expression (4.22). On peut par contre distinguer deux types de vecteurs de support. Ceux pour lesquels les variables d'écart sont nulles et les autres. Les premiers sont à une distance γ^* de la surface de décision et les deuxièmes à une distance inférieure à γ^*. Conservons la même notation que dans le cas de la *marge dure* pour l'ensemble des vecteurs de support indifféremment de leur catégorisation suivant les deux possibilités précédemment évoquées.

$$I_{\text{SV}}(\boldsymbol{\alpha}) = \{i | \alpha_i > 0, 1 \leq i \leq m\} \tag{4.30}$$

Nous désignons les vecteurs de support de la première catégorie comme les vecteurs de support non limités et ceux de la deuxième catégorie comme les vecteurs de support limités[4]. Les deux ensembles les caractérisant seront notés respectivement $I_{\mathrm{SV}\otimes}$ et $I_{\mathrm{SV}\ominus}$ avec $I_{\mathrm{SV}\otimes} \cup I_{\mathrm{SV}\ominus} = I_{\mathrm{SV}}$. Les définitions de ces deux ensembles sont les suivantes :

$$I_{\mathrm{SV}\otimes}(\boldsymbol{\alpha}) = \{i|\alpha_i > 0,\xi_i = 0,\, 1 \leq i \leq m\} \tag{4.31}$$

$$I_{\mathrm{SV}\ominus}(\boldsymbol{\alpha}) = \{i|\alpha_i > 0,\xi_i > 0,\, 1 \leq i \leq m\} \tag{4.32}$$

Dans le cas d'une pénalisation linéaire, les vecteurs de support limités sont ceux dont les multiplicateurs de Lagrange correspondants sont égaux à C. Pour la détermination de la valeur de b^*, seuls les vecteurs de support non limités peuvent être utilisés. Ce qui correspond à reformuler (4.23) de la façon suivante :

$$\forall i \in I_{\mathrm{SV}\otimes} : b = 1 - \sum_{j \in I_{\mathrm{SV}}} \alpha_j^* y_j K(\mathbf{x}_i,\mathbf{x}_j) \tag{4.33}$$

Choix du type de pénalisation

La pénalisation qui utilise $p = 1$ est généralement préférée dans le cas de la classification supervisée, car elle est plus proche de la notion de minimisation du nombre d'erreurs. Pour illustrer cette remarque, prenons le cas où un exemple a une variable d'écart plus grande que les autres (*i.e.* il est plus difficile à classer correctement que les autres), alors les erreurs plus faibles sur les autres exemples auront un effet mineur sur la pénalisation par rapport à celle du premier cas. La pénalisation en $p = 2$ est préférée dans le cas où les SVM sont adaptés au problème de régression [CRISTI00]. Dans ce cas, la méthode est plus proche d'une minimisation de l'erreur quadratique.

4.1.2 Solution optimale

De la même façon que l'on a utilisé la représentation lagrangienne pour formuler l'expression de la version duale du problème primal de la recherche d'un hyperplan optimal, il est possible, à partir de l'expression (4.18) et des contraintes qui lui sont liées (4.15) et (4.17), de définir la formulation lagrangienne correspondante au problème dual. La détermination des critères caractérisant le point selle permet de définir les conditions que doit satisfaire une solution optimale $\boldsymbol{\alpha}^*$. Posons $Q_{i,j} = y_i y_j K(\mathbf{x}_i,\mathbf{x}_j)$ dans le où la pénalisation utilise une $p = 1$ et $Q_{i,j} = y_i y_j \left(K(\mathbf{x}_i,\mathbf{x}_j) + \frac{\delta_{i,j}}{C}\right)$ dans le cas où la pénalisation utilise une $p = 2$. L'expression lagrangienne du problème dual devient :

$$L(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i \alpha_j Q_{i,j} - \sum_{i=1}^{m} \alpha_i - \sum_{i=1}^{m} \delta_i \alpha_i - \beta \sum_{i=1}^{m} y_i \alpha_i + \sum_{i=1}^{m} \mu_i(\alpha_i - C) \tag{4.34}$$

avec μ_i, δ_i et β les mutliplicateurs de Lagrange associés (remarque: μ_i et le terme associé dans (4.34) n'interviennent pas dans la version avec pénalisation en $p = 2$, car les α_i ne

4. Limités dans le sens qu'ils ne peuvent pas être à une distance γ^* de la frontière de décision.

sont pas limités à C, il en sera de même pour les expressions les utilisant dans la suite de cette section). Posons:

$$F_i = y_i(\sum_{j=1}^{m} \alpha_j Q_{i,j} - 1) \tag{4.35}$$

Les conditions KKT qui définissent le point selle sont les suivantes:

$$\forall i : \frac{L}{\partial \alpha_i} = (F_i - \beta)y_i - \delta_i + \mu_i, \delta_i \geq 0, \mu_i \geq 0, \delta_i \alpha_i = 0, \mu_i(\alpha_i - C) = 0 \tag{4.36}$$

Ces conditions peuvent être simplifiées en considérant les trois cas :

1. $\alpha_i = 0$:
$$\delta_i \geq 0, \mu_i = 0 \rightarrow (F_i - \beta)y_i \geq 0 \tag{4.37}$$
2. $0 < \alpha_i < C$:
$$\delta_i = 0, \mu_i = 0 \rightarrow (F_i - \beta)y_i = 0 \tag{4.38}$$
3. $\alpha_i = C$:
$$\delta_i = 0, \mu_i > 0 \rightarrow (F_i - \beta)y_i \leq 0 \tag{4.39}$$

Définissons les deux ensembles d'indices suivants :

$$I_{\text{up}}(\boldsymbol{\alpha}) \equiv \{i|(\alpha_i < C, y_i = +1) \ \vee \ (\alpha_i > 0, y_i = -1)\} \tag{4.40}$$

et

$$I_{\text{low}}(\boldsymbol{\alpha}) \equiv \{i|(\alpha_i < C, y_i = -1) \ \vee \ (\alpha_i > 0, y_i = +1)\} \tag{4.41}$$

On peut exprimer la condition KKT qui vérifie qu'une solution est bien optimale de la façon suivante:

$$\exists \beta : \forall i \in I_{\text{low}}(\boldsymbol{\alpha}), j \in I_{\text{up}}(\boldsymbol{\alpha}) : F_i \leq \beta \leq F_j \rightarrow \boldsymbol{\alpha} = \arg\max\left(Q\left(\boldsymbol{\alpha}\right)\right) \tag{4.42}$$

Une autre formulation qui ne fait pas intervenir β et qui est donc plus facile à utiliser en pratique est la suivante:

$$F_{i_{\text{low}}(\boldsymbol{\alpha})} < F_{i_{\text{up}}(\boldsymbol{\alpha})} \rightarrow \boldsymbol{\alpha} = \arg\max\left(Q\left(\boldsymbol{\alpha}\right)\right) \tag{4.43}$$

avec

$$i_{\text{low}}(\boldsymbol{\alpha}) = \arg\max_{i \in I_{\text{low}}(\boldsymbol{\alpha})} F_i \tag{4.44}$$

$$i_{\text{up}}(\boldsymbol{\alpha}) = \arg\min_{i \in I_{\text{up}}(\boldsymbol{\alpha})} F_i \tag{4.45}$$

On peut remarquer, à partir de (4.38), que lorsque la solution $\boldsymbol{\alpha}$ est optimale, la valeur de b^* est directement déterminée par les valeurs des F_i correspondant aux vecteurs de support.

Dans le cas où la solution $\boldsymbol{\alpha}$ n'est pas optimale, les deux multiplicateurs de Lagrange définies par i_{low} et i_{up} ont une signification particulière. Ils correspondent, pour une solution $\boldsymbol{\alpha}$ donnée, aux deux exemples qui violent le plus les conditions KKT. Nous verrons par la suite qu'ils jouent un rôle particulier avec les algorithmes de type SMO.

Un autre point important est que pour des raisons de convergence asymptotique, on n'utilise pas le critère (4.43) directement, mais la formulation suivante:

$$F_{i_{\text{low}}(\boldsymbol{\alpha})} < F_{i_{\text{up}}(\boldsymbol{\alpha})} + \epsilon \rightarrow \boldsymbol{\alpha} = \arg\max\left(Q\left(\boldsymbol{\alpha}\right)\right) \tag{4.46}$$

La valeur de ϵ permet d'annuler les problèmes numériques qui font que pour une valeur nulle de ϵ, la convergence n'est pas vérifiée expérimentalement. Une valeur courante pour ϵ est 10^{-3}.

4.1.3 Méthodes d'entraînement

Maintenant que nous pouvons déterminer qu'une solution est optimale grâce à la condition (4.43), reste le problème de produire la solution $\boldsymbol{\alpha}$. Au cours des dernières années plusieurs méthodes ont été proposées. La première est d'utiliser directement un solveur de problème quadratique classique. Les méthodes de ce type utilisent des techniques à base de descente de gradient[5]. Le nombre d'exemples devient rapidement un problème car il est nécessaire de manipuler des matrices de taille $m \times m$. En 1997, Osuna et al [OSUNA97] ont prouvé qu'un large problème de programmation quadratique pouvait être décomposé en une résolution d'une série de sous-problèmes, si certaines conditions sont respectées. Ils ont défini un algorithme pour le problème des SVM qui permet de traiter des bases de tailles plus importantes. Platt [PLATT99A] a ensuite poussé ce raisonnement à l'extrême en proposant une solution où chaque sous-problème ne contient que 2 exemples. Des améliorations ont par la suite été proposées à sa méthode [KEERTH01, FAN05].

4.1.4 Optimisation Séquentielle par modification Minimale (SMO)

Nous allons présenter, dans les grandes lignes, les principes essentiels des algorithmes de type SMO. Cette présentation est aussi orientée pour aider à la compréhension de la deuxième partie de ce chapitre. Avant de commencer la description de cet algorithme, introduisons une nouvelle notation, à partir de l'expression (4.35), pour réduire la longueur de certaines expressions utilisées dans cette section et les suivantes :

$$G_i = y_i Fi = \sum_{j=1}^{m} \alpha_j Q_{i,j} - 1 \tag{4.47}$$

Nous notons également par $\mathbf{G} = (G_1, \cdots, G_m)$, le vecteur qui contient l'ensemble des valeurs des G_i. Insistons sur le fait que les valeurs de $\mathbf{G}$ sont directement exploitées dans les implémentations de SMO. Elles sont en particulier utiles à la détermination du statut optimal d'une solution $\boldsymbol{\alpha}$ (*cf.* section 4.1.2).

Dans son utilisation classique (*i.e.* initialisation courante) l'algorithme SMO commence par une solution $\boldsymbol{\alpha} = 0$. Il choisit deux exemples tels qu'il existe une modification de leurs multiplicateurs de Lagrange qui permette d'augmenter la valeur de la fonction

5. Il existe des méthodes basées sur l'utilisation de techniques de point intérieur adapté au problème quadratique [VANDER99] (voir le chapitre 5 de [ABE05] pour plus de détails sur leur utilisation dans le cadre des SVM).

objectif $Q(\boldsymbol{\alpha})$. Il modifie ensuite la valeur de ces deux variables de façon à ce que l'augmentation de $Q(\boldsymbol{\alpha})$ soit maximale. Puis il répète cette action avec deux autres exemples, jusqu'à atteindre l'optimum. Comme la fonction est convexe, ce processus est certain d'atteindre l'optimum global. Dû à des problèmes de convergence numérique [KEERTH01], c'est le critère (4.46) qui est utilisé pour vérifier que l'optimum est atteint.
Lorsque l'on n'agit que sur deux variables α d'indice i_1 et i_2 en ajoutant les quantités $\Delta\alpha_{i_1}$ et $\Delta\alpha_{i_2}$, la variation de $Q(\boldsymbol{\alpha})$ est égale à (résultat déductible de (4.19)):

$$\begin{aligned}\Delta Q(\Delta\boldsymbol{\alpha}) = \quad & \Delta\alpha_{i_1}G_{i_1} + \Delta\alpha_{i_2}G_{i_2} + \Delta\alpha_{i_1}\Delta\alpha_{i_2}Q_{i_1,i_2} \\ & + \tfrac{1}{2}\left(\Delta\alpha_{i_1}^2 Q_{i_1,i_1} + \Delta\alpha_{i_2}^2 Q_{i_2,i_2}\right)\end{aligned} \tag{4.48}$$

De plus, dû à la contrainte (4.17), on a :

$$y_{i_1}\Delta\alpha_{i_1} + y_{i_2}\Delta\alpha_{i_2} = 0 \tag{4.49}$$

On remarque donc que le calcul de la variation $\Delta\alpha_{i_1}$ (et donc de $\Delta\alpha_{i_2}$) pour que ΔQ soit maximale dépend d'une simple équation du second degré à une seule variable. La determination est triviale, mis à part le fait qu'il faut tenir compte des contraintes de boîtes sur les valeurs minimum et maximum des α_i.

D'un point de vue plus général, l'algorithme SMO est exploitable pour différentes conditions d'initialisation et d'arrêt. En effet, toute solution $\boldsymbol{\alpha}$ qui vérifie $\forall i : 0 \leq \alpha_i \leq C$ est une solution possible d'initialisation pour SMO. Si $Q(\boldsymbol{\alpha}) > 0$ pour cette valeur d'initialisation, alors le nombre d'itérations nécessaires pour atteindre l'optimum sera plus faible que pour $\boldsymbol{\alpha} = 0$. Un autre point important est que les dernières itérations de l'algorithme SMO n'apportent pas grand changement à la fonction de décision correspondante. Pour cela, la valeur de ϵ peut être considérée comme un paramètre libre. S'il est judicieusement choisi, il permet de réduire le nombre d'itérations nécessaires à l'arrêt de l'algorithme SMO, sans pour autant dégrader le comportement de la fonction de décision produite.

Un autre point important est que si l'algorithme SMO doit être utilisé à partir d'un point quelconque d'initialisation différent de $\boldsymbol{\alpha} = 0$, alors si les valeurs de $\mathbf{G}$ sont déjà connues, il est préférable de les fournir à l'initialisation, car les temps nécessaires aux calculs de l'ensemble des valeurs de $\mathbf{G}$ ne sont pas négligeables, comme l'illustre l'expression (4.47). Donnons maintenant un synopsis de l'algorithme SMO qui prend en compte la possibilité d'initialiser différemment et de spécifier la tolérance sur le critère d'arrêt :

Algorithme 1 SMO($\boldsymbol{\alpha}$,$\mathbf{G}$,ϵ)

tantque ARRÊT($\boldsymbol{\alpha}$) $> \epsilon$ **faire**
(i_1,i_2)=MEILLEURCOUPLE($\mathbf{G}$)
($\Delta\alpha_{i_1}$,$\Delta\alpha_{i_2}$)=VARIATIONOPTIMALE(i_1,i_2)
($\boldsymbol{\alpha}$,$\mathbf{G}$)=MISEAJOUR($\boldsymbol{\alpha}$,$\mathbf{G}$,(i_1,$\Delta\alpha_{i_1}$),(i_2,$\Delta\alpha_{i_2}$))
fin tantque

Détaillons les différentes sous-parties de cet algorithme.

ARRÊT

Cette fonction détermine si l'expression (4.46) est vérifiée :

$$\text{ARRÊT}(\boldsymbol{\alpha}) = F_{i_{\text{low}}(\boldsymbol{\alpha})} - F_{i_{\text{up}}(\boldsymbol{\alpha})} \tag{4.50}$$

MEILLEURCOUPLE

C'est l'étape la plus importante de l'algorithme SMO. Si le couple de variables choisi ne permet qu'une faible augmentation de ΔQ, alors la convergence de SMO est très lente. La solution la plus simple consisterait à tester tous les couples et à choisir celui produisant la plus grande augmentation. Malheureusement la complexité est en $O(m^2)$ ce qui est trop contraignant pour une simple itération de l'algorithme. Des heuristiques ont alors été proposées. La plus couramment utilisée correspond à choisir les deux exemples violant le plus les contraintes KKT. Soit $i_1 = i_{\text{low}}(\boldsymbol{\alpha})$ et $i_2 = i_{\text{up}}(\boldsymbol{\alpha})$. De plus, la sélection de ce couple est relatée [HUSH03] comme étant une approximation du premier ordre de $Q(\boldsymbol{\alpha} + \Delta\boldsymbol{\alpha}) - Q(\boldsymbol{\alpha})$.
Une autre possibilité est d'utiliser un critère basé sur une approximation du second ordre [FAN05]. Le principe est de choisir i_1 ou i_2 en utilisant la régle précédente, puis de choisir le deuxième indice en cherchant celui qui maximise l'expression (4.48). Les expérimentations dans [FAN05] tendent à montrer que ce critère est plus performant que le précédent. La recherche d'une méthode de sélection de i_1,i_2 telle que ΔQ soit le plus proche possible de l'optimal avec une complexité réduite reste un domaine de recherche actif.

VARIATIONOPTIMALE

Après sélection des deux multiplicateurs α_{i_1} et α_{i_2} à modifier, cette étape détermine les modifications $\Delta\alpha_{i_1}$ et $\Delta\alpha_{i_2}$ à appliquer pour avoir une augmentation maximum de la valeur de la fonction objectif. Comme vu précédemment, la valeur de $\Delta\alpha_{i_1}$ est facile à déterminer à partir de (4.48) et est égale à :

$$\Delta\alpha_{i_1} = \frac{G_{i_2} - G_{i_1}}{Q_{i_1,i_1} + Q_{i_2,i_2} - s2Q_{i_1,i_2}} \tag{4.51}$$

avec $s = y_{i_1} y_{i_2}$. La valeur de $\Delta\alpha_{i_2}$ est obtenue directement par :

$$\Delta\alpha_{i_2} = -s\Delta\alpha_{i_1} \tag{4.52}$$

Comme les formulations (4.51) et (4.52) ne tiennent pas compte du fait que $0 \leq \alpha + \Delta\alpha \leq C$ doit être vérifié pour les deux variables, la fonction VARIATIONOPTIMALE retourne la plus grande variation possible en respectant ces contraintes.

MISEAJOUR

Cette étape réalise la mise à jour des valeurs des deux variables à modifier dans $\boldsymbol{\alpha}$. Elle réalise en particulier la mise à jour de $\mathbf{G}$:

$$\forall i \in \{1, \cdots, m\} : G_i \leftarrow G_i + \Delta\alpha_{i_1} Q_{i_1,i} + \Delta\alpha_{i_2} Q_{i_2,i} \tag{4.53}$$

Bien que cela ne soit pas nécessaire pour produire la fonction de décision, cette étape met à jour la valeur de la fonction objectif à partir de l'équation (4.48).

Pour terminer avec SMO, indiquons que la librairie logicielle: *libsvm* [CHANG01] est actuellement la version la plus évoluée implémentant cet algorithme.

4.1.4.1 Utilisation d'un cache

La plupart des implémentations de SMO utilisent un cache pour stocker les valeurs de $Q_{i,j}$ déjà calculées, ceci afin de réduire les temps d'entraînements des SVM lorsque le nombre d'exemples est important. En effet, s'il n'est pas possible de stocker l'ensemble des m^2 valeurs de la matrice de Gram lorsque le nombre m d'exemples est important, le calcul systématique des valeurs des $Q_{i,j}$ à partir des valeurs $K(\mathbf{x}_i,\mathbf{x}_j)$ pour des couples (i,j) précédemment utilisés est une perte de temps non négligeable, en particulier lorsque le nombre d'attributs caractérisant chaque exemple est important. L'utilisation de techniques de cache performantes [PLATT99A, CHANG01] permet de réduire significativement ces temps d'entraînements.

4.1.4.2 Technique de Shrinking

L'utilisation de la technique de Shrinking permet de réduire le nombre de variables α_i modifiables par chaque itération de SMO. Cela a pour effet de réduire la durée d'apprentissage si les variables modifiables sont correctement choisis. L'idée est de déterminer à partir des premières itérations quels sont les vecteurs de support non limités potentiels et de chercher à ne modifier que les α_i leur correspondant. Lorsque la solution est optimale par rapport à ce nombre réduit de variables , il est vérifié que la solution globale est optimale et dans le cas contraire un nouveau jeu de variables modifiables est déterminé. Plusieurs techniques ont été proposées pour réaliser cette sélection [PLATT99A, COLLOB01, DONG03, FAN05]. Pour plus de détails, se reporter aux articles référencés précédemment.

4.1.5 Adaptabilité des SVM à un problème donné

Le problème relatif aux SVM définit un cadre général à la résolution de problèmes d'apprentissage supervisé. Pour obtenir des résultats satisfaisants, il est nécessaire de tenir compte des particularités du problème. Le premier point concerne le choix d'un noyau. Une fois le noyau choisi, il est important de régler correctement les valeurs libres du SVM. Cette partie correspond à la sélection du modèle (modèle se limitant aux hyper-paramètres du SVM pour cette section). D'autres informations, facilement accessibles à partir d'une base d'entraînement peuvent être prises en compte avant d'utiliser directement l'algorithme des SVM sur ces données. La première est la proportion d'exemples de chaque classe. La deuxième est la présence de doublons parmi les exemples de la base d'entraînement. Les sections suivantes permettent de comprendre comment ces différentes notions sont prises en compte avec les SVM.

4.1.5.1 Les fonctions noyaux

Il est en général beaucoup plus naturel de définir une fonction noyau $K(\mathbf{x}_i,\mathbf{x}_j)$ qui vérifie la condition $K(\mathbf{x}_i,\mathbf{x}_j) = \langle\Phi(\mathbf{x}_i),\Phi(\mathbf{x}_j)\rangle$ plutôt que de créer une fonction Φ de projection qui permette d'obtenir un espace $\mathcal{H}$ suffisamment "flexible" et ensuite de rechercher si une formulation simple de K existe pour que la relation $\langle\Phi(\mathbf{x}_i),\Phi(\mathbf{x}_j)\rangle \equiv K(\mathbf{x}_i,\mathbf{x}_j)$ soit vérifiée. De plus, des phénomènes de sur-apprentissage peuvent apparaître si la marge

correspondante est faible pour les données prises dans cet espace de redescription préconstuit, s'il est trop bien adapté aux données d'entraînement.

On ne peut pas pour autant définir une fonction noyau K sans prendre certaines précautions. En particulier, il est indispensable que l'espace de redecription existe et donc que la fonction Φ existe, même si elle n'est pas facilement exprimable. Dans le cas contraire, le problème de recherche d'un hyperplan optimal n'a pas lieu d'exister[6].

Propriétés des fonctions noyaux

Une fonction noyau K est une fonction symétrique,

$$\forall \mathbf{x}_1,\mathbf{x}_2 \in \mathcal{X} : K(\mathbf{x}_1,\mathbf{x}_2) = K(\mathbf{x}_2,\mathbf{x}_1) \tag{4.54}$$

qui vérifie l'inégalité de Cauchy-Shwarz,

$$\forall \mathbf{x}_1,\mathbf{x}_2 \in \mathcal{X} : K(\mathbf{x}_1,\mathbf{x}_2)^2 \leq K(\mathbf{x}_1,\mathbf{x}_1)K(\mathbf{x}_2,\mathbf{x}_2) \tag{4.55}$$

Cependant ces conditions ne sont pas suffisantes pour garantir l'existence d'un espace de redescription dans lequel K correspond à un produit scalaire. La fonction noyau K doit aussi vérifier le théorème de Mercer [MERCER09] (pour plus de détail sur ce théorème se reporter au chapitre 3 de [CRISTI00]). Une des conséquences de ce théorème est que toute matrice de Gram

$$\mathbf{K} = (K(\mathbf{x}_i,\mathbf{x}_j))_{i,j=1}^{m} \tag{4.56}$$

est définie semi-positive (elle n'a pas de valeur propre négative).

L'astuce des fonctions noyaux

C'est la possibilité de tranformer une fonction noyau dans un problème en une autre fonction noyau tout en gardant les propriétés de la méthode les utilisant. La formulation duale des SVM (*cf.* section 4.1.1) utilise cette possibilité.

Relation entre les distances

Il est possible d'utiliser l'astuce des fonctions noyaux pour les méthodes basés sur la distance euclidienne en utilisant le fait que:

$$d(\mathbf{x}_i,\mathbf{x}_j)^2 = \langle \mathbf{x}_i - \mathbf{x}_j, \mathbf{x}_i - \mathbf{x}_j \rangle = \langle \mathbf{x}_i,\mathbf{x}_i \rangle + \langle \mathbf{x}_j,\mathbf{x}_j \rangle - 2 \langle \mathbf{x}_i,\mathbf{x}_j \rangle \tag{4.57}$$

En particulier la distance entre deux points dans l'espace de redescription est facile à déterminer:

$$d(\Phi(\mathbf{x}_i),\Phi(\mathbf{x}_j))^2 = K(\mathbf{x}_i,\mathbf{x}_i) + K(\mathbf{x}_j,\mathbf{x}_j) - 2K(\mathbf{x}_i,\mathbf{x}_j) \tag{4.58}$$

Cette possibilité est par exemple utilisée dans le cadre de l'application de la règle du plus proche voisin dans un espace de redescription donné [KARACA04].

6. Bien qu'expérimentalement des fonctions noyaux "impropres" [LIN03B] soient parfois utilisées et que les fonctions de décision produites soient parfaitement exploitables.

Quelques fonctions noyaux

Une liste non exhaustive de quelques fonctions noyaux couramment utilisées avec les SVM:

- le noyau linéaire $K(\mathbf{x}_1,\mathbf{x}_2) = \langle \mathbf{x}_1,\mathbf{x}_2 \rangle$,
- le noyau polinomial $K(\mathbf{x}_1,\mathbf{x}_2) = (\langle \mathbf{x}_1,\mathbf{x}_2 \rangle + 1)^d$
- le noyau gaussien $K(\mathbf{x}_1,\mathbf{x}_2) = \exp(\gamma \left\| \mathbf{x}_1 - \mathbf{x}_2 \right\|^2)$
- le noyau KMOD $K(\mathbf{x}_1,\mathbf{x}_2) = a \left[\exp(\frac{\gamma}{\|\mathbf{x}_1-\mathbf{x}_2\|^2+\sigma^2}) - 1 \right]$ avec $a = \frac{1}{\exp(\frac{\gamma}{\sigma^2}-1)}$.

Les paramètres d, γ et σ sont les paramètres libres de ces différentes fonctions noyaux et ils ont une grande influence sur les capacités en généralisation des SVM.

Il est aussi possible de définir de nouvelles fonctions noyaux en combinant des fonctions noyaux de bases. La somme ou le produit de deux fonctions noyaux valides est également une fonction noyau valide (pour plus de détails sur les possibilités de combinaison des fonctions noyaux se reporter par exemple au chapitre 3 de [CRISTI00]).

4.1.5.2 Cas de duplication d'exemples

Lorsque plusieurs exemples identiques sont présents dans une base d'entraînement, ce qui peut se produire par exemple dans le cadre d'une estimation par *bootstrap*, il est logique de chercher à résoudre un problème de taille plus réduite en tenant compte du nombre de duplications de certains exemples. En effet, les exemples identiques ont tous le même statut de : vecteur de support non limité, vecteur de support limité ou non vecteur de support. Ils ont également tous la même valeur α. Soit m_i le nombre d'exemples z_i identiques dans une base d'entraînement de m exemples avec $\sum_{i=1}^{l} m_i = m$ et l le nombre d'exemples différents. Il est possible de résoudre un problème de taille l, au lieu d'un problème de taille m, en ne conservant qu'un exemplaire des exemples dupliqués dans la base d'entraînement, si le problème d'optimisation est modifié de la façon suivante. L'idée est de considérer que chaque exemple a une constante de régularisation différente $C_i = m_i \cdot C$ qui dépend du nombre d'occurences m_i de l'exemple i dans la base d'entraînement initiale (C conservant sa signification première). Le problème primal (4.25) devient :

$$Q(\mathbf{w}) = \frac{1}{2} \left\| \mathbf{w} \right\|^2 + \frac{1}{p} \sum_{i=1}^{m} C_i \xi_i^p \tag{4.59}$$

Pour le problème dual avec $p = 1$ seules les contraintes sur les multiplicateurs de Lagrange changent $0 \leq \alpha_i \leq C_i$. Dans le cas où la pénalisation est $p = 2$, c'est la fonction noyau qui est modifiée $\tilde{K}(\mathbf{x}_i,\mathbf{x}_j) = K(\mathbf{x}_i,\mathbf{x}_j) + \delta_{i,j}/C_i$ pour le problème dual [ABE05].

4.1.5.3 Cas des bases non equilibrées

Lorsqu'une classe est fortement minoritaire par rapport à une autre, le problème d'optimisation avec pénalisation est généralement à la défaveur de cette classe [ABE05]. A l'extrême on obtient un classificateur qui considère que tous les exemples sont de la classe majoritaire. Globalement cet effet est problématique. Pour réduire ce biais Veropoulos et al

[VEROPO99] proposent d'utiliser deux constantes de régularisation différentes C_+ et C_- pour les exemples de chaque classe[7]. Sans perte de généralité, considérons que la classe négative est celle minoritaire, l'idée est de choisir $C_- > C_+$, pour pénaliser plus fortement les exemples de classe négative mal placés par rapport à la frontière de décision ($\xi > 0$). En considérant les remarques de la section 4.1.5.2, les constantes C_+ et C_- peuvent être interprétées comme le nombre d'occurrences de chaque exemple suivant la classe de ces exemples. En choisissant $C_- > C_+$, cela correspond à augmenter virtuellement le nombre d'exemples de la classe minoritaire. Pour équilibrer les deux classes, une solution consiste à choisir $C_+ = C$ et $C_+ = \frac{m_+/m_-}{C}$ avec m_+ et m_- respectivement le nombre d'exemples de classe positive et négative.

4.1.5.4 Estimation de la probabilité

Il est souvent nécessaire d'avoir, en plus de la classe prédite par le classificateur, une estimation de la probabilité de confiance dans la sélection de cette classe. La sortie d'un SVM ne fournit pas cette estimation, mais la distance à la frontière de décision par rapport à la notion de marge géométrique. Platt [PLATT99B] a proposé une méthode pour produire une estimation de cette probabilité à partir de cette distance à la frontière de décision. Il utilise la fonction sigmoïde suivante :

$$p(\omega = +1|\ \mathbf{x}) = \frac{1}{1 + \exp\left(a_1 \cdot f(\mathbf{x}) + a_2\right)} \tag{4.60}$$

pour transformer la sortie $f(\mathbf{x})$ du SVM en une estimation de la probabilité $p(\omega = +1|\ \mathbf{x})$. Les paramètres a_1 est a_2 sont déterminés en maximisant la vraisemblance à partir d'un jeu de données ($f(\mathbf{x}_i)$,y_i) issu de la base d'apprentissage (pour plus de détails ce reporter à [PLATT99B]). Lin et al [LIN03A] améliorent la méthode de résolution du problème d'optimisation pour la détermination de a_1 est a_2[8]. L'équation (4.60) n'est pas la seule proposée pour réaliser cette estimation, même si elle est la plus utilisée actuellement[9].

Lorsque le problème est multi-classe, les probabilités $p(\omega_i|\ \mathbf{x})$ sont généralement déterminées à partir des probabilités $p_j(\omega = +1|\ \mathbf{x})$ relatives à chaque SVM j intervenant dans le schéma de combinaison (voir le chapitre 6 pour plus de détails), les valeurs $p_j(\omega = +1|\ \mathbf{x})$ étant généralement déterminées à partir de l'expression (4.60). Zadrozny et al [ZADROZ02] proposent une autre méthode qui est basée sur l'utilisation d'une regression *isotonic* [ROBERT88] pour réaliser cette estimation, mais son exploitation semble actuellement anecdotique.

7. Voir également [CALLUT05] pour une autre utilisation de C_+ et C_-.

8. Nous utilisons la méthode de Lin et al [LIN03A] pour l'estimation des valeurs de a_1 est a_2 dans le chapitre 6.

9. Rüping [RÜ04] propose une méthode simple qui ne nécessite pas la résolution d'un problème d'optimisation et cite également d'autres méthodes dans ce même article.

4.2 Entraînement d'ensembles de SVM pour la validation croisée

La section 2.6 décrit les trois procédures les plus utilisées de validation croisée. Toutes partagent la particularité de réaliser un ensemble d'apprentissage à partir de bases d'entraînement qui sont toutes issues d'une base initiale de référence. Chaque base correspond à un sous-ensemble de la base initiale (avec possibilité de doublons dans le cas du *boostrap*). Chaque base d'entraînement partage donc un ensemble de données en commun avec les autres. Du fait qu'il existe une certaine corrélation entre les différentes phases d'entraînements à réaliser, nous allons nous intéresser dans le cas des SVM aux méthodes permettant d'exploiter cette corrélation avec des algorithmes d'entraînement de type SMO. Plus particulièrement, nous nous intéressons à définir les informations exploitables, à partir d'un premier apprentissage sur la base d'entraînement complète, qui permettront de réduire la complexité des prochaines phases d'entraînement à réaliser sur des sous-ensembles de cette base. Les vecteurs de support jouent un rôle particulier pour cette problématique, en particulier à travers les valeurs de la solution optimale $\boldsymbol{\alpha}^*$. Descoste et al [DECOST00] insistent sur cette possibilité et donnent aux méthodes les exploitant le nom de *alpha seeding*.

Un autre point important est la possibilité d'agir sur la valeur de ϵ dans la condition d'arrêt de SMO. Si ϵ est correctement choisie, elle doit permettre de réduire les temps d'entraînement de chacune des phases d'apprentissage, en réduisant le nombre d'itérations pour atteindre le critère de convergence.

Explorons en premier les approches proposées par d'autres chercheurs avant d'expliquer la notre[10].

4.2.1 Approches existantes

4.2.1.1 Techniques d'*alpha seeding*

Avant de rentrer dans les détails de cette technique, introduisons quelques notations. L'objectif est de définir une procédure qui détermine des conditions efficaces d'initialisation de SMO à partir d'un apprentissage prélimaire. Soit le vecteur d'entiers $\mathbf{B}_k$ dont les valeurs $\mathbf{B}_k(i)$ indiquent le nombre d'occurrences de l'exemple z_i dans la $k^{ème}$ base d'entraînement. Notons $\boldsymbol{\alpha}_0$ et $\mathbf{G}_0$, les valeurs retournées par le premier apprentissage avec la base initiale définie par $\mathbf{B}_0$ ($\forall i : \mathbf{B}_0(i) = 1$). Une technique d'*alpha seeding*, dans le cadre qui nous intéresse, est une fonction qui, à partir des valeurs de $\boldsymbol{\alpha}_0$, $\mathbf{G}_0$ et $\mathbf{B}_k$ produit des valeurs d'initialisation efficaces $\tilde{\boldsymbol{\alpha}}_k$ et $\tilde{\mathbf{G}}_k$ pour l'algorithme SMO. Ces valeurs doivent permettre de réduire significativement le nombre d'itérations et les temps nécessaires à la détermination des valeurs $\boldsymbol{\alpha}_k$ et $\mathbf{G}_k$, ces valeurs correspondant à la résolution de SMO($\tilde{\boldsymbol{\alpha}}_k$,$\tilde{\mathbf{G}}_k$,$\epsilon$) pour la base d'apprentissage définie par $\mathbf{B}_k$. Il est important de noter que tout exemple qui a un nombre d'occurrences égal à zéro dans $\mathbf{B}_k$ est considéré comme

10. Dans cette étude nous nous intéressons plus particulièrement à des apprentissages avec des SVM qui utilisent la $p = 1$ comme pénalisation. L'ensemble des expérimentations concernent ce type de pénalisation. Cependant, toutes les méthodes proposées sont applicables avec l'utilisation de $p = 2$, même si l'on peut supposer des adaptations possibles dans ce cas.

retiré de la base d'entraînement. Les valeurs des α_i correspondants sont obligatoirement égales à zéro et l'algorithme SMO utilisé a été modifié pour prendre en compte ces restrictions.
De plus, nous considérons que l'ensemble des valeurs $Q_{i,j}$ sont accessibles[11]. Elles permettent d'accélérer SMO en lui évitant de les calculer, ce qui serait impossible si on était dans le cas d'un premier apprentissage. Ces valeurs permettent de déterminer plus rapidement la valeur de la fonction de décision (4.22) pour les exemples dans la base de test. En effet $K(\mathbf{x},\mathbf{x}_i)$ dans l'expression (4.22) devient pour un exemple z_j ($\forall j : \mathbf{B}_k(j) = 0$) de la base de test égal à $y_j y_i Q_{j,i}$. Dans la suite de ce chapitre nous désignons par "*base* $\mathbf{B}_k$", la base induite par les valeurs du vecteur $\mathbf{B}_k$.

Decoste et al [DECOST00] énumèrent différentes utilisations des techniques d'*alpha seeding*. Toutes ont pour objectif d'accélérer les phases d'apprentissage :

1. L'erreur de *leave-one-out* sur une base d'apprentissage avec des SVM,
2. Des apprentissages avec des valeurs de C croissantes[12],
3. Variations des paramètres libres d'un noyau[13],
4. Heuristiques pour l'initialisation du premier apprentissage.

Dans le cadre de cette étude, seule la méthode proposée pour l'accélération de la procédure de *leave-one-out* est prise en compte[14]. Le principe est simple, si un apprentissage est nécessaire[15] dans le cas de la suppression de z_k de la base initiale, il suffit de rendre α_k égal à zéro dans $\tilde{\boldsymbol{\alpha}}_k$ afin de représenter le fait que z_k est retiré de la base d'apprentissage. Pour réaliser cette action, tout en respectant la contrainte (4.17), Decoste et al proposent d'ajouter une fraction identique à tous les exemples de la même classe qui sont vecteurs de support non limités, et dont la somme correspond à la valeur à soustraire à α_k. La procédure correspondante est détaillée dans l'algorithme 2 et l'application de ($\tilde{\boldsymbol{\alpha}}_k$, $\tilde{\mathbf{G}}_k$)

11. Si le problème est de dimension trop importante, seules celles dans le cache sont accessibles
12. Hastie et al ont montré par la suite l'efficacité de cette possibilité [HASTIE04A].
13. Une étude préliminaire illustre cette possibilité [CRISTI99], même si l'algorithme utilisé est une variante des SVM.
14. bien que les autres possibilités soient également intéressantes (*cf.* section 8).
15. Nous reviendrons plus en détails dans la section 4.2.3 sur les critères qui permettent de ne pas avoir à réaliser toutes les phases d'apprentissage dans le cadre de la procédure *leave-one-out*.

Algorithme 2 Decoste-AS($\boldsymbol{\alpha}$,$\mathbf{G}$,k)

tantque $\alpha_k > 0$ **faire**
  $I_{\text{as}} \leftarrow \{i : y_i = y_k, 0 \leq \alpha_i \leq C\}$)
  $\delta \leftarrow \frac{\alpha_k}{|I_{\text{as}}|}$
  pour $i \in I_{\text{as}}$ **faire**
    $\delta_i \leftarrow \min(\delta, C - \alpha_i)$
    $\alpha_i \leftarrow \alpha_i + \delta_i$
    $\alpha_k \leftarrow \alpha_k - \delta_i$
    MiseAJour($\mathbf{G}$,i,k,δ_i) \\Mise à jour des valeurs de $\mathbf{G}$ à partir des valeurs de i, k, δ_i
  fin pour
fin tantque
Retourner ($\boldsymbol{\alpha}$,$\mathbf{G}$)

$=$ DECOSTE-AS($\boldsymbol{\alpha}$,$\mathbf{G}$,k) fournit les valeurs d'initialisation pour l'entraînement avec la base contenant tous les exemples sauf z_k. L'une des motivations pour n'agir que sur les vecteurs de support non limités (produits par la première phase d'apprentissage) est liée à la nature des implémentations de SMO. Elles ont tendance à modifier les valeurs des α de ces exemples, avant de modifier celles des autres ($\alpha = 0$ ou $\alpha = C$). L'utilisation de techniques de *shrinking* en est l'explication. Le plus gros problème de cette méthode est qu'elle modifie un grand nombre de variables α_i. Le coût de la mise à jour[16] devient alors important dans de nombreux cas ($O\left(|I_{\text{as}}| \cdot m\right)$).

Quelques années plus tard, Lee et al [LEE04], ont proposé une autre méthode. Ils avaient remarqué le problème crucial relatif à la mise à jour de $\mathbf{G}$. Leur méthode a pour objectif de ne modifier que quelques variables de $\boldsymbol{\alpha}$ afin d'obtenir que α_k soit nul, afin de réduire le coût de la mise à jour de $\mathbf{G}$. Ils ont également noté que si I_{as} est vide, la méthode de Decoste et al échoue. Leur second objectif est d'éviter cet écueil. Partant de cette constatation, ils ne se limitent plus aux exemples de la même classe, mais à l'ensemble des exemples qui sont de type vecteur de support non limité. Reste alors à définir une stratégie pour choisir quel α_i doit être modifié pour diminuer α_k. Remarquant que les valeurs de F_i jouent un rôle important dans le critère d'arrêt, ils préconisent de choisir i de façon à ce que les F_i correspondants ne soient pas trop modifiés. Ils proposent comme heuristique d'estimer la magnitude des variations des F_i approximativement[17] par $\delta \cdot t_{i,k}$ avec $t_{i,k} = Q_{i,i} - y_i y_k Q_{i,i}$. L'idée proposée est de trier les α_i candidats en fonction des valeurs $t_{i,k}$ et de choisir en premier celui ayant la plus petite valeur. L'algorithme 3 donne les détails de cette procédure.

Algorithme 3 Lee-AS($\boldsymbol{\alpha}$,$\mathbf{G}$,k)

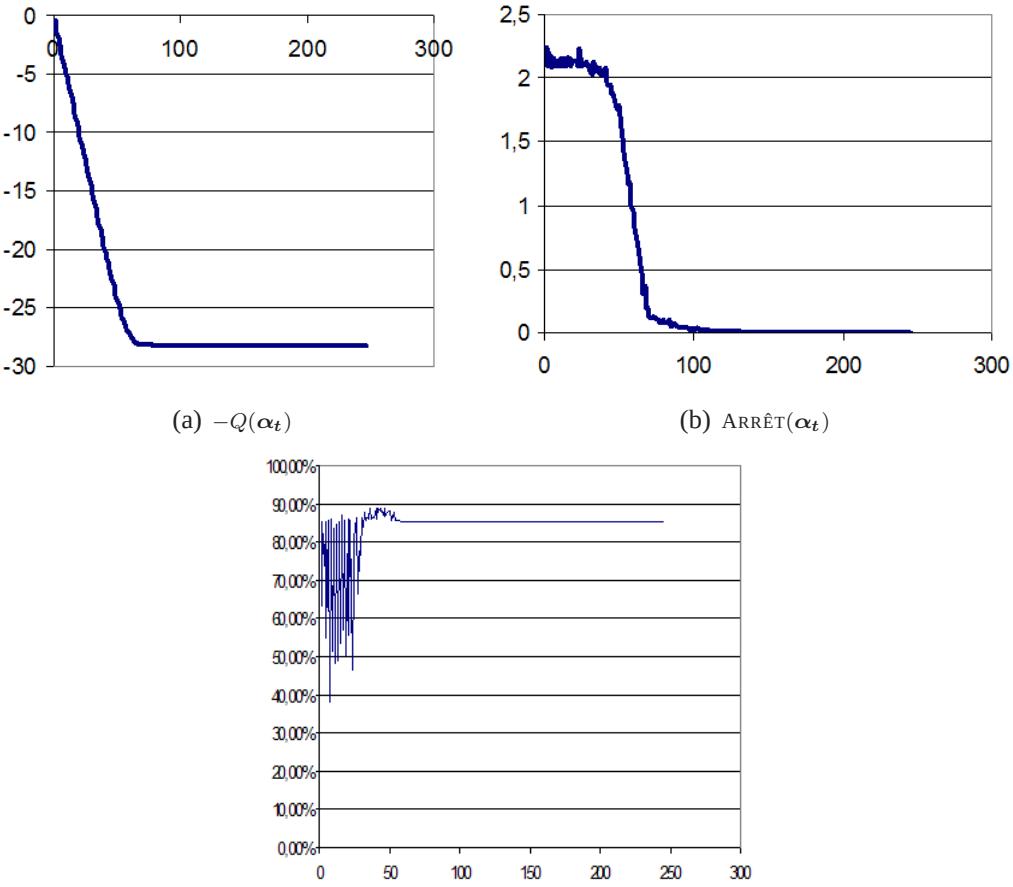
$I_{\text{as}} = \{i : 0 \leq \alpha_i \leq C\}$)
Calculer les $t_{i,k}$ pour tous les $i \in I_{\text{as}}$
Trier I_{as} en fonction de l'ordre croissant induit par les valeurs de $t_{i,k}$
$j \leftarrow 0$
tantque $\alpha_k > 0$ **faire**
  $i \leftarrow I_{\text{as}}(j)$, $s \leftarrow y_i y_k$
  $\delta \leftarrow \min\left(\alpha_k, \max\left(s(C - \alpha_i), -s\alpha_i\right)\right)$
  $\alpha_i \leftarrow \alpha_i + s\delta$
  $\alpha_k \leftarrow \alpha_k - \delta$
  MiseAJour($\mathbf{G}$,i,k,s,δ) \\Mise à jour des valeurs de $\mathbf{G}$ à partir des valeurs de i, k, s, δ
  $j \leftarrow j + 1$
fin tantque
Retourner ($\boldsymbol{\alpha}$,$\mathbf{G}$)

16. Cette mise à jour peut être plus efficace en la réalisant à la fin de l'étape de l'algorithme 2, à partir de la différence de $\boldsymbol{\alpha}$ et $\tilde{\boldsymbol{\alpha}}_k$, mais elle reste toujours de coût important.

17. Normalement, il est nécessaire de vérifier pour les autres F_j avec $j \neq i$ l'effet des modifications de α_i et α_k, mais cela se traduit par une complexité trop importante. Il y a donc une seconde approximation. Approximation dont les effets sont atténués lorsqu'on utilise un noyau gaussien (voir [LEE04] pour plus de détails).

4.2.1.2 Utilisation d'un critère d'arrêt prématuré

L'algorithme SMO peut nécessiter un grand nombre d'itérations pour atteindre le critère d'arrêt, alors que la solution est pourtant proche de l'optimum. Cet effet est souvent dû à l'utilisation d'une valeur trop stricte pour ϵ. C'est généralement le cas, lorsque $\epsilon = 10^{-3}$, valeur qui est choisie par défaut dans la plupart des implémentations de cet algorithme [CHANG01]. Cependant, les dernières itérations ont généralement un effet négligeable sur le comportement de la fonction de décision [BURBID02] (voir par exemple les graphiques de la figure 4.1) et elles correspondent à des modifications mineures des valeurs des α_i. Le fait de choisir une solution d'initialisation proche de l'optimum, grâce à une technique d'*alpha seeding*, peut alors avoir une efficacité réduite. Il est donc important de choisir une valeur plus grande de ϵ pour rendre la technique d'*alpha seeding* plus efficace. A notre connaissance, peu de recherches ont été réalisées dans ce domaine, mise à part deux études qui fournissent des résultats préliminaires. Lee et al [LEE00] réalisent plusieurs expérimentations avec plusieurs bases d'apprentissage couramment utilisées comme *benchmark*. Ils comparent l'estimation de l'erreur de *leave-one-out* (sans utiliser de technique d'*alpha seeding*) avec $\epsilon = 10^{-3}$ et $\epsilon = 10^{-1}$. Les résultats obte-



(a) $-Q(\boldsymbol{\alpha_t})$ (b) ARRÊT$(\boldsymbol{\alpha_t})$

(c) Taux de reconnaissance

FIG. 4.1 – *Evolution de la solution produite par SMO au cours de ces itérations. Les figures montrent l'évolution: (a) de la fonction objectif (au signe près), (b) du critère d'arrêt, (c) du taux de reconnaissance sur une base de validation (base* **Australian***).*

nus illustrent que les temps nécessaires pour cette estimation sont fortement réduits avec $\epsilon = 10^{-1}$, alors que les variations sur l'estimation de l'erreur de *leave-one-out* sont très faibles, en particulier dans le cas de la sélection d'un modèle qui contient la largeur de bande d'un noyau gaussien et la valeur de C. Le modèle sélectionné n'est pas forcément le même, mais la fonction de décision produite a des performances en généralisation similaires.
Lee et al [LEE04] proposent une procédure plus évoluée. Elle consiste à réaliser une première phase d'entraînement avec $\epsilon = 10^{-1}$. La valeur de $f(x_k)$, sortie du SVM, est ensuite calculée. Si $y_k f(x_k) > 0$ ou $y_k f(x_k) < -0{,}25$, l'exemple z_k est considéré comme respectivement bien classé ou mal classé. Dans le cas contraire, l'entraînement est repris avec une valeur de $\epsilon = 10^{-3}$ et la décision d'exemple bien ou mal classé est réalisée à partir de la valeur de $y_k f(x_k)$ pour ce dernier critère d'arrêt. Cette heuristique correspond à un grand nombre d'observations réalisées par Lee et al [LEE04]. Cette procédure est moins rapide que la précédente, mais elle tend à diminuer les variations constatées entre un entraiment avec $\epsilon = 10^{-3}$ et $\epsilon = 10^{-1}$, même si, comme avec la précédente étude, il est vérifié que ces variations sont très faibles.

4.2.2 Notre approche

4.2.2.1 Nouvelle technique d'*alpha seeding*

Notre méthode a pour fondement les travaux de Decoste et al [DECOST00] ainsi que ceux de Lee et al [LEE04]. Elle part des constatations que pour avoir une technique d'*alpha seeding* performante, deux points essentiels doivent être pris en compte:

1. Le nombre de multiplicateurs modifiés, pour rendre une variable α_k nulle dans $\boldsymbol{\alpha}$, doit être le plus faible possible, afin de réduire le coût de la mise à jour de $\mathbf{G}$. Seul la méthode de Lee et al [LEE04] prend en compte ce point de manière efficace.

2. Les modifications nécessaires de $\boldsymbol{\alpha}$, pour produire cette annulation, doivent produire une solution $\tilde{\boldsymbol{\alpha}}$ la plus proche possible de la solution optimale, afin de réduire le nombre d'itérations nécessaires pour atteindre l'optimum avec l'algorithme SMO, lorsque la valeur d'initialisation est $\tilde{\boldsymbol{\alpha}}$. Les deux méthodes précédentes utilisent des heuristiques pour atteindre cet objectif. Dans [DECOST00], l'heuristique correspond à modifier faiblement les valeurs des α par une répartition homogène de leurs variations. Dans [LEE04], la méthode essaie de modifier faiblement les variations des F_i pour rester proche du critère d'arrêt.

Notre approche part de la remarque que plus une solution $\tilde{\boldsymbol{\alpha}}$ a une valeur de $Q(\tilde{\boldsymbol{\alpha}})$ élevée et plus elle est proche de l'optimal. Nous allons maintenant décrire une méthode qui permet, dans la majorité des cas, de ne modifier qu'une seule variable α_i pour permettre à α_k d'être nul. De plus, cette méthode choisit i de façon à ce que $Q(\tilde{\boldsymbol{\alpha}})$ soit maximum sous la contrainte de ne modifier qu'un α_i. Elle permet donc de prendre en compte les deux points essentiels précédemment évoqués. Pour augmenter les chances de pouvoir modifier une seule variable de $\boldsymbol{\alpha}$ afin d'obtenir que la variable α_k soit nulle, nous n'imposons aucune limitation sur le choix de la variable α_i à modifier. Ceci permet d'augmenter les possibilités de produire une valeur élevée de $Q(\tilde{\boldsymbol{\alpha}})$. La seule contre-partie, dans le cas de l'utilisation de techniques de *shrinking*, est une augmentation légère du nombre de

variables à utiliser, si α_i ne fait pas partie des vecteurs de support non limités dans la solution $\boldsymbol{\alpha}_0$. Les principales étapes sont les suivantes :

1. Déterminer l'ensemble des variables α_i qui permettent d'annuler α_k par la seule modification de leurs valeurs.
2. Chercher parmi ces variables celles qui permettent d'avoir la plus grande valeur de $Q(\tilde{\boldsymbol{\alpha}})$, puis réaliser l'annulation de la variable α_k. Terminer en retournant la valeur de $\tilde{\boldsymbol{\alpha}}$.
3. Dans le cas où l'ensemble à l'étape (1) est vide, choisir α_i de façon à ce que sa modification permette un bon compromis entre une diminution importante de α_k et une valeur élevée de $Q(\tilde{\boldsymbol{\alpha}})$. Réaliser la modification, puis recommencer au point 1.

Pour réduire le coût de l'évaluation de $Q(\tilde{\boldsymbol{\alpha}})$, nous utilisons le fait que $Q(\tilde{\boldsymbol{\alpha}})$=$Q(\boldsymbol{\alpha}) + \Delta Q(\boldsymbol{\Delta\alpha})$ avec $\boldsymbol{\Delta\alpha} = \tilde{\boldsymbol{\alpha}} - \boldsymbol{\alpha}$. La valeur de $\Delta Q(\boldsymbol{\Delta\alpha})$ est facile à déterminer lorsque seulement deux variables sont modifiées à partir de la relation (4.48) et le coût de cette évaluation est faible. On peut remarquer que notre méthode d'*alpha seeding* est très proche (l'objectif à atteindre n'est par-contre pas le même) d'une itération de l'algorithme SMO lorsqu'un critère de second ordre est utilisé. En particulier, leurs coûts en temps de calcul sont similaires: seulement une variable α_i est modifiée. Plusieurs expérimentations présentées dans les sections suivantes montrent que, dans une très grande majorité des cas, notre méthode n'a besoin de modifier qu'une seule variable (en plus de α_k). L'algorithme 4 donne les détails de notre méthode d'*alpha seeding*, dont ceux du compromis.

Algorithme 4 Notre-AS(α,G,k,$\mathbf{B}$)

tantque $\alpha_k > 0$ **faire**
  $I_1 \leftarrow \{i | 0 \leq \alpha_i + y_i y_k \alpha_k \leq C, \mathbf{B}(i) > 0\}$
  si $I_1 \neq \emptyset$ **alors**
    $i \leftarrow \arg\max_{i \in I_1} \Delta Q(i,k)$
    $\delta_{\max}(i) = \alpha_k$
  sinon
    $I_\delta \leftarrow \{i | \exists \delta > 0 : 0 \leq \alpha_i + y_i y_k \delta \leq C, \mathbf{B}(i) > 0\}$
    pour $i \in I_\delta$ **faire**
      $\delta_{\max}(i) \leftarrow \max(y_i y_k (C - \alpha_i), -y_i y_k \alpha_i)$
    fin pour
    $\Delta Q_{\min} \leftarrow \min_{i \in I_\delta} \Delta Q(i,k)$
    $i \leftarrow \arg\max_{i \in I_\delta} [\Delta Q(i,k) - \Delta Q_{\min}] \cdot \delta_{\max}(i)$
  fin si
  $\Delta\alpha_k \leftarrow -\delta_{\max}(i)$
  $\Delta\alpha_i \leftarrow y_i y_k \delta_{\max}(i)$
  MiseAJour($\boldsymbol{\alpha}$,$\mathbf{G}$,(k,$\Delta\alpha_k$),(i,$\Delta\alpha_i$)) \\ Mise à jour de $\boldsymbol{\alpha}$, $\mathbf{G}$ à partir des modifications de α_k et α_i
fin tantque
Retourner ($\boldsymbol{\alpha}$,$\mathbf{G}$)

Précisons quelques notations utilisées pour cet algorithme :

- I_1 correspond à l'ensemble des indices des α_i qui permettent d'avoir $\alpha_k = 0$ par la seule modification de l'α_i correspondant,
- I_δ est l'ensemble des indices des α_i qui permettent de diminuer la valeur de α_k d'une quantité supérieure à zéro,
- $\delta_{\max}(i)$ indique de combien la variable α_k peut être diminuée au maximum en ne changeant que la valeur de α_i,
- $\Delta Q(i,k)$ est le résultat de l'expression (4.48) avec $i_1 = i$, $i_2 = k$, $\Delta\alpha_{i_1} = y_i y_k \delta_{\max}(i)$ et $\Delta\alpha_{i_2} = -\delta_{\max}(i)$. Cette valeur correspond à l'accroissement (lorsqu'elle est positive) de la valeur de la fonction objectif,
- $\Delta Q_{\min}$ correspond à l'accroissement le plus faible possible (ou à la décroissance la plus forte si $\Delta Q_{\min}$ peut être négatif) de la fonction objectif à partir de la modification d'une variable α_i dont les indices sont présents dans I_δ.

Quelques remarques :

- Les valeurs des $\Delta Q(i,k)$ peuvent être toutes négatives, dans ce cas c'est la plus faible diminution qui est choisie.
- Comme notre méthode a un coût réduit pour annuler un α_k, il nous a semblé logique de l'étendre au-delà de l'utilisation classique correspondant à l'estimation de l'erreur de *leave-one-out*. Le cas de la validation croisée en k parties et du *boostrap* ont été logiquement pris en compte. En effet, la production d'une solution $\tilde{\boldsymbol{\alpha}}$ d'initialisation correspond, dans l'optique d'une technique d'*alpha seeding*, à choisir une solution $\tilde{\boldsymbol{\alpha}}$ dans laquelle plusieurs variables α_k sont annulées. De plus, cette solution doit être proche de l'optimal pour être intéressante. Une utilisation itérative de l'algorithme 4 est alors envisageable. Les sections 4.2.4 et 4.2.5 donnent les détails de ces deux procédures relativement aux cas à prendre en compte.
- Le compromis choisi correspond à un produit entre l'augmentation de Q par rapport au pire cas représenté par $\Delta Q_{\min}$ et la diminution de α_k. Cela traduit l'objectif de choisir une solution qui diminue suffisamment α_k et produit une solution $\tilde{\boldsymbol{\alpha}}$ proche de l'optimal. Il pourrait être utile de mesurer l'influence de favoriser la proximité à l'optimal ou au contraire la diminution de α_k. Par exemple avec la formulation suivante: $[\Delta Q(i,k) - \Delta Q_{\min}]^\beta \cdot [\delta_{\max}(i)]^{\beta^{-1}}$ avec β un terme à régler. Comme expérimentalement, le compromis a une fréquence d'utilisation très faible, nous avons choisi dans cette étude de ne pas approfondir cette possibilité. Néanmoins dans le cas du *boostrap*, lorsqu'un grand nombre de variables α_k doivent être annulées, favoriser la proximité à l'optimal peut être considéré comme une alternative intéressante.
- $\mathbf{B}$ indique quelles variables α_i peuvent être modifiées. Dans le cas du *leave-one-out*, c'est simplement B_k, c'est-à-dire tous les exemples sauf k. Dans le cas général, la logique voudrait que l'on se limite exclusivement aux exemples présents dans la base d'entraînement. Dans les prochaines sections, nous examinerons sous quelles conditions il est possible de dépasser cette limitation.

4.2.2.2 Nouveau critère d'arrêt prématuré

L'exploitation d'un Critère d'Arrêt Prématuré (CAP), conjointement avec une technique d'*alpha seeding*, permet, dans le cadre de l'utilisation de l'algorithme SMO, de

réduire significativement la durée de chaque nouvelle phase d'apprentissage. La justification du relâchement de la condition d'arrêt a été évoquée précédemment. Notre objectif est de définir un critère d'arrêt dont la valeur de ϵ permet un réglage plus fin que de choisir systématiquement $\epsilon = 10^{-1}$. Notre approche restera cependant une heuristique. Les remarques de la section 4.2.1.2 ont montré qu'il est parfois utile de choisir une valeur de ϵ plus faible pour certains exemples si l'on désire obtenir une estimation de l'erreur de validation croisée plus proche de celle qui correspond à une utilisation systématique de $\epsilon = 10^{-3}$. D'un point de vue plus pratique, un compromis entre faible variation de l'estimation de l'erreur de validation croisée et temps de convergence de SMO doit être trouvé. Il est important de ne pas perdre de vue que ces valeurs d'erreurs ne sont que des estimations de l'erreur de généralisation et non la valeur exacte. De plus, ces estimateurs sont plus ou moins biaisés par le fait que ces techniques utilisent toutes des données issues du même ensemble d'apprentissage initial. L'objectif est alors de définir un critère d'arrêt qui produise une solution $\boldsymbol{\alpha}$ suffisamment proche de l'optimal $\boldsymbol{\alpha}^*$. Sous-entendu, la fonction de décision correspondant à $\boldsymbol{\alpha}$ a un comportement similaire à $\boldsymbol{\alpha}^*$ pour la majorité des exemples d'apprentissage (et par extension les autres). L'heuristique proposé par Lee et al [LEE04] semble indiquer que l'arrêt prématuré de SMO avec $\epsilon = 10^{-1}$ tend à classer incorrectement une proportion d'exemples qui aurait été bien classés avec $\epsilon = 10^{-3}$. Cela fournit une mesure légérement pessimiste de l'erreur de généralisation, mais comme l'estimation de l'erreur de généralisation avec des techniques de validation croisée tend à être plutôt optimiste, cet effet est globalement bénéfique s'il n'est pas trop important.

Nous pensons que le choix d'une valeur efficace de ϵ est fortement lié aux données, au choix d'une fonction noyau (paramètres libres compris) et à la valeur de C. Dans le cadre d'une procédure de validation croisée, seules les données changent (pour les k phases suivantes d'entraînements), les autres éléments étant fixés par l'étude d'un modèle donné. De plus, la variabilité des données d'entraînement est plus réduite dans ce cas, car c'est un sous-ensemble des données utilisées dans la première phase d'entraînement et non un jeu de données totalement différent du premier. Ceci est particulièrement vrai dans le cas de la l'estimation de l'erreur de *leave-one-out*. Les valeurs efficaces de ϵ pour ces différentes phases d'entraînement doivent donc être fortement corrélées. Notre idée est alors de déterminer une valeur efficace de ϵ avec la première phase d'entraînement (celle qui produit $\boldsymbol{\alpha}_0$) et de considérer qu'elle est efficace pour les k autres phases d'entraînement à produire. La valeur de ϵ du critère d'arrêt n'est donc plus fixée, mais variable suivant le modèle utilisé.

Avant de donner les détails de notre méthode, introduisons quelques notations. Notons:

- $\boldsymbol{\alpha}(t)$, la valeur de $\boldsymbol{\alpha}$ à la $t^{\text{ème}}$ itération de l'algorithme SMO.
- $Q(\boldsymbol{\alpha}(t))$ la valeur de la fonction objectif pour la $t^{\text{ème}}$ itération de SMO.
- $\text{ARRÊT}(\boldsymbol{\alpha})$ a la même signification que dans l'expression (4.50).
- $A(t) \equiv \text{ARRÊT}(\boldsymbol{\alpha}(\boldsymbol{t}))$
- $t(\epsilon)$ est la première itération à partir de laquelle le critère d'arrêt est inférieur à ϵ pour toutes les itérations suivantes ($t(\epsilon) \equiv \arg\min\{t : \forall t' > t : \text{ARRÊT}(\boldsymbol{\alpha}(t')) < \epsilon\}$).
- $t_{\max} \equiv t(10^{-3})$ est le nombre maximum d'itérations de SMO lorsque le critère d'arrêt par défaut est utilisé ($\epsilon = 10^{-3}$).
- $q_Q(t) = [Q(\boldsymbol{\alpha}(\infty)) - Q(\boldsymbol{\alpha}(t))] / Q(\boldsymbol{\alpha}(\infty))$ est une mesure de qualité ($q_Q \in [0,1]$) qui caractérise la proximité d'une solution avec l'optimal ($q_Q = 0$ caractérise la proximité maximale). Sans problème numérique (et de temps de calcul), il est lo-

gique de prendre comme référence la solution produite lorque t tend vers l'infini. D'un point de vue plus pratique, l'approximation suivante est prise en compte, à savoir $Q(\boldsymbol{\alpha}(\infty)) = Q(\boldsymbol{\alpha}(t_{\max}))$ en considérant que l'entraînement avec $\epsilon = 10^{-3}$ pour l'algorithme SMO en est une "très bonne" approximation.

– $\tilde{\epsilon}$: critère d'arrêt pour les prochaines phases d'entraînement d'une procédure de validation croisée.

Comme précisé auparavant, nous recherchons une méthode qui permette de choisir une valeur efficace de $\tilde{\epsilon}$. Il suffit pour cela de choisir $\tilde{\epsilon}$ de façon à ce que la solution $\boldsymbol{\alpha}(t(\tilde{\epsilon}))$ soit proche de l'optimal α^*. q_Q mesurant la proximité avec la solution optimale, nous avons choisi $q_Q = 10^{-3}$ comme une mesure efficace de la proximité avec la solution optimale. Notons que les valeurs de ARRÊT($\boldsymbol{\alpha}(t)$) ne sont pas strictement décroissantes en fonction de t, à la différence de celles de $q_Q(t)$. Soit $\tilde{t}$ la première itération pour laquelle $q_Q(\tilde{t}) < 10^{-3}$. En tenant compte de la remarque antérieure, la valeur de $\tilde{\epsilon}$ choisie est la plus grande possible, à la condition que l'arrêt de la procédure SMO ne soit pas antérieur à $\tilde{t}$, afin de s'arrêter le plus vite possible après $\tilde{t}$. Nous utilisons la première phase d'apprentissage pour déterminer la valeur de $\tilde{t}$, puis celle de $\tilde{\epsilon}$ qui corresponde au critère décrit précédemment. Pour cela les valeurs de $Q(\boldsymbol{\alpha}(t))$ et ARRÊT($\boldsymbol{\alpha}(t)$) (i.e., $A(t)$) sont enregistrées pendant cette première phase d'entraînement[18]. L'algorithme 5 réalise la procédure décrite.

Algorithme 5 NOTRE-CRITÈRE-ARRÊT(q_Q,A)

$\tilde{t} \leftarrow \underset{0 \leq t \leq t_{\max}}{\arg\min} \left(\{ t : q_Q(t) < 10^{-3} \} \right)$
$\tilde{\epsilon} = A(0)$
pour $t \in [\, 1, \tilde{t}\,]$ **faire**
  si $A(t) < \tilde{\epsilon}$ **alors**
    $\tilde{\epsilon} = A(t)$
  fin si
fin pour
retourner $\tilde{\epsilon}$

4.2.3 Cas de l'estimation rapide de l'erreur de leave-one-out

4.2.3.1 Spécificités

Dans cette section nous noterons f_k la fonction de décision produite par un SVM qui a été entraîné avec la base $\mathbf{B}_k$. f_0 est la fonction de décision produite par le premier entraînement, avec la base $\mathbf{B}_0$.

Un des avantages de la procédure de *leave-one-out* est qu'une seule variable α_k doit être forcée à zéro. Cet effet positif est réduit par le nombre de phases d'entraînement à réaliser, qui est normalement de m. Dans le cadre des SVM, la procédure de *leave-one-out*

18. Pour ne pas avoir à garder trop de valeurs en mémoire, seul un échantillonnage à intervalles réguliers est réalisé lorsque le nombre d'exemples est important ($\Delta t = \max(\lceil m/100 \rceil, 1)$).

peut utiliser plusieurs règles qui réduisent le nombre effectif d'apprentissages à réaliser. Les règles évoquées sont au nombre de trois [LEE04] :

1. Si $\alpha_k = 0$ alors $y_k \cdot f_k(x_k) \geq 1$,
2. Si $y_k \cdot f_0(x_k) < 0$ alors $y_k \cdot f_k(x_k) < 0$.
3. Si $y_k \cdot f_0(x_k) > 2\alpha_k R^2$ alors $y_k \cdot f_k(x_k) > 0$.

La règle 1 est évidente. Pour les justifications théoriques pour les règles 2 et 3 se reporter à [LEE04] pour plus de détails. Dans la règle 3, la valeur de R correspond au rayon minimum de la sphère englobant les données dans l'espace de redescription. Il est courant de choisir [LEE04]:

$$R^2 = \underset{1 \leq i,j \leq m}{\arg\max} \left(K(\mathbf{x}_i,\mathbf{x}_i) - K(\mathbf{x}_i,\mathbf{x}_j) \right) \tag{4.61}$$

Dans le cas d'un noyau Gaussien, $R = 1$ est une valeur appropriée. Les règles (1) et (3) permettent de prédire que le point sera bien classé par f_k sans avoir à réaliser effectivement cet apprentissage. La règle (2) permet de prédire que le point sera forcément mal classé par f_k également sans phase d'apprentissage. Le nombre d'apprentissages effectifs à réaliser dépend des données et du modèle utilisé [LEE04]. Pour les exemples nécessitant une phase d'entraînement, l'utilisation d'une procédure d'*alpha seeding* permet de réduire le temps total d'entraînements. L'utilisation d'un critère d'arrêt efficace, comme celui de notre méthode (algorithme 5) permet de réduire encore plus les temps d'entrainement.

L'algorithme 6 (*FAS-SMO-LOO-CV: Fast Alpha Seeding SMO with Leave One Out Cross Validation*) donne une description précise de la procédure d'évaluation de l'erreur de *leave-one-out*. Cette procédure utilise les deux techniques précédemment décrites, ainsi que les trois règles qui viennent d'être évoquées. Précisons quelles sont les notations utilisées dans cet algorithme:

- $(v)^m$ désigne un vecteur de taille m dont tous les éléments ont la valeur v.
- Le $\mathbf{B}$ en indice de $\text{SMO}(\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}},\tilde{\epsilon})_B$ désigne que l'algorithme SMO ne peut changer les valeurs des $\tilde{\alpha}_i$ et $\tilde{G}_i$ que pour les élements présents dans la base $\mathbf{B}$, même si l'ensemble des données des autres exemples est encore accessible.
- La procédure correspondant à $\text{MALCLASSÉ}(\boldsymbol{\alpha}_k,\mathbf{G}_k,z_k)$ construit la fonction de décision des SVM à partir de $\boldsymbol{\alpha}_k$ et $\mathbf{G}_k$, puis détermine si l'exemple z_k est mal classé par la fonction de décision correspondante.

4.2.3.2 Résultats expérimentaux

Avant de donner les détails des différentes expérimentations, nous donnons la description de plusieurs notations utilisées dans cette étude:

- T_M^{LOO} et ni_M^{LOO} représentent respectivement le cumul des laps de temps et des nombres d'itérations nécessaires à l'estimation de l'erreur de *leave-one-out* avec une méthode particulière M.
- $G_T^{\text{LOO}} = T_{M_1}^{\text{LOO}} / T_{M_2}^{\text{LOO}}$ correspond au gain en temps réalisé grâce à l'utilisation de la méthode M_2 à la place de la méthode M_1.
- G_{ni}^{LOO} représente, de la même manière que pour G_T^{LOO}, le gain en nombre d'itérations global de SMO lorsque la méthode M_2 est utilisée à la place de la méthode M_1.

La méthode M_2 utilisée est précisée pour chaque estimation de ces gains. La méthode M_1 correspond dans tous les cas à une initialisation classique ($\boldsymbol{\alpha} = 0$ et $\epsilon = 10^{-3}$). Les descriptions des bases **Australian**, **Heart**, **German** et **Adult** sont données en annexe A. Comme la base **Adult** a un nombre important d'exemples (1605) pour une procédure de *leave one out*, l'utilisation directe de la méthode M_1 n'est pas possible dans des temps exploitables. Le temps nécessaire pour cette méthode avec la base **Adult** est par-contre estimable avec une bonne précision à partir de la durée d'entraînement avec la base complète. En effet, la durée d'entraînement (et le nombre d'itérations de SMO) avec m et $m-1$ exemples sont quasiment identiques (pour ce type d'initialisation) lorsque m est suffisamment grand. Les valeurs de G_T^{LOO} et G_{ni}^{LOO} peuvent donc être estimées pour la base **Adult** avec une bonne précision.

Plusieurs expérimentations ont été réalisées avec notre méthode rapide de *leave one out* pour mettre en évidence ses avantages et spécificités. Note méthode est également comparée avec celles précédemment citées.

Notre méthode d'*alpha seeding*

La première expérimentation a pour objectif de mettre en valeur les effets positifs de notre méthode d'*alpha seeding*. Pour cette première expérimentation, la valeur de $\tilde{\epsilon}$ est systématiquement de 10^{-3}. Il n'y a donc pas d'utilisation d'un critère d'arrêt prématuré. Dans cette expérimentation, les règles (2) et (3) de la section précédente ne sont pas utilisées (la règle (1) l'est forcément implicitement, puisque la solution $\boldsymbol{\alpha}$ est par définition déjà optimale dans ce cas). L'intérêt principal de ne pas utiliser ces deux règles est de mesurer les effets positifs de notre méthode pour un maximum de cas où il est nécessaire de réaliser $\alpha_k = 0$. Ceci permet de mettre en évidence la possibilité d'étendre notre mé-

Algorithme 6 FAS-SMO-LOO-CV(Z_m)

$(\boldsymbol{\alpha}_0,\mathbf{G}_0) \leftarrow$ SMO$((0)^m,(-1)^m,10^{-3})_{B_0}$
$(q_Q,A) \leftarrow$ DERNIERENREGISTREMENTSMO()
$\tilde{\epsilon} \leftarrow$ NOTRE-CRITÈRE-ARRÊT (q_Q,A)
$ne \leftarrow 0$ \\ compte le nombre d'erreurs
pour $k \in [1,m]$ **faire**
  $(a,e) \leftarrow$ LES3RÈGLES$(\boldsymbol{\alpha}_{0,k}, z_k)$ \\ a est vrai si une des 3 règles est applicable
  si $a =$ faux **alors**
    $\mathbf{B}_k \leftarrow (1 - \delta_{i,k})_{1 \leq i \leq m}$
    $(\tilde{\boldsymbol{\alpha}}_k,\tilde{\mathbf{G}}_k) \leftarrow$ NOTRE-AS$(\boldsymbol{\alpha}_0,\mathbf{G}_0,k,\mathbf{B}_k)$
    $(\boldsymbol{\alpha}_k,\mathbf{G}_k) \leftarrow$ SMO$(\tilde{\boldsymbol{\alpha}}_k,\tilde{\mathbf{G}}_k,\tilde{\epsilon})_{B_k}$
    $e \leftarrow$ MALCLASSÉ$(\boldsymbol{\alpha}_k,\mathbf{G}_k,z_k)$
  fin si
  $ne \leftarrow ne + e$
fin pour
$e_{\mathrm{LOO}} = ne/m$
retourner e_{LOO}

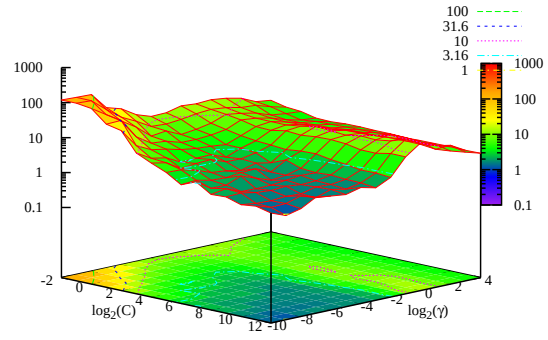
Base	**min**	**max**	**moy**	**écart-type**
Autralian	0,95	119,1	**9,67**	18,97
Heart	1,25	81,38	**8,24**	14,90
German	1,34	78,85	**7,13**	8,32

TAB. 4.1 – *Statistiques sur* $\mathrm{G}_T^{\mathrm{LOO}}$ *(Notre technique d'*alpha seeding *est utilisée sans CAP).*

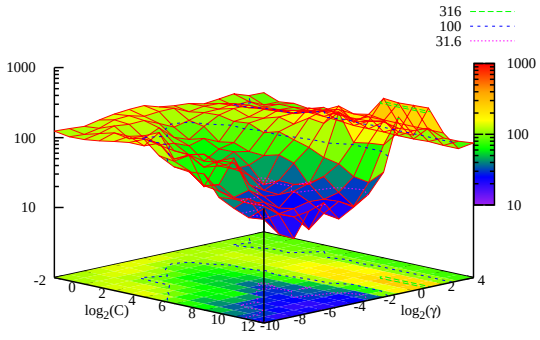
thode au cas de la validation croisée en k parties et au cas du *boostrap*. En effet, pour ces méthodes de validation croisée, il n'est plus possible d'utiliser les 3 règles précédentes. Trois bases ont été utilisées pour cette première expérimentation avec un noyau Gaussien. Pour chaque base, l'estimation de l'erreur de *leave one out* est réalisée pour un ensemble de modèles. Nous avons utilisé la technique de *grid search* qui correspond à tester tous les couples correspondant au produit cartésien d'un ensemble de valeurs pour C et γ (γ représente ici la largeur de bande du noyau Gaussien). La liste des valeurs testées pour C et γ sont les suivantes: $\gamma \in \{2^x | x \in [-10,...,2]\}$, $C \in \{2^x | x \in [-2,...,12]\}$. Le tableau 4.1 donne des statistiques sur les gains en temps d'entraînement pour l'ensemble des couples testés lorsque notre méthode d'*alpha seeding* est utilisée à la place d'une initialisation classique ($\boldsymbol{\alpha} = 0$). Les valeurs minimum (0,95 dans le pire cas) de ce tableau montrent que notre méthode n'est que très rarement plus lente qu'une initialisation classique ($(\alpha) = 0$). Les gains en temps de calcul peuvent être très importants, mais cela ne se produit que dans de rares cas, comme l'illustre la figure 4.2(a). En moyenne le facteur d'accélération est environ de 8. La comparaison des deux figures 4.2(a) et 4.2(c) illustre que ce n'est pas dans les régions avec le plus de vecteurs de support que les gains sont les plus importants, relativisant ainsi l'importance des règles (2) et (3). La lecture de la figure 4.2(d) montre que certains modèles demandent plus d'itérations pour atteindre la convergence. Le cas d'une forte régularisation qui impose de classer correctement la grande majorité des exemples à partir d'un espace d'hypothèses qui a une plus faible expressivité (utilisation d'une Gaussienne avec une décroissance faible) peut expliquer la difficulté rencontrée par SMO pour construire le meilleur séparateur et donc l'augmentation du nombre d'itérations. Cet effet persiste pour les autres phases d'entraînement, même lorsque la solution d'initialisation est proche de l'optimal. Les figures 4.2(d) et 4.2(f) illustrent que ce ne sont pas forcément les couples qui produisent les meilleurs solutions, même si les fonctions de décision correspondantes ne sont pas les moins performantes en généralisation. En comparant les figures 4.2(e) et 4.2(f), on remarque que l'erreur de *leave one out* est fidèle à l'erreur sur la base validation et peut être utilisée pour une sélection efficace d'un modèle.

Utilisation de notre critère d'arrêt prématuré

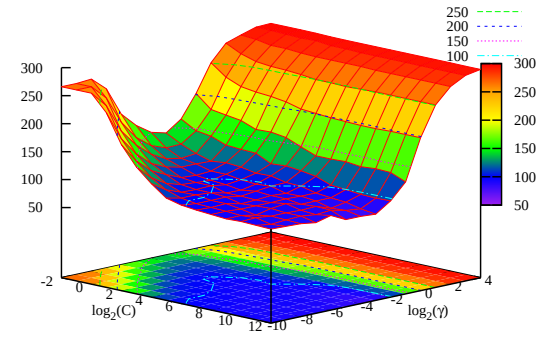
Les résultats précédents ont montré que la technique d'*alpha seeding* accélère l'estimation de l'erreur de *leave one out*. Pour autant, le facteur d'accélération est limité par la lente convergence de SMO durant les dernières itérations. La seconde expérimentation permet d'illustrer les propriétés de notre méthode lorsque notre critère d'arrêt prématuré est utilisé conjointement à notre technique d'*alpha seeding*. L'objectif est d'améliorer le facteur d'accélération. Le tableau 4.2 donne différentes statistiques sur les valeurs du CAP produit par notre méthode avec les mêmes bases que celles utilisées précédemment. L'en-



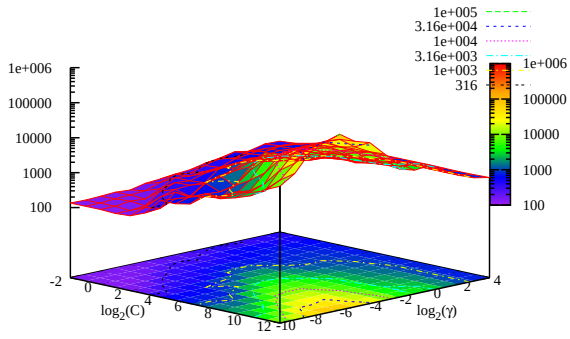
(a) $\mathrm{G}_T^{\mathrm{LOO}}$ sans CAP



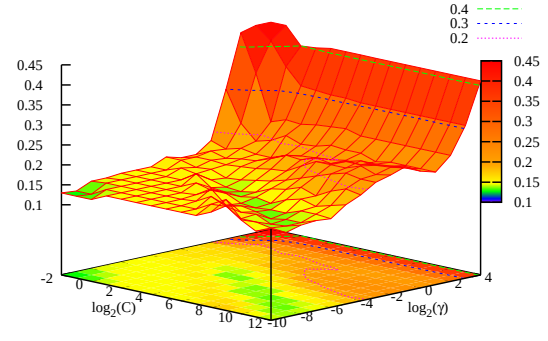
(b) $\mathrm{G}_T^{\mathrm{LOO}}$ avec CAP



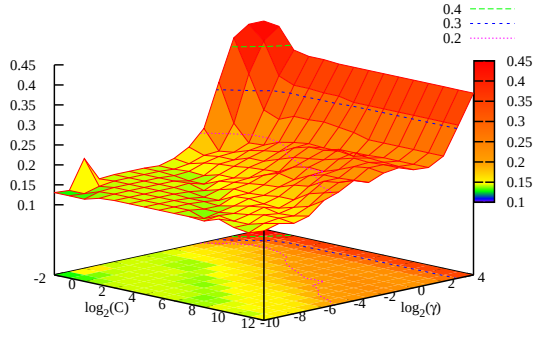
(c) n_{VS}



(d) n_{it}



(e) e_{LOO}



(f) e_{v}

FIG. 4.2 – *Graphiques représentant différents résultats pour l'ensemble des modèles* (C,γ) *testés avec la base* ***Australian****. (a et b) Surface des gains* $\mathrm{G}_T^{\mathrm{LOO}}$ *sans (a) et avec (b) Critère d'arrêt prématuré. (c) Nombre de vecteurs de support. (d) Nombre d'itérations de l'algorithme SMO avec la première phase d'entraînement. (e) Erreur estimée de* leave one out. *(f) Erreur mesurée sur la base de validation.*

Base	**min**	**max**	**moy**	**écart-type**
Australian	0,057	1,665	**0,164**	0,186
Heart	0,062	1,785	**0,178**	0,216
Geman	0,056	1,986	**0,149**	0,218
Adult	0,058	1,854	**0,287**	0,300

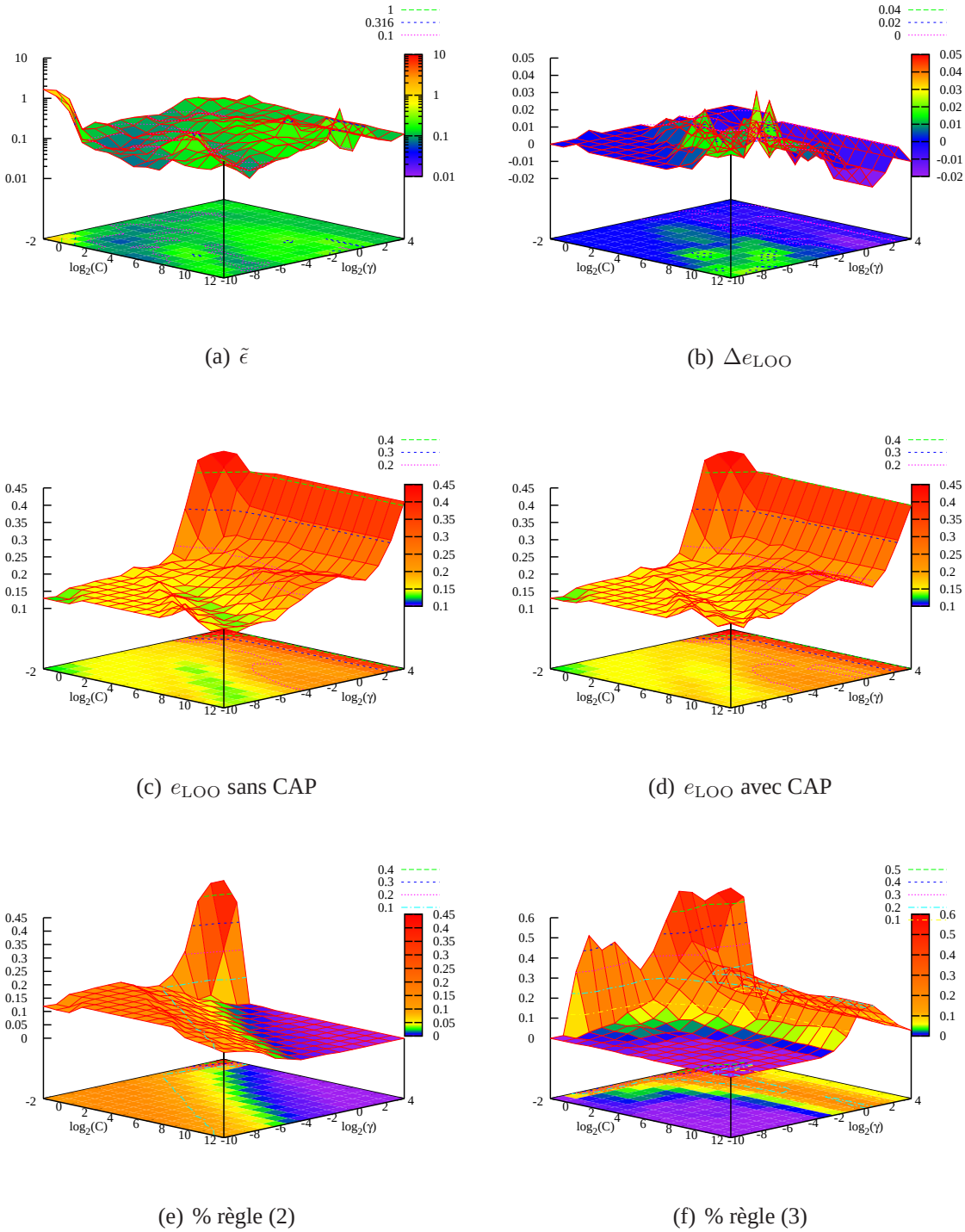
TAB. 4.2 – *Statistiques sur les valeurs sélectionnées pour $\tilde{\epsilon}$ avec notre critère d'arrêt prématuré pour les différents modèles (C,γ) testés.*

semble des couples de modèles (C,γ) utilisés sont également les mêmes que précédemment. La figure 4.3(a) illustre les variations de $\tilde{\epsilon}$ pour différents modèles avec la base **Australian**. On peut observer que l'heuristique de Lee et al (*i.e.* $\tilde{\epsilon} = 0{,}1$) est assez proche de la valeur moyenne de notre méthode, même si pour certains modèles il y a d'importantes différences. On peut également remarquer que pour la base **Adult**, l'écart avec cette heuristique est plus important.

Dans l'expérimentation précédente nous avions réalisé une procédure de validation croisée sans critère d'arrêt prématuré. Nous avons réutilisé ces résultats pour comparer les variations sur l'erreur de *leave one out* avec et sans notre critère d'arrêt prématuré. Le tableau 4.2 donne des statistiques sur ces écarts et la figure 4.3(b) illustre dans quelle région ces écarts sont importants dans le cas de l'utilisation de la base **Australian**. Globalement ces écarts sont assez faibles. De plus, les écarts les plus importants ne sont pas localisés dans les régions avec les meilleurs taux de reconnaissance, mais dans celles où la convergence de SMO est la plus lente (*cf.* figure 4.2(d)). La sélection d'un modèle efficace à partir de l'erreur de *leave one out* est alors faiblement perturbée par l'utilisation de ce critère d'arrêt prématuré (*cf.* figures 4.3(c) et 4.3(d)). Par contre les gains en temps de calcul pour obtenir ces estimations sont grandement améliorés, comme l'illustrent les statistiques dans le tableau 4.4. Les figures 4.2(a) et 4.2(b) permettent de comparer l'apport sur les gains pour l'ensemble des modèles testés avec la base Australian. On remarque que le gain est plus faible dans les régions où la convergence de SMO est plus lente.

L'apport sur l'augmentation de l'accélération est encore plus important avec la base **Adult**, ce qui semble indiquer que l'accélération est d'autant plus importante que la taille de la base d'apprentissage est grande. La figure 4.4(a) illustre que pour la majorité des modèles la solution $\tilde{\boldsymbol{\alpha}}$ est proche de la solution optimale $\boldsymbol{\alpha}^*$. Les figures 4.4(b), 4.4(c) et 4.4(d) illustrent comment l'effet du sur-apprentissage avec un mauvais modèle est mis en évidence avec l'estimation de l'erreur de *leave one out*. Cela illustre par la même occasion la capacité à "sur-apprendre" de l'algorithme des SVM avec un nombre significatif d'exemples lorsque le modèle n'est pas bien choisi. Les figures 4.4(c) et 4.4(d) illustrent que la corrélation entre l'erreur *leave one out* et l'erreur de validation reste importante. La sélection d'un modèle optimal à partir de l'erreur estimée de *leave one out* ($\min(e_{\mathrm{LOO}}) = 17{,}69\%$) produit une fonction de décision qui a une erreur sur la base de validation de $16{,}37\%$. Pour information, la sélection du meilleur modèle directement à partir de la base de validation a une erreur mesurée sur cette même base de $15{,}68\%$.

Lorsque les règles (2) et (3) (*cf.* section 4.2.3.1) sont utilisées, il y a une baisse d'environ 20% du nombre de phases d'entraînement à réaliser. Le tableau 4.5 donne des informations statistiques sur l'application de ces 3 règles pour les bases **Australian** et **Adult** et les figures 4.3(e) et 4.3(f) illustrent les régions où ces règles (2) et (3) ont un taux



(a) $\tilde{\epsilon}$

(b) Δe_{LOO}

(c) e_{LOO} sans CAP

(d) e_{LOO} avec CAP

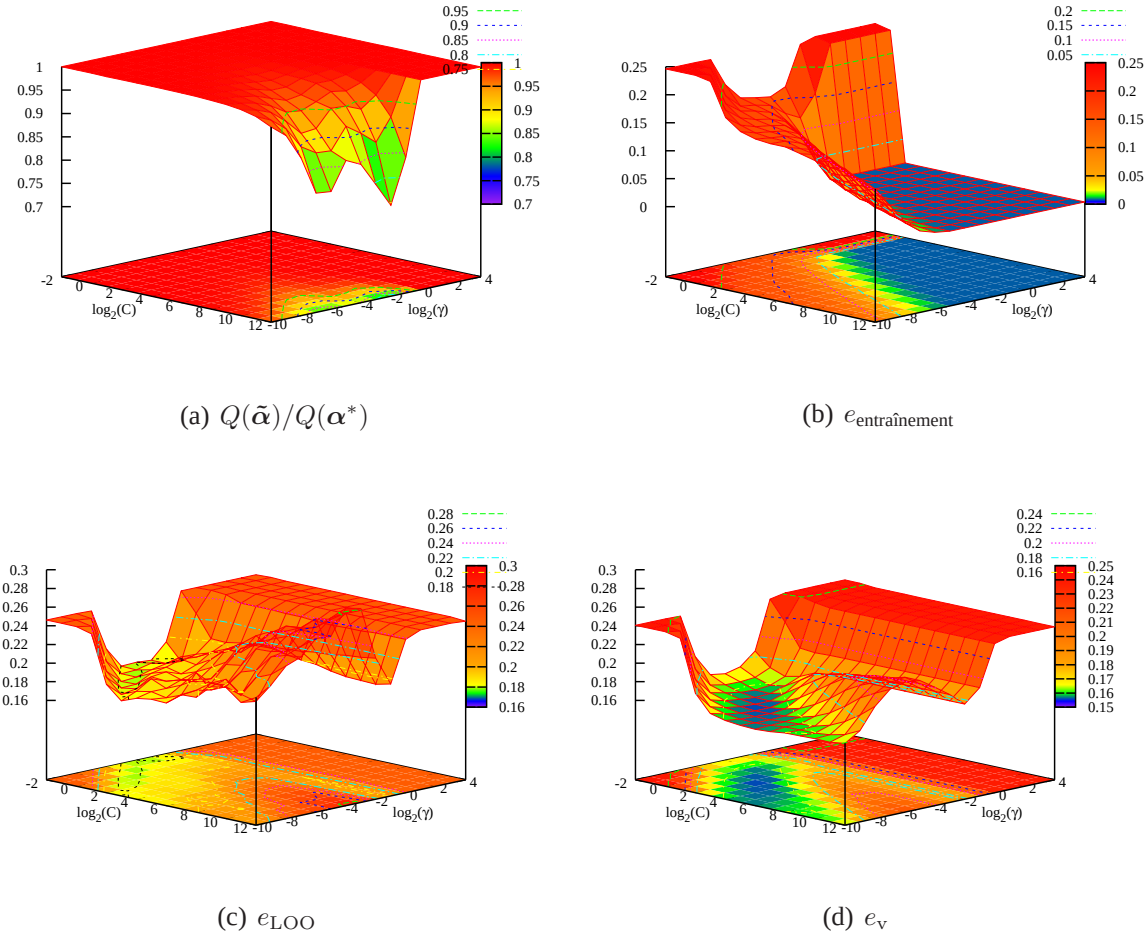
(e) % règle (2)

(f) % règle (3)

FIG. 4.3 – *Graphiques représentant différents résultats pour l'ensemble des modèles* (C,γ) *testés avec la base* ***Australian****. (a) Valeurs de* $\tilde{\epsilon}$ *avec le CAP. (b) Ecarts sur l'estimation de l'erreur de LOO entre sans CAP et avec CAP . (c et d) Erreur estimée de* leave one out *sans CAP (même figure que 4.2(e)) et avec CAP. (e et f) Pourcentage d'exemples pour lesquels la règle (2) ou (3) est applicable.*

Base	min	max	moy	écart-type
Australian	-2,0%	+3,67%	**+0,16%**	±0,86%
Heart	-1,67%	+5,0%	**+0,59%**	±1,05%
German	-0,5%	+7,5%	**+1,23%**	±1,85%

TAB. 4.3 – *Statistiques sur les variations de l'erreur de LOO (Δe_{LOO}) entre l'utilisation de notre CAP et un CAP fixe ($\epsilon_{\mathrm{LOO}} = 10^{-3}$).*



(a) $Q(\tilde{\boldsymbol{\alpha}})/Q(\boldsymbol{\alpha}^*)$ (b) $e_{\text{entraînement}}$

(c) e_{LOO} (d) e_{v}

FIG. 4.4 – *Graphiques représentant différents résultats pour l'ensemble des modèles (C,γ) testés avec la base **Adult**. (a) Rapport $Q(\tilde{\boldsymbol{\alpha}})/Q(\boldsymbol{\alpha}^*)$. (b, c et d) erreurs évaluées directement sur la base d'entrainement (b), avec la procédure de* leave one out *(c) et avec la base de validation (d).*

Base	min	max	moy	écart-type
Australian	12,72	364,44	**99,01**	±60,79
Heart	25,08	379,59	**52,10**	±63,75
Geman	17,96	224,43	**102,43**	±50,70
Adult	82,30	26580	**1587**	±4159

TAB. 4.4 – *Statistiques sur G_T^{LOO} lorsque notre méthode d'*alpha seeding *et d'arrêt prématuré de SMO sont combinées.*

Base		**min**	**max**	**moy**	**écart-type**
Australian	règle 1	0,33%	73,3%	**42,25%**	±25,21%
Adult	règle 1	1,43%	95,95%	**35,94%**	±25,22%
Australian	règle 2	0,00%	44,33%	**7,14%**	±7,36%
Adult	règle 2	0,62%	24,61%	**7,53%**	±8,03%
Australian	règle 3	0,00%	59,33%	**8,92%**	±12,71%
Adult	règle 3	0,00%	73,95%	**25,34%**	±30,70%
Australian	G_T^{LOO}	21,79%	266,3	**86,20**	±46,52
Adult	G_T^{LOO}	1,28%	5294	**828,3**	±822,1

TAB. 4.5 – *Statistiques sur l'utilisation des 3 règles pour les deux bases **Australian** et **Adult** pour l'ensemble des modèles (C,γ) testés. Les 6 premières lignes relatives aux règles 1 à 3 donnent le pourcentage d'exemples concerné par l'application de ces règles. Les 2 dernières lignes donnent les valeurs de G_T^{LOO} lorsque les 3 règles sont exploitées.*

d'application important. On remarque que c'est la règle (1) qui est majoritairement applicable. Malgré tout, la réduction du nombre de phases d'entraînement est significative et l'utilisation de ces règles doit être réalisée en pratique, car elle diminue encore plus les temps d'estimation de l'erreur de LOO. Globalement les facteurs d'accélération sans et avec l'utilisation de ces deux règles restent importants (voir les deux dernières lignes du tableau 4.5), même si ces facteurs diminuent car le nombre de cas d'entraînements de durée nulle augmentent, alors que la durée de la première phase d'entraînement avec la base complète reste la même.

Comparaison entre notre CAP et celui de Lee et al

Nous avons comparé l'utilisation de notre critère d'arrêt avec celui proposé par Lee et al [LEE00] ($\tilde{\epsilon} = 0{,}1$). Les couples de paramètres et les bases utilisées sont les mêmes que pour les expérimentations précédentes (les règles (2) et (3) n'ont pas été utilisées). Le tableau 4.6 permet de comparer le gain en nombre d'itérations (colonnes G_{ni}^{LOO}) et les variations sur l'erreur LOO estimée (colonnes Δe_{LOO}) pour ces deux CAP par rapport à une initialisation classique de SMO $\tilde{\epsilon} = 10^{-3}$). Pour les deux CAP, c'est notre méthode d'*alpha seeding* qui est utilisée .

Les résultats dans le tableau 4.6 illustrent que les écarts sur l'estimation de l'erreur de LOO entre un des CAP et le critère d'arrêt par défaut ($\tilde{\epsilon} = 10^{-3}$) sont faibles et du même ordre de grandeur avec ces deux CAP, mais notre critère permet de réduire plus significativement le nombre d'itérations de SMO. Globalement notre CAP est plus performant en moyenne.

Comparaison des différentes techniques d'*alpha seeding*

Nous avons également comparé les trois techniques d'*alpha seeding* présentées précédemment. Les couples de paramètres et les bases utilisées sont les mêmes que pour les expérimentations précédentes (les règles (2) et (3) n'ont pas été utilisées). Pour ces trois techniques nous avons utilisé notre CAP.

	Notre CAP		CAP de Lee et al	
Bases	G_{ni}^{LOO}	Δe_{LOO}	G_{ni}^{LOO}	Δe_{LOO}
Australian	100	$+1{,}6 \pm 0{,}9\%$	76	$+1{,}8 \pm 0{,}6\%$
Heart	49	$+0{,}6 \pm 1{,}0\%$	39	$+0{,}5 \pm 0{,}9\%$
German	109	$+1{,}2 \pm 1{,}8\%$	102	$+0{,}8 \pm 1{,}3\%$
Adult	1787	-	467	-

TAB. 4.6 – *Comparaison entre notre CAP et celui de Lee et al.*

	n_α			G_{ni}^{LOO}			$\mathrm{G}_{\mathrm{T}}^{\mathrm{LOO}}$		
Bases	AS_1	AS_2	AS_3	AS_1	AS_2	AS_3	AS_1	AS_2	AS_3
Australian	1,0	2,9	110	100	89	110	99	82	41
Heart	1,1	2,4	82	49	51	112	52	51	28
German	1,1	2,7	223	109	109	341	102	104	38
Adult	1,0	5,1	789	1787	529	699	1587	460	115

TAB. 4.7 – *Comparaison entre les trois techniques d'*alpha seeding*. Les notations* AS_1, AS_2 *et* AS_3 *désignent respectivement notre technique d'*alpha seeding*, celle de Lee et al et celle de DeCoste et al. Les trois premières colonnes, désignées par* n_α*, donnent le nombre moyen (pour l'ensemble des couples* (C,γ) *testés) de multiplicateurs* α *modifiés pour annuler un multiplicateur* α_k *avec ces 3 techniques. Les 2 autres groupements de colonnes donnent les valeurs moyennes de* G_{ni}^{LOO} *et* $\mathrm{G}_{\mathrm{LOO}}^{\mathrm{T}}$ *avec ces 3 techniques.*

Les résultats dans le tableau 4.7 illustrent que c'est notre technique qui modifie le moins de multiplicateurs α et qui a donc un coût de mise à jour des valeurs de $\mathbf{G}$ le plus faible. Le nombre de multiplicateurs α modifiés par notre méthode est de 1 dans la très grande majorité des cas. Il est donc très difficile d'avoir un coût de mise à jour des valeurs de $\mathbf{G}$ plus faible. Cet effet est également important pour étendre notre technique à d'autres méthodes de validation croisée (voir les sections suivantes). La technique de Lee et al est fortement pénalisée par ses coûts de mise à jour, car elle modifie un nombre important de multiplicateurs α. Ceci est particulièrement marqué lorsqu'on compare les valeurs dans les colonnes $\mathrm{G}_{\mathrm{LOO}}^{\mathrm{T}}$, alors que pour les trois premières bases les valeurs d'initialisation $\tilde{\boldsymbol{\alpha}}$ sont plus performantes avec cette technique. Les résultats avec la base **Adult** indiquent également que notre technique devient plus performante que les autres techniques lorsque le nombre d'exemples est important.

Lorsque les règles (2) et (3) sont utilisées, les valeurs $\mathrm{G}_{\mathrm{T}}^{\mathrm{LOO}}$ diminuent environ dans les mêmes proportions (828, 299 et 88 respectivement pour les techniques AS_1, AS_2 et AS_3 avec la base **Adult**). L'utilisation de ces règles spécifiques à la procédure de LOO ne change donc pas les meilleures performances de notre technique d'*alpha seeding*.

Synthèse de ces expérimentations

En résumé de ces différents résultats, il apparaît que notre méthode (AS + CAP) permet d'estimer avec une bonne précision l'erreur de LOO et de réduire de façon très significative les temps nécessaires à cette estimation. Ces résultats montrent également que notre méthode est globalement plus performante que celles qui ont été proposées précédemment.

4.2.4 Cas de l'estimation rapide de l'erreur de la k validation croisée

4.2.4.1 Spécificités

L'objectif est de pouvoir réaliser des sélections de modèles avec une procédure de validation croisée en k parties avec des valeurs de k relativement élevées sans que les temps de ces multiples entraînements soient trop pénalisants. Plus la valeur de k est grande est plus le nombre de données en commun avec la base d'entrainement initiale $\mathbf{B}_0$ est grande. De plus, les solutions produites par les SVM sont généralement parcimonieuses. La plupart des valeurs d'une solution $\boldsymbol{\alpha}_0$ sont donc nulles. Dans ces conditions, pour produire une solution $\tilde{\boldsymbol{\alpha}}_k$ d'initialisation valide avec la base $\mathbf{B}_k$, le nombre de vecteurs de support à annuler est faible en moyenne. La méthode que nous proposons pour produire une solution $\tilde{\boldsymbol{\alpha}}_k$ efficace pour un apprentissage $\mathbf{B}_k$ prend en compte plusieurs particularités. Premièrement le nombre de variables α à annuler devient généralement plus important avec des valeurs faibles de C. La raison est que le nombre d'exemples dans la marge diminue avec l'augmentation de la pénalité appliquée aux exemples concernés. Lee et al [LEE04] ont observé cet effet expérimentalement. De plus, l'augmentation est plus marquée pour les vecteurs de support limités, indépendamment de la classe des exemples. Notre méthode tient compte de cette particularité. Pour cela, elle détermine les indices des vecteurs de support limités qui devront être annulés pour les deux classes considérées. Notons $I_\oplus$ et $I_\ominus$, ces deux ensembles. Leurs définitions sont les suivantes:

$$I_\oplus = \{j : \mathbf{B}_k(j) = 0, \tilde{\alpha}_j = C, y_j = +1\} \tag{4.62}$$

$$I_\ominus = \{j : \mathbf{B}_k(j) = 0, \tilde{\alpha}_j = C, y_j = -1\} \tag{4.63}$$

A cause du fait que les exemples dans ces deux ensembles sont de classes opposées, l'annulation des deux variables dans $\tilde{\boldsymbol{\alpha}}$, pour les exemples qui ne sont pas dans $\mathbf{B}_k$, est facile à réaliser. Pour cela, il suffit de réaliser des modifications par paires (j_1,j_2) et de prendre j_1 et j_2 respectivement dans $I_\oplus$ et $I_\ominus$. La contrainte (4.17) est alors forcément respectée, puisque les variations sont identiques ($\Delta\alpha = -C$) avec des élements de classe opposée ($y_{j_1} = -y_{j_2}$). Ce processus peut être repété tant que l'un de ces deux ensembles n'est pas vide. Cette méthode constitue une heuristique rapide pour annuler un grand nombre de variables α liées aux exemples non présents dans $\mathbf{B}_k$. La mise à jour des variables $\mathbf{G}_i$ est facile à réaliser à partir de l'expression suivante:

$$\Delta G_i = C y_i (K(\mathbf{x}_{j_1},\mathbf{x}_i) - K(\mathbf{x}_{j_2},\mathbf{x}_i)) \tag{4.64}$$

et la variation de la fonction objectif est alors égale à:

$$\Delta Q = \frac{C^2}{2}(2Q_{j_1,j_2} + Q_{j_1,j_1} + Q_{j_2,j_2}) \tag{4.65}$$

Notons que seule la mise à jour des G_i correspondant à des exemples Z_i présents dans la base B_k est nécessaire à l'initialisation de SMO. Bien que les variations de la fonction objectif soient faciles à déterminer dans ces conditions (expression (4.65)), notre heuristique n'en tient pas compte pour choisir un couple "idéal" à chaque fois. Premièrement car il est nécessaire d'examiner $|I_\oplus| \times |I_\ominus|$ couples à chaque fois, ce qui augmenterait sérieusement la complexité de l'heuristique. Deuxièmement, parce que cela n'apporte rien lorsque les deux ensembles ont la même taille. On peut alors supposer que le gain serait

faible, en réalisant cette recherche exhaustive, pour des ensembles qui ont des nombres d'indices très proches. Dans la majorité des cas, c'est ce qui est observé. Pour toutes ces raisons, j_1 et j_2 sont sélectionnés en fonction de leurs indices croissants dans $I_\oplus$ et $I_\ominus$.

Pour le reste des variables α_j à annuler, nous utilisons notre méthode d'*alpha seeding* pour chacune de ces variables. Les variables α_i qui peuvent être modifiées pour permettre d'annuler une variable α_j donnée sont celles qui sont dans la base d'entraînement ($\forall i \in \{r|\mathbf{B}_k(r) > 0\}$). Il est possible de modifier également celles qui ne sont pas dans la base d'entraînement, à condition qu'elles soient de classe opposée à l'exemple j considéré ($\forall i \in \{r|\mathbf{B}_k(r) = 0 \wedge (y_r \neq y_j)\}$), car dans ce cas $\Delta\alpha_i$ est forcément négatif. Il n'y alors aucun risque de créer une nouvelle variable à annuler par la suite et donc d'avoir une procédure dont l'arrêt n'est pas garanti. Les chances de trouver une variable α_i qui permette d'annuler la variable α_j par sa seule modification sont augmentées (le tri par ordre croissant des valeurs des α_j à annuler est réalisé pour accroître cette possibilité). Un autre avantage d'avoir un nombre plus important de variables α_i potentiellement modifiables est d'augmenter les possibilités pour la procédure NOTRE-AS de sélectionner des couples (i,j) qui permettent de réaliser des variations ΔQ importantes. L'algorithme 7 (*FAS-SMO-kCV : Fast Alpha Seeding SMO k Cross Validation*) donne les détails de la méthode d'évaluation rapide de la procédure de validation croisée en k parties. Détaillons certaines notations utilisées dans cet algorithme:

- ANNULERCOUPLES correspond à la procédure itérative décrite précédemment pour l'annulation de couple de variables. n_a est le nombre de couples (j_1,j_2) à annuler à partir des ensembles $I_\oplus$ et $I_\ominus$. A chaque annulation d'un couple, $\tilde{\boldsymbol{\alpha}}$ et $\tilde{\mathbf{G}}$ sont mis à jour. Pour $\tilde{\mathbf{G}}$, c'est l'expression (4.64) qui est utilisée.
- $\mathbf{B}(I)$ référence les α_i que la fonction NOTRE-AS peut modifier et dont les indices sont présents dans I ($B(i) = 1$ si $i \in I$ et 0 sinon).
- $\bar{\mathbf{B}}$ est le complémentaire de $\mathbf{B}$ et il est définis comme : $\bar{\mathbf{B}}(r) = 1$ si $\mathbf{B}(r) = 0$ et 1 sinon.
- $Z_m^{\mathbf{B}}$ est une base d'exemples avec $B(i)$ occurrences de l'exemple $z_i \in Z_m$.
- NOMBREMALCLASSÉS est une fonction qui compte le nombre d'exemples mal classés dans la base de test $Z_m^{\mathbf{B}}$ à partir de la fonction de décision déductible des valeurs de $\boldsymbol{\alpha}$ et $\mathbf{G}$.

4.2.4.2 Résultats expérimentaux

Nous avons réalisé des expérimentations avec les mêmes bases de données que celles utilisées à la section 4.2.3.2 avec l'algorithme 7 pour différentes valeurs de k. Les mêmes couples d'hyper-paramètres (C,γ) que pour l'estimation de l'erreur LOO ont été utilisés. L'objectif est de mesurer l'accélération produite par notre algorithme pour l'estimation de l'erreur de validation croisée en k parties par rapport à une procédure classique. Les mêmes k parties de la base initiale sont utilisées pour l'ensemble des couples d'hyper-paramètres (C,γ) pour une valeur de k donnée. Soit t_0 le temps nécessaire à l'entraînement d'un SVM avec la base Z_m complète et t_k le temps nécessaire à la réalisation des k itérations de l'algorithme 7. Nous notons $r_k = t_k/t_0$ le rapport de ces deux valeurs. Dans

Algorithme 7 FAS-SMO-kCV(Z_m,k)

$(\boldsymbol{\alpha}_0,\mathbf{G}_0) \leftarrow \text{SMO}((0)^m,(-1)^m,10^{-3})_{\mathbf{B}_0}$
$(q_Q,A) \leftarrow$ DERNIERENREGISTREMENTSMO()
$\tilde{\epsilon} \leftarrow$ NOTRE-CRITÈRE-ARRÊT(q_Q,A)
pour $i \in [1,k]$ **faire**
  $(\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}}) \leftarrow (\boldsymbol{\alpha}_0,\mathbf{G}_0)$
  $I_\oplus \leftarrow \{j : \mathbf{B}_i(j) = 0,\tilde{\alpha}_j = C,y_j = +1\}$
  $I_\ominus \leftarrow \{j : \mathbf{B}_i(j) = 0,\tilde{\alpha}_j = C,y_j = -1\}$
  $n_a \leftarrow \min(|I_\oplus|,|I_\ominus|)$
  $(\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}}) \leftarrow$ ANNULERCOUPLES$(I_\oplus,I_\ominus,n_a,\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}})$
  $J \leftarrow \{r : \mathbf{B}_i(r) = 0,\ \tilde{\alpha}_r > 0\}$
  TRIER$(J,\tilde{\boldsymbol{\alpha}})$ \\ Tri par ordre croissant des valeurs $\tilde{\alpha}_{j\in J}$
  pour $j \in J$ **faire**
    $I \leftarrow \{r|(\mathbf{B}_i(r) > 0) \vee (y_r \neq y_j \wedge \mathbf{B}_i(r) = 0)\}$
    $(\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}}) \leftarrow$ NOTRE-AS$(\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}},\ j,\ \mathbf{B}(I))$
    $(\boldsymbol{\alpha},\mathbf{G}) \leftarrow \text{SMO}(\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}},\tilde{\epsilon})_{\mathbf{B}_i}$
  fin pour
  $e_i \leftarrow$ NOMBREMALCLASSÉS$(\boldsymbol{\alpha},\ \mathbf{G},\ Z_m^{\bar{\mathbf{B}}_i})/|Z_m^{\bar{B}_i}|$
fin pour
$e_{\text{kCV}} = 1/k \sum e_i$
retourner e_{kCV}

le cadre d'une procédure de validation croisée en k parties classique r_k est environ[19] égal à k. Si r_k est beaucoup plus faible que k lorsque que la valeur de k augmente, alors l'effet d'accélération est significatif. Pour chaque valeur de k testée, la valeur moyenne e_{kCV} des e_i, ainsi que l'écart-type de ces valeurs sont mesurés. La mesure de l'erreur e_{v} sur la base de validation l'est également pour l'ensemble des couples (C,γ). Les tableaux 4.8 et 4.9 donnent les résultats obtenus pour certains des couples (C,γ) testés pour les deux bases **Australian** et **Adult**. Les couples de (C,γ) ont été choisies pour mettre en évidence les écarts importants pouvant exister sur les capacités de généralisation en fonction des valeurs de ces hyper-paramètres.

Les valeurs de r_k dans les tableaux 4.8 et 4.9 illustrent qu'il y a une accélération importante de l'estimation de l'erreur de validation croisée en k parties, notamment pour les valeurs importantes de k. Le temps nécessaire à une estimation avec $k = 100$ est généralement inférieur à 3 fois le temps nécessaire à une estimation avec $k = 5$ (première phase d'entraînement prise en compte). Les causes principales de cet effet sont que lorsque le nombre de phases d'entraînement augmentent, le nombre de valeurs α à annuler par entraînement diminue et que la solution optimale $\boldsymbol{\alpha}_0$ est plus représentative des solutions optimales $\boldsymbol{\alpha}_k$ car le nombre d'exemples en commun entre les bases d'entraînement correspondantes augmente. Ces deux causes ont respectivement pour effet de réduire les temps nécessaires à la production d'une solution d'initialisation $\tilde{\boldsymbol{\alpha}}$ et de réduire le nombre

19. $r_k \approx k(1-1/k)^\beta$ est une formulation plus précise ($\beta \approx 2$ avec SMO), mais cette première approximation est suffisante pour les expérimentations réalisées, en particulier pour des valeurs de $k > 10$.

$k=$			5		10		20		50		100	
C	γ	e_{v}	r_k	e_{kCV}	r_k	e_{kCV}	r_k	e_{kCV}	r_k	e_{kCV}	r_k	e_{kCV}
0,25	0,001	13,1	1,1	13±4,1	1,35	12±5,7	1,55	12,7±7,4	1,62	12,7±12	1,7	13±18
0,25	4	42,1	1,27	44±3,4	1,52	44±10	1,72	43,3±14	1,86	43,3±21	2,23	43,3±28
1	0,016	14,4	0,60	14,7±5,1	0,75	14,7±6,7	0,90	14,7±7,0	1,11	14,7±13	1,37	14,7±19
16	0,016	13,6	0,56	34,3±18	0,74	35±23	1,14	28±16	1,28	22±20	2,7	19,3±23
16	16	37,9	1,19	42,7±2,24	1,51	40,3±8,3	1,88	41,3±13	2,44	40,7±20	2,75	40,3±27

TAB. 4.8 – *Résultats de validation croisée en k parties avec la base* **Australian** *pour certains couples d'hyper-paramètres (C,γ). Les colonnes r_k représentent le coût relatif des k itérations de l'algorithme 7 par rapport à la première phase d'apprentissage. e_{kCV} correspond à l'estimation de l'erreur de généralisation avec la procédure de validation croisée en k parties (les valeurs sont des pourcentages).*

$k=$			5		10		20		50		100	
C	γ	e_{v}	r_k	e_{kCV}	r_k	e_{kCV}	r_k	e_{kCV}	r_k	e_{kCV}	r_k	e_{kCV}
0,25	0,001	24,1	0,74	24,6±2,3	0,82	24,6±4,2	0,89	24,2±5,7	0,95	24,6±8,7	0,99	24,6±11
1	0,25	16,4	1,12	21,6±2,1	1,30	22,1±3,8	1,41	21,1±3,7	1,54	21,5±8,6	1,44	21,8±11
1	16	23,9	1,62	24,6±2,3	1,94	24,7±3,9	2,09	24,9±5,6	2,06	24,7±8,6	2,12	24,8±11
4	0,016	15,6	0,65	29,9±8,85	0,77	30,1±4,9	0,89	23,7±4,1	0,93	24,5±9,1	1,09	23,8±11
16	16	23,9	1,61	24,6±2,4	1,86	24,7±3,9	2	24,9±5,6	2,16	24,7±8,8	2,2	24,7±11

TAB. 4.9 – *Résultats de validation croisée en k parties avec la base* **Adult** *pour certains couples d'hyper-paramètres (C,γ). Les colonnes r_k représentent le coût relatif des k itérations de l'algorithme 7 par rapport à la première phase d'apprentissage. e_{kCV} correspond à l'estimation de l'erreur de généralisation avec la procédure de validation croisée en k parties (les valeurs sont des pourcentages).*

d'itérations nécessaires à l'algorithme SMO pour atteindre le CAP pour chacun des ces entraînements. Notons que déjà à partir de $k = 10$ l'accélération de l'estimation de e_{kCV} est significative.

Ces résultats illustrent également que les valeurs e_{kCV} sont fortement corrélées avec les valeurs de e_{v} dans la grande majorité des cas (plus avec la base **Australian** qu'avec la base **Adult**), c'était une observation normalement attendue. En examinant plus précisément les résultats des tableaux 4.8 et 4.9, il apparaît que certains couples (C,γ) sont moins sensibles aux variations dues à l'échantillonnage de la base d'apprentissage que d'autres, cet effet semble diminuer lorsque la taille de la base d'entraînement augmente (exemples : (C ;γ)=(16 ; 0,016) avec la base **Australian** et (C ;γ)=(4 ; 0,016) avec la base **Adult**). Une régularisation importante (C trop grand) peut être la cause de cet effet. Le maintien ou l'écart d'exemples plus ou moins représentatifs du problème d'apprentissage (*cf.* section 5.3.1), conséquence de la procédure d'échantillonnage, peut expliciter cette sensibilité. Dans les pires cas, un couple d'hyper-paramètres peut être optimal pour l'apprentissage avec l'ensemble des données (identifiable avec des valeurs de k importantes), alors qu'avec un sous-ensemble réduit de ces données (représenté par l'utilisation de valeurs faibles de k) ce couple n'est pas identifié comme optimal. Des résultats du chapitre 5 vont également dans ce sens. L'analyse des valeurs de e_{kCV} pour différentes valeurs de k permet également d'identifier des couples (C,γ) qui sont moins sensibles aux variations des exemples de la base d'apprentissage[20].

20. Ces résultats sont particulièrement intéressants dans le cadre des idées développées dans le chapitre 5, si l'on souhaite remplacer la règle du plus proche voisin par la fonction de décision produite par un SVM. Pour plus de détails, voir le chapitre 5, en particulier les remarques de la conclusion.

Bien que l'objectif premier de ce chapitre soit de produire des méthodes d'estimations rapides de l'erreur pour différentes procédures de validation croisée, ces remarques semblent indiquer que pour produire une fonction de décision optimale avec des SVM à partir d'un ensemble de données fixé et avoir une estimation la plus précise possible de ses capacités de généralisation, il est préférable, dans le cas d'une procédure de validation croisée en k parties, d'utiliser une valeur de k suffisamment grande. La variance est incontestablement plus grande, mais comme cela est vrai pour tous les modèles, la valeur moyenne est celle la plus représentative. Notre méthode permet alors de réaliser cette estimation, pour une valeur de k importante, avec un coût en temps exploitable. Une autre alternative est de choisir comme hyper-paramètres ceux qui produisent un taux d'erreur faible et stable (*i.e* fortement indépendant de la valeur de k). Pour cette autre alternative, notre méthode permet également de réduire très significativement le coût de mesure de cette stabilité. Le choix entre ces deux façons de procéder n'est pas évident et des réflexions futures devront être réalisées.

4.2.5 Cas de l'estimation rapide de l'erreur de *bootstap*

4.2.5.1 Spécificités

Dans le cas du *boostrap*, certains exemples peuvent être présents plus d'une fois dans la base d'entraînement. Comme cela a été indiqué à la section 4.1.5.2 de ce chapitre, il est possible de prendre en compte la présence d'exemples dupliqués dans la base d'entraînement d'un SVM de façon astucieuse. Les valeurs des vecteurs $\mathbf{B}_i$ indiquent quel sont les exemples qui sont dupliqués plusieurs fois et en quelle quantité. L'algorithme SMO utilise alors les valeurs $\mathbf{B}_i$ pour fixer une valeur de pénalisation $C_j = \mathbf{B}_i(j)C$ à chaque exemple présent dans la base d'entraînement (*cf.* section 4.1.5.2). Cette base d'entraînement utilise en moyenne 63% des exemples dans la base initiale [EFRON97] (doublons non comptés), ce qui oblige généralement notre méthode à devoir annuler un nombre important de $\tilde{\alpha}_j$ dans $\tilde{\boldsymbol{\alpha}}$ (comme dans le cas d'une validation croisée en k parties lorsque k est égal à 3), en particulier quand un modèle testé est peu approprié et produit une solution $\boldsymbol{\alpha_0}$ avec un grand nombre de vecteurs de support.

Pour réduire le nombre de variables à annuler, nous allons définir une méthode qui permet de prendre en compte les solutions produites par chaque phase d'entraînement de SMO sur différentes bases $\mathbf{B}_i$. L'idée est que le nombre de vecteurs de support pour une solution $\boldsymbol{\alpha_i}$ produite à partir de $\mathbf{B}_i$ est plus faible que le nombre de vecteurs de support pour une solution $\boldsymbol{\alpha_0}$ produite à partir de B_0, tout au moins dans la majorité des cas. La raison est simple et elle est liée au fait que les bases $\mathbf{B}_i$ ont en probabilité un nombre d'exemples (doublons non comptés) de 63% de la base initiale. Comme le nombre de vecteurs de support est proportionnel à la taille de la base pour un modèle donné, il est normal d'observer cet effet. Il est alors plus judicieux de prendre une des solutions $\boldsymbol{\alpha_i}$ d'un précédent entraînement que celle du premier entraînement $\boldsymbol{\alpha_0}$. Pour que cette solution soit suffisamment informative en tant que solution d'initialisation adaptée, il est également nécessaire qu'elle partage un grand nombre d'exemples en commun avec la nouvelle base d'entraînement. C'est forcément le cas entre la base $\mathbf{B}_0$ et une base $\mathbf{B}_i$ particulière, puisque la base $\mathbf{B}_0$ utilise tous les exemples en un exemplaire, mais entre deux bases $\mathbf{B}_i$ différentes le nombre d'exemples en commun peut être faible. Il est important de prendre en compte ces deux

types d'informations et de réaliser un compromis pour choisir un vecteur d'initialisation $\boldsymbol{\alpha_i}$ performant à partir des résultats d'entraînement antérieurs. Pour réaliser ce compromis, deux notions de distance ont été définies.

La première mesure la proximité entre deux bases d'entraînement $\mathbf{B}_1$ et $\mathbf{B}_2$ en prenant en compte le nombre d'occurrences en commun pour chaque exemple z_j présent dans ces deux bases :

$$d_p(\mathbf{B}_1,\mathbf{B}_2) = \frac{1}{2m}\sum_{j=1}^{m}|\mathbf{B}_1(j) - \mathbf{B}_2(j)| \tag{4.66}$$

La deuxième mesure l'adéquation d'une solution $\boldsymbol{\alpha}$ comme solution d'initialisation de SMO pour une base d'entraînement B donnée :

$$d_a(\mathbf{B},\boldsymbol{\alpha}) = \frac{1}{0{,}368\ m}\sum_{j=1}^{m} I\left(\mathbf{B}\left(j\right) = 0 \wedge \alpha_j > 0\right) \tag{4.67}$$

Elle compte le nombre de variables α_j qu'il sera nécessaire d'annuler, si la solution $\boldsymbol{\alpha}$ est utilisée comme valeur d'initialisation de SMO. Cette mesure est normalisée[21] par rapport au nombre moyen d'exemples non utilisés dans une procédure de *bootstrap*.

A partir de ces deux mesures de distances et en tenant compte des précédentes remarques, nous avons défini un critère de qualité d'une solution $\boldsymbol{\alpha}_1$ relativement à la base $\mathbf{B}_1$ qui l'a produite pour l'entraînement avec une nouvelle base $\mathbf{B}_2$ comme :

$$q(\mathbf{B}_2,\mathbf{B}_1,\boldsymbol{\alpha}_1) = d_p(\mathbf{B}_2,\mathbf{B}_1)\cdot d_a(\mathbf{B}_2,\boldsymbol{\alpha}_1) \tag{4.68}$$

Une autre particularité du *boostrap* est que tous les vecteurs de support qui sont limités n'ont pas forcément la même valeur α car le nombre d'occurrences n'est pas le même. La procédure d'annulation rapide de couples de variables α dans le cas de la procédure de validation croisée en k parties doit alors être adaptée. Elle reste valide si deux vecteurs limités sont de classes différentes et qu'ils ont respectivement le même nombre d'occurrences d'exemples. L'algorithme 8 (*FAS-SMO-BOOTSTRAP : Fast Alpha Seeding SMO bootstrap*) prend en compte cette particularité. Quelques remarques sur les différences entre cet algorithme et le précédent :

– La procédure STOCKERSOLUTION enregistre les données $(\boldsymbol{\alpha},\mathbf{G})$ d'une solution $\mathbf{B}_i$, pour qu'elle puisse être retrouvée par la suite. La procédure SOLUTION réalise l'action contraire qui consiste à retrouver les données d'un précédent entraînement avec une base $\mathbf{B}_i$ précisée.

21. Il est possible, dans certains cas, que d_a soit supérieure à 1. Cela se produit lorsque la base utilise peu d'exemples (donc beaucoup de doublons) et que pour la solution $\boldsymbol{\alpha}$ la majorité des exemples sont des vecteurs de support. Il n'est pas problèmatique d'avoir une valeur supérieure à 1, dans ce cas là, car elle traduit le fait que cette solution est fortement inadaptée comme initialisation de SMO pour cette base d'entraînement.
Dans le cas contraire où la base $\mathbf{B}$ utilise peu de doublons, la valeur de d_a sera systématiquement inférieure à 1 quelle que soit la solution $\boldsymbol{\alpha}$ considérée. Dans ce cas, la base $\mathbf{B}$ a peu de variables α_i à annuler puisqu'elle utilise plus d'exemples (> 63%) de la base initiale. C'est la proximité entre les deux bases, mesurée par d_p, qui doit alors être favorisée pour que les résultats d'entraînement avec la première soient fortement informatifs pour l'initialisation de SMO pour la seconde. Dans ces conditions la limitation de d_a à une valeur inférieure à 1 dans le pire des cas n'est pas problématique.
Au regard de la définition du compromis (4.68) réalisé à partir de ces deux mesures d_p et d_a et des remarques précédentes, la valeur du coefficient de normalisation semble être un bon choix.

- L_{fait} est une liste qui contient toutes les bases qui ont déjà été utilisées dans une des phases d'entraînement précédentes.
- GÉNÉRERBASEAVECREMISE est la procédure qui réalise la constitution de k tirages aléatoires avec remise à partir de la base initiale.
- MEILLEUREQUALITÉ réalise la recherche de la base de référence $\mathbf{B}_{\text{ref}}$ qui est la plus adéquate avec la base $\mathbf{B}_i$ parmi les bases déjà traitées (i.e dans L_{fait}) à partir de l'utilisation du critère de qualité (4.68).
- $o_{\max}$ correspond au nombre maximum d'occurrences pour l'ensemble des bases d'entraînement $\mathbf{B}_i$ produites par la procédure GÉNÉRERBASEAVECREMISE.
- Les procédures ENREGISTRERSTATISTIQUESNAIVE et ENREGISTRERSSTATISTIQUEOOB enregistre les informations nécessaires au calcul des deux estimations d'erreurs : naïve, OOB (*Out Of Bag*) telles que décrites à la section 2.6.3. Le taux d'erreur $e_{0.632}$ est ensuite facile à déterminer à partir de ces statistiques.

4.2.5.2 Résultats expérimentaux

Le même type d'expérimentations que pour la validation croisée en k parties ont été réalisées. Le nombre maximum d'échantillons est fixé à 200 et ils sont communs à l'ensemble des couples d'hyper-paramètres testés. Les durées d'entraînement et les valeurs de l'erreur 0.632 (*cf.* section 2.6.3) sont mesurées dès que les 20, 50, 100 et 200 premiers échantillons de la base d'apprentissage sont utilisés. Les tableaux 4.10 et 4.11 donnent les résultats obtenus à partir de ces mesures. Dans ces tableaux r_k garde la même signification. Ces résultats illustrent qu'il y a une accélération significative pour l'estimation de l'erreur de *bootstrap* par rapport a une procédure classique des entraînements, même si elle n'est pas aussi importante qu'avec les deux procédures de validation croisée précédentes. Pour information, les valeurs moyennes de r_k pour k égal 200 sont de 23 avec la base **Australian** et de 16 avec la base **Adult**, alors qu'avec une procédure classique elles sont respectivement de 61 et 65. Le rapport r_k/k a une diminution faible lorsque k augmente (les valeurs moyennes des rapports $r_{20}/20$ et $r_{200}/200$ sont respectivement de 0,155 et 0,115 avec la base **Australian** et de 0,085 et 0,080 avec la base **Adult**). Si la recherche d'une solution performante d'initialisation avec l'utilisation du critère (4.68) permet de réduire en moyenne les temps d'entraînement d'un SVM avec l'augmentation du nombre d'échantillons déjà traités, la durée nécessaire à cette recherche augmente également avec ce nombre d'échantillons. Ce deuxième effet réduit globalement l'accélération de l'estimation de $e_{0.632}$.
Pour réduire les temps nécessaires à cette recherche, il n'est sûrement pas nécessaire d'enregister l'ensemble des solutions relatives aux différents $\mathbf{B}_i$. En effet, lorsque deux $\mathbf{B}_i$ sont proches, si la solution relative à l'entraînement de l'une est utile à l'autre, la solution relative à l'autre ne sera que faiblement utile aux $\mathbf{B}_i$ suivants, connaissant déjà la solution de la première de ces deux bases. L'ordre de traitement des $\mathbf{B}_i$ a également une influence sur la possibilité de sélectionner une solution appropriée et cet ordre doit pouvoir être changé pour optimiser le choix des solutions utiles à la production de solutions d'initialisation efficaces à partir d'un nombre réduit de variables α à annuler. Une possibilité est de commencer par les bases $\mathbf{B}_i$ qui exploitent le plus d'exemples différents (*i.e.* non dupliquées), car elles seront toutes proches de $\boldsymbol{\alpha}_0$. L'ensemble de ces remarques indique qu'il est possible d'améliorer la procédure d'initialisation de SMO dans le cadre du *bootstap* pour réduire les temps d'estimation de $e_{0.632}$.

Les valeurs de $e_{0.632}$ dans les tableaux 4.10 et 4.11 illustrent que la corrélation entre $e_{0.632}$ et e_{v} est (logiquement) importante. Comme pour la section expérimentale précédente, ces résultats mettent en évidence que les mêmes couples (C,γ) produisent des fonctions de décisions très sensibles à la constitution de la base d'entraînement[22].

22. Dans ces expérimentations, les estimations de e_{v} avec $k = 20$ et $k = 200$ sont sensiblement les mêmes pour la sélection d'un couple (C,γ). Néanmoins, dans le cas de problèmes de sélections d'attributs, par exemple, il est connu qu'il est généralement nécessaire de prendre une valeur de k d'autant plus grande que la taille de la base d'apprentissage est réduite et que le nombre d'attributs est important.Pouvoir réaliser rapidement des estimations de e_v avec des valeurs élevées de k reste donc un défi important.

Algorithme 8 FAS-SMO-BOOTSTRAP(Z_m,k)

$(\boldsymbol{\alpha}_0,\mathbf{G}_0) \leftarrow$ SMO$((0)^m,(-1)^m,10^{-3})_{B_0}$
STOCKERSOLUTION$((\boldsymbol{\alpha}_0,\mathbf{G}_0), \mathbf{B}_0)$
$(q_Q,A) \leftarrow$ DERNIERENREGISTREMENTSMO()
$\tilde{\epsilon} \leftarrow$ NOTRE-CRITÈRE-ARRÊT(q_Q,A)
AJOUTERDANSLISTE$(L_{\text{fait}},\mathbf{B}_0)$
$L_{\text{afaire}} \leftarrow$ GÉNÉRERBASEAVECREMISE(k,m)
pour $\mathbf{B}_i \in L_{\text{afaire}}$ **faire**
  $\mathbf{B}_{\text{ref}} \leftarrow$ MEILLEUREQUALITÉ$(\mathbf{B}_i,L_{\text{fait}})$
  $(\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}}) \leftarrow$ SOLUTION$(\mathbf{B}_{\text{ref}})$
  $o_{\max} \leftarrow \arg\max_{1\leq j\leq m}(\mathbf{B}_i(j))$
  pour $o \in [1, o_{\max}]$ **faire**
    $I_{\oplus} \leftarrow \{j : \mathbf{B}_i(j) = 0,\tilde{\alpha}_j = o \cdot C, y_j = +1\}$
    $I_{\ominus} \leftarrow \{j : \mathbf{B}_i(j) = 0,\tilde{\alpha}_j = o \cdot C, y_j = -1\}$
    $n_a \leftarrow \min(|I_{\oplus}|,|I_{\ominus}|)$
    $(\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}}) \leftarrow$ ANNULERCOUPLES$(I_{\oplus},I_{\ominus},n_a,\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}})$
  fin pour
  $J \leftarrow \{r : \mathbf{B}_i(r) = 0, \tilde{\alpha}_r > 0\}$
  TRIER$(J,\tilde{\boldsymbol{\alpha}})$ \\ Tri par ordre croissant des valeurs $\tilde{\alpha}_{j\in J}$
  pour $j \in J$ **faire**
    $I \leftarrow \{r|(\mathbf{B}_i(r) > 0) \vee (y_r \neq y_j \wedge \mathbf{B}_i(r) = 0)\}$
    $(\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}}) \leftarrow$ NOTRE-AS$(\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}}, j, \mathbf{B}(I))$
    $(\boldsymbol{\alpha},\mathbf{G}) \leftarrow$ SMO$(\tilde{\boldsymbol{\alpha}},\tilde{\mathbf{G}},\tilde{\epsilon})_{B_i}$
    STOCKERSOLUTION$((\boldsymbol{\alpha},\mathbf{G}), \mathbf{B}_i)$
    AJOUTERDANSLISTE$(L_{\text{fait}},\mathbf{B}_i)$
    ENREGISTRERSTATISTIQUESNAIVE$(\boldsymbol{\alpha}, \mathbf{G}, Z_m^{\mathbf{B}_i})$
    ENREGISTRERSTATISTIQUESOOB$(\boldsymbol{\alpha}, \mathbf{G}, Z_m^{\bar{\mathbf{B}}_i})$
  fin pour
fin pour
retourner l'erreur $e_{.632}$ à partir des statistiques enregistrées

$k=$			20		50		100		200	
C	γ	e_{v}	r_k	$e_{0.632}$	r_k	$e_{0.632}$	r_k	$e_{0.632}$	r_k	$e_{0.632}$
0,25	0,001	13,1%	2,06	13,2%	5,73	13,1%	11,6	14,4%	22,1	13,4%
0,25	4	42,1%	3,35	33,5%	8,08	33,6%	15,5	34,6%	32	33,8%
1	0,016	14,4%	3,25	12%	8,73	12%	16,1	12,4%	28,7	13,1%
16	0,016	13,6%	4,71	21,7%	11,1	18,2%	15,2	24,8%	18,9	27,1%
16	16	37,9%	2,15	44,6%	4,36	44,7%	9,12	46,8%	23,4	46,6%

TAB. 4.10 – *Résultats du* bootstrap *avec k échantillonages de la base* **Australian** *pour certains couples d'hyper-paramètres (C,γ). Les colonnes r_k représentent le coût relatif des k itérations de l'algorithme 8 par rapport à la première phase d'apprentissage. $e_{0.632}$ correspond à l'estimation de l'erreur de généralisation avec la procédure de* bootstrap.

$k=$			20		50		100		200	
C	γ	e_{v}	r_k	$e_{0.632}$	r_k	$e_{0.632}$	r_k	$e_{0.632}$	r_k	$e_{0.632}$
0,25	0,001	24,1%	0,91	24,6%	2,51	24,5%	5,52	24,6%	10,6	24,6%
1	0,25	16,4%	1,79	20,1%	4,2	19,3%	8,71	19,2%	17,8	19,8%
1	16	23,9%	1,93	23,9%	4,98	23,7%	14,3	23,4%	29,9	23,6%
4	0,016	15,6%	1,5	18,1%	2,26	23,9%	8,05	22,3%	14	21,8%
16	16	23,9%	1,5	24,6%	3,51	24,5%	7,27	24,6%	7,83	24,6%

TAB. 4.11 – *Résultats du* bootstrap *avec k échantillonages de la base* **Adult** *pour certains couples d'hyper-paramètres (C,γ). Les colonnes r_k représentent le coût relatif des k itérations de l'algorithme 8 par rapport à la première phase d'apprentissage. $e_{0.632}$ correspond à l'estimation de l'erreur de généralisation avec la procédure de* bootstrap.

4.2.6 Conclusion et discussion

Dans ce chapitre, nous avons rappelé les détails mathématiques et techniques relatifs à la résolution du problème lié aux SVM. Ces explications ont permis une description des principes de base de l'algorithme SMO. Les conditions d'initialisation et d'arrêt classique de cet algorithme ont été présentées, ainsi que les possibilités de changement de ces conditions. Nous nous sommes ensuite intéressés à l'exploitation de ces possibilités pour réduire les temps nécessaires à l'estimation de l'erreur de généralisation dans le cadre de l'utilisation de techniques de validation croisée. Dans un premier temps, nous nous sommes restreint à l'estimation de l'erreur de *leave one out*. Un état de l'art des techniques d'*alpha seeding* et d'utilisation de critères d'arrêt prématuré, pour l'estimation de cette erreur avec l'algorithme SMO, a été réalisé. Nous avons proposé des améliorations de ces deux points et argumenté les raisons de ces choix. Les résultats expérimentaux mettent en évidence que la méthode que nous avons définie à partir de ces améliorations est plus performante que les précédentes. L'estimation de l'erreur de *leave one out* avec des bases comportant plusieurs centaines d'exemples est maintenant exploitable dans des temps raisonnables. Nous avons ensuite étendu les principes de notre méthode à l'estimation de l'erreur de validation croisée en k parties et l'erreur de *bootstrap*. Les résultats expérimentaux mettent en évidence que l'accélération pour l'estimation de l'erreur de validation croisée en k parties est importante, notamment lorsque la valeur de k est élevée. Ces résultats mettent également en évidence que l'accélération de la procédure de *bootstrap* est significative et nous donnons des indices pour améliorer dans le futur cette méthode.

Plusieurs remarques peuvent être formulées en marge de ces résultats :

1. La valeur ϵ du critère d'arrêt peut être utilisée comme un paramètre libre. Il est alors possible de l'utiliser pour découvrir des modèles prometteurs (et écarter les autres) dans des temps réduits en choisissant initialement des valeurs suffisament grandes de ϵ. De plus les mauvaises solutions sont généralement celles qui demandent le plus d'itérations à SMO dans le cas d'une valeur faible de ϵ. La conséquence est que davantage de temps peut être consacré à explorer l'espace de recherche. Les implications pourraient être importantes pour améliorer la procédure de recherche avec tabous du chapitre 5 ou l'algorithme évolutionnaire du chapitre 6 (ϵ aurait un rôle similaire, d'un certain point de vue, à la température dans le recuit simulé).
2. AdaBoost (*cf.* section 2.5.1) utilise de multiples phases d'entraînement. Notre algorithme FAS-SMO-BOOSTRAP doit pouvoir être adapté à AdaBoost lorsque les classificateurs de base sont des SVM et que chaque SVM impliqué utilise la même fonction noyau[23].
3. Une autre piste prometteuse est d'étendre les techniques d'*alpha seeding* à d'autres modifications que la présence ou non de certains exemples d'apprentissage comme les attributs, les paramètres libres de fonctions noyaux, la constante[24] C, etc.

23. Ce n'est pas toujours le cas avec AdaBoost [XUCHUN05].
24. Il existe déjà une méthode lorsque seulement C est modifié [HASTIE04A].

CHAPITRE 5

CONSTRUCTION DE FONCTIONS DE DÉCISION SIMPLIFIÉES À PARTIR DE SVM

Sommaire

5.1 Introduction

La construction de fonctions de décision simplifiées est essentielle dans les applications ayant des contraintes de temps réel importantes. L'objectif est alors de pouvoir produire des solutions fortement parcimonieuses tout en conservant des capacités en généralisation performantes. La réduction de la complexité des fonctions de décision des SVM peut être réalisée en agissant sur deux points essentiels [1] :

– la réduction de la taille de la base d'apprentissage, en ne conservant que les exemples les plus informatifs,
– la sélection des attributs les plus utiles pour un problème donné de classification.

La complexité, en temps d'évaluation de la fonction de décision d'un SVM, est linéairement dépendante du nombre de vecteurs de support. Comme ce nombre est borné par le nombre d'exemples, l'utilisation d'une base d'apprentissage de taille réduite permet de s'assurer une limitation contrôlée de sa complexité. Les SVM ont par nature la possibilité de sélectionner un nombre réduit d'exemples comme vecteurs de support, mais les données d'apprentissage étant bruitées dans la majorité des cas, ce nombre est généralement élevé. En effet, le nombre de vecteurs de support est proportionnel au taux d'erreur minimum réalisable sur la base d'apprentissage et à la taille de la base d'apprentissage (*cf.* équation (2.52) à la section 2.4.5.6) . Bien que la notion de marge molle soit importante pour les SVM, elle a pour effet indésirable de capturer l'ensemble des exemples mal classés pour définir la fonction de décision des SVM. Il est alors envisageable de supprimer de la base d'apprentissage les exemples les moins informatifs pour réduire la complexité de la

1. Les fonctions noyaux peuvent être plus ou moins complexes à évaluer. Le choix d'une fonction noyau par rapport à une autre peut donc influencer les temps de décision. Dans cette étude nous ne tiendrons pas compte de cette possibilité.

base d'apprentissage. De cette idée découle la possiblité que la suppression préalable des données les plus bruitées permet de produire des fonctions de décision plus performantes avec l'algorithme des SVM. Plusieurs résultats expérimentaux dans ce chapitre confirment cette possibilité. Un autre effet intéressant de la réduction de la base d'apprentissage est qu'elle permet de réduire les temps d'entraînement des SVM. Cet effet positif peut être annulé par la complexité de l'algorithme réalisant cette réduction. La difficulté est alors de définir une méthodologie permettant de réaliser une réduction pertinente dans des temps raisonnables. Deux pistes sont explorées dans ce chapitre pour prendre en compte ces considérations. L'une est basée sur l'utilisation de la règle du Plus Proche Voisin (PPV) et l'autre sur l'utilisation de techniques issues de la classification non supervisée.

5.1.1 Exploitation de la règle PPV

La première approche proposée exploite les capacités prédictives de la règle PPV pour estimer les capacités informatives d'un ensemble d'apprentissage. L'avantage de la règle PPV est qu'il n'y a pas de temps d'apprentissage car il n'y pas d'hyper-paramètres à régler (si la distance utilisée est fixée) comme dans le cas des SVM à partir du moment que tous les exemples d'apprentissage sont utilisés.

Lorsqu'un sous-ensemble des exemples d'apprentissage doit être sélectionné, il existe une importante littérature sur les méthodes de condensation et d'édition propres à la règle PPV [KUNCHE04] (*cf.* section 2.4.1). Ces méthodes ont pour objectif de réduire les temps conséquents de décision de la règle PPV et d'améliorer les performances en généralisation sur des données bruitées. Dans le chapitre 2, nous avons insisté sur le fait que la sélection d'un sous-ensemble d'exemples réduit correspondait à la définition d'un principe d'inférence de minimisation du risque structurel pour la règle PPV et de façon plus générale correspondait à la réalisation d'un schéma de compression. La règle PPV a des connexions fortes avec celles produites par les SVM lorsqu'un noyau gaussien est utilisé [KARACA04]. Il est également possible d'étendre la règle PPV pour qu'elle puisse *travailler* directement dans l'espace de redescription [KARACA04]. Pour l'ensemble de ces raisons, nous pensons que l'exploitation correcte de cette règle permet de définir des méthodes de sélection d'exemples informatifs utiles à la production de fonctions de décision rapides et performantes avec les SVM. Nous montrerons, entre autres, comment ces méthodes utilisent des critères de marge géométrique ou d'hypothèse pour garantir une robustesse à la méthode de sélection.

5.1.2 Exploitation de techniques de classification non supervisées

Un problème rencontré avec la majorité des algorithmes d'apprentissage est que les temps d'entraînement augmentent rapidement avec la taille de la base d'apprentissage. La sélection d'un sous-ensemble de données pertinentes, telle qu'évoquée dans le cadre de la première approche, peut devenir inexploitable si les temps de traitement deviennent conséquents, et cela même si elle produit une sélection performante. Dans la plupart des cas, l'augmentation de la taille de la base d'apprentissage permet d'obtenir des estimations plus performantes en généralisation, mais elle s'accompagne d'une redondance qui concerne à la fois les exemples informatifs et les non informatifs. Il en va de même pour les

attributs. Prenons pour exemple le domaine de l'imagerie. Il est facile aujourd'hui d'utiliser de nombreuses catégories d'attributs pour caractériser des objets dans des images : forme, couleur, texture, etc. Le constat est le même pour de nombreux autres domaines car actuellement les capacités de stockage ne sont plus un facteur limitatif. Outre les problèmes de malédiction de la dimensionnalité [BELLMA61], l'ajout d'attributs redondants conduit souvent à une augmentation inutile des temps d'apprentissage. Comme le sujet essentiel de ce chapitre concerne la production de fonctions de décision de complexités réduites, cette redondance d'attributs doit être également évitée. La réduction du nombre d'attributs utilisés peut, par la même occasion, réduire les effets indésirables de la malédiction de la dimensionnalité.

L'idée principale de cette seconde approche est d'exploiter des techniques de classification non-supervisées pour réduire la redondance inutile dans les données, et ainsi découvrir une représentation simplifiée et représentative de la base initiale. Le but est de réaliser, certes avec une certaine approximation, un mécanisme de condensation similaire à la première approche. Comme les SVM sont sensibles à une sélection efficace des hyper-paramètres du noyau et de la constante de régularisation C, cette seconde approche a également pour objectif de réaliser une optimisation de ces paramètres. La recherche d'un modèle performant combinant le choix des hyper-paramètres des SVM, la simplification de la base en nombre d'exemples et en nombre d'attributs utilisés correspond à l'exploration d'un espace de recherche de dimension importante. Pour pallier cet inconvénient, une méthode méta-heuristique à base de recherche avec tabous a été définie. Un nouveau critère de qualité qui correspond à un compromis entre capacité de généralisation et rapidité de la décision est proposé. Il permet à l'utilisateur de spécifier quel taux de dégradation de la qualité prédictive de la fonction de décision il est prêt à accepter, si cela conduit à une amélioration significative des temps de décision[2].

5.1.3 Organisation du chapitre

Dans la première partie de ce chapitre nous rappelons l'importance de la sélection d'un modèle efficace pour réaliser un apprentissage de qualité avec l'exploitation de la règle PPV ou des SVM. En particulier, nous montrons le rôle essentiel de la sélection d'exemples et d'attributs pertinents dans le choix d'un modèle. Nous insistons sur le fait que ce rôle est encore plus important lorsque l'objectif est de produire une fonction de décision de complexité réduite. Un état de l'art des approches qui réduisent le nombre d'exemples et d'attributs pour les SVM et la règle PPV est ensuite réalisé. Comme notre étude s'est portée davantage sur la réduction du nombre d'exemples, nous insistons sur cette possibilité. L'importance de la sélection des hyper-paramètres pour les SVM est rappelée, et il est précisé qu'elle peut également contribuer à la réduction du nombre d'exemples exploités par la fonction de décision.

Dans la seconde partie de ce chapitre, la première approche qui a pour objectif de réaliser la réduction du nombre d'exemples par sélection itérative d'exemples pertinents est présentée. Nous montrons le parallèle réalisable entre attributs pertinents et exemples pertinents. Le principe général de notre méthode est ensuite exposé et nous montrons qu'elle correspond à la réalisation d'un schéma de compression. Quatre critères basés sur des

2. Dans le chapitre 7 on propose une application de son utilisation pour réaliser un classificateur de pixels rapide, dans le cadre d'un schéma de segmentation d'images couleur.

critères de confiance sont proposées pour guider notre schéma de compression dans sa sélection itérative des exemples pertinents. Plusieurs résultats expérimentaux sont proposés pour illustrer à la fois la simplification des fonctions de décision produites, mais également les effets sur les capacités en généralisation. Des modifications du critère d'arrêt relatif à notre algorithme sont proposés pour les deux critères de confiance les plus performants, ceci afin de les optimiser. La méthode de sélection d'exemples pertinents proposée ayant des temps de calcul importants, nous proposons une méthode simple afin de pouvoir appliquer notre méthode à des bases de données de tailles plus importantes. Plusieurs expérimentations sont réalisées avec ces améliorations. Nous terminons par une discussion sur cette première approche. Notre deuxième approche utilise la quantification vectorielle pour produire des prototypes représentatifs de la base d'apprentissage. Nous montrons en premier lieu comment la quantification vectorielle peut produire une simplification utile à une sélection rapide des hyper-paramètres des SVM. Nous montrons également que la notion d'hyper-paramètres optimaux est dépendante de la simplification appliquée à la base d'apprentissage. Ces résultats impliquent que la sélection de modèle doit prendre en compte simultanément le choix des hyper-paramètres et des simplifications à réaliser pour être efficace. Cette efficacité dépend du compromis à réaliser entre augmentation des capacités de généralisation et diminution de la complexité des fonctions de décision produites. Nous proposons un nouveau critère de qualité qui mesure l'efficacité de ce compromis. Pour ce critère de qualité l'utilisateur peut spécifier l'importance de la réduction de complexité par rapport aux capacités de généralisation. Nous proposons ensuite une méthode méta-heuristique à base de recherche avec tabous utilisant des techniques de diversification et d'intensification pour réaliser une sélection efficace d'un modèle qui optimise au mieux ce critère de qualité. Plusieurs expérimentations sont réalisées sur des bases de données de référence.

Remarques: Dans l'annexe A est donnée la description de l'ensemble des bases d'apprentissage qui sont utilisées avec nos deux approches. Dans cette annexe figurent également de nombreux résultats correspondant à des utilisations classiques des méthodes PPV et SVM. Ces résultats servent de référence à la première approche plus particulièrement.

5.2 Importance de la sélection d'un modèle

Dans le chapitre 4 nous avons vu l'importance de la section d'un modèle sur les capacités en généralisation d'un SVM lorsque certains paramètres sont libres, en l'occurrence, la largeur de bande d'un noyau gaussien[3] σ et la valeur de la constante C de régularisation. En effet, si l'algorithme des SVM permet de sélectionner un hyperplan de séparation optimal, lorsque les différents paramètres libres sont fixés, la façon de réaliser le choix des valeurs optimales à donner aux paramètres libres d'un modèle reste à définir. La procédure de type *grid search* [CHANG01], utilisée dans le chapitre 4, est suffisante dans le cas où le

3. Nous utilisons la notation σ pour l'expression d'un noyau gaussien $K(\mathbf{x}_1,\mathbf{x}_2) = \exp(\frac{\|\mathbf{x}_1-\mathbf{x}_2\|^2}{2\sigma^2})$, car la notation γ utilisée dans le chapitre 4 est maintenant réservée dans ce chapitre aux critères de marge. La relation liant γ du chapitre 4 à σ de ce chapitre étant $\gamma = \frac{1}{2\sigma^2}$.

noyau n'a qu'un paramètre à sélectionner (en plus de la valeur de la constante de régularisation C), mais elle devient difficilement exploitable avec plus de deux paramètres. Parmi les différents types de paramètres libres que l'on puisse rencontrer, il y a :

- La sélection d'un sous-ensemble d'attributs pertinents de taille n.
- L'utilisation d'un noyau avec plusieurs paramètres libres (voir par exemple [AYAT02]).
- La sélection d'un noyau i parmi un ensemble de n noyaux différents (noyau gaussien, noyau polynomial, etc.).
- L'élimination de k exemples aberrants.
- La recherche d'un sous-ensemble de m' exemples les plus pertinents.
- La recherche d'un schéma multi-classe efficace correspondant à la combinaison de k classificateurs binaires.

Cette liste n'est pas exhaustive, mais donne un aperçu des différentes approches possibles. De plus, dans le cadre d'une application d'apprentissage sur des données réelles, il n'y a rien d'exceptionnel à penser qu'il est nécessaire d'agir sur l'ensemble de tous ces choix à la fois.

La procédure de type *grid search* est limitative et lourde en nombre de phases d'entraînement dès que le nombre de paramètres libres à gérer est supérieur à deux. Une possibilité, afin de réduire l'explosion combinatoire correspondant à l'augmentation du nombre de paramètres est de définir des procédures d'apprentissage en cascade agissant individuellement sur chaque paramètre (ou un ensemble réduit de ceux-ci). Notre première approche est de cette catégorie. Cette façon de procéder peut poser des problèmes lorsque la modification d'un paramètre invalide les choix réalisés sur les autres, précédemment. Cela oblige à réaliser des procédures de *feedback* difficiles à contrôler. Une autre possibilité est d'agir sur ces différents paramètres simultanément, ou plus modestement sur un ensemble de ces paramètres. Dans le cas où l'espace des modèles est immense, l'utilisation d'heuristiques ou de méta-heuristiques permet de contrôler efficacement le problème de l'explosion combinatoire. Notre deuxième approche est de cette seconde catégorie.

L'utilisation d'un espace des modèles de taille importante a généralement pour effet d'augmenter le degré de flexibilité de l'espace des hypothèses envisageables (indépendamment du fait d'utiliser les SVM comme briques élémentaires pour la construction d'une fonction de décision). Pour limiter les problèmes de sur-apprentissage, il est nécessaire de guider le choix d'une hypothèse en contrôlant sa flexibilité. Des mesures efficaces sur des notions de marge géométrique ou d'hypothèse peuvent être prises en considération. La notion de schéma de compression efficace, donc de parcimonie, est une autre possibilité pour limiter les risques de sur-apprentissage. Si l'objectif est également de produire des fonctions de décision de complexité réduite pour accélérer les traitements des applications les utilisant, la définition de ce schéma de compression comme principe d'inférence est particulièrement approprié. Bien que différentes, nos deux approches définissent des schémas de compression.

Avant de décrire nos deux approches, nous nous intéresserons aux différentes méthodes de sélection de modèle réalisées ces dernières années et qui impliquent les SVM (ou la règle PPV), en particulier celles ayant inspiré les deux méthodes que nous proposons.

5.2.1 Approches réduisant le nombre d'exemples

Nous faisons ici référence à différentes méthodes qui ont la possibilité de réduire le nombre d'exemples utilisés par une fonction de décision produite par des SVM ou la règle PPV. Cette réduction a pour objectif, soit de réduire la complexité de la fonction de décision en acceptant une légère dégradation des performances en généralisation, soit de supprimer les exemples aberrants qui dégradent le pouvoir de généralisation d'un processus d'inférence. La suppression d'exemples non informatifs est commune à ces deux objectifs. La combinaison de ces deux objectifs peut produire des méthodes qui réduisent significativement la complexité des fonctions de décision tout en augmentant le pouvoir de généralisation.

5.2.1.1 Réduction avant entraînement

Les approches de cette catégorie sont les plus couramment utilisées. Les méthodes de condensation et d'édition, déjà citées en section 2.4.1, peuvent être exploitées pour réaliser cette réduction. L'une de ces méthodes a été proposée par Karacali et al [KARACA03]. Elle est d'un intérêt particulier, car elle est basée sur un principe de minimisation du risque structurel (*cf.* section 2.4.1.4). Dans l'absolu, elle ne garantit pas une réduction de la base initiale car le sous-ensemble d'exemples sélectionné doit induire une règle PPV cohérente avec tous les exemples de la base initiale. Cela peut conduire à sélectionner tous les exemples de la base initiale, bien qu'en pratique une réduction significative de la complexité de la règle PPV soit observée [KARACA04]. Karacali et al comparent la performance de la règle de décision PPV simplifiée à celle produite par un SVM. Dans les deux cas le même espace de redescription est utilisé. Cela est rendu possible car la relation (4.58) permet de calculer la distance entre deux points dans cet espace pour la règle PPV. Ils montrent alors, à partir de quelques expériences, que pour certains espaces de redescription, la décision PPV est plus performante que celle des SVM. Pour ceux-là, Le nombre d'exemples utilisé par les règles PPV condensées est par contre plus important que le nombre de vecteurs de support des SVM.

Ou et al [OU03] utilisent la règle des k plus proches voisins (k devant être fixé à l'avance) pour choisir les exemples d'entraînement d'un SVM. Leur méthode supprime de la base initiale chaque exemple dont les k-PPV ont la même classe car ils sont considérés comme non informatifs. L'algorithme proposé dans [OU03] commence par les exemples qui sont les plus éloignés des exemples des autres classes. Cet éloignement est mesuré à partir de la base initiale. L'algorithme répète ce processus d'élimination itérativement, jusqu'à ce qu'il n'y ait plus d'exemple à supprimer. L'objectif principal est de supprimer de la base les exemples qui ne sont pas utiles à la décision d'un SVM (*i.e.* qui ne seront pas vecteurs de support).

Shin et al [SHIN02] exploitent autrement la règle des k-PPV. Il l'utilisent pour déterminer une probabilité locale d'appartenance à chaque classe. Ils définissent à partir de ce critère une mesure de proximité de la frontière de décision et une mesure de confiance dans la classe d'un exemple. Un exemple est conservé dans la base réduite, s'il a à la fois une

mesure de proximité et de confiance suffisamment grande, ce qui caractérise sa pertinence par rapport au problème d'apprentissage[4].

Une autre possibilité est d'utiliser des méthodes de classification non-supervisée pour découvrir des régions où il y a une forte concentration d'exemples dont les descriptions sont quasi-similaires. L'objectif est de les résumer par quelques prototypes les caractérisant. Yu et al [YU03A] utilisent l'algorithme BIRCH [ZHANG96] pour la production des prototypes à partir d'une représentation hiérarchique des données d'apprentissage. Koggalage et al[KOGGAL04] utilisent l'algorithme des *k-moyennes* pour découvrir des régions dites *sécuritaires* dans lesquelles tous les exemples peuvent être supprimés.

L'algorithme des SVM est lui-même un bon candidat pour réaliser cette sélection, puisqu'il n'utilise qu'un nombre restreint d'exemples dans sa fonction de décision. Dong et al dans [DONG05] utilisent cette possibilité pour réduire la taille de la base d'apprentissage sur des jeux de données de grande taille. D'abord la méthode proposée divise la base initiale en k parties et réalise ensuite, en parallèle, un entraînement des SVM sur chaque partie de la base. La réunion de l'ensemble des vecteurs de support communs à chaque fonction de décision est utilisée pour produire une nouvelle base d'entraînement qui est de taille plus réduite que la première. Cette nouvelle base est utilisée pour produire la fonction de décision finale. Les résultats dans [DONG05] montrent que, sur de grandes bases, les capacités de généralisation sont similaires à un entraînement classique de SVM, mais les temps d'entraînement sont fortement réduits. Collobert et al [COLLOB02] proposent une méthode dont le principe est similaire, sauf que l'ensemble des fonctions de décision produites sert à l'entraînement d'un réseau de neurones pour produire la décision finale à partir des sorties de ces SVM.

L'algorithme des SVM peut également être modifié tout en gardant les principes fondateurs des SVM. Lee et al [LEE01] proposent un algorithme nommé RSVM (*Reduced* SVM) qui impose une limitation du nombre maximum de vecteurs de support en changeant la nature du problème à optimiser tout en conservant le principe essentiel de marge géométrique maximale. Dans [LIN03C] Lin et al proposent quatre implémentations différentes du principe général des RSVM et ils montrent qu'elles ont de bonnes capacités en généralisation, à condition de choisir une valeur efficace pour le nombre maximum de vecteurs de support.

Brièvement évoquons d'autres approches comme : la sélection des exemples par résolution d'un problème de programmation linéaire [YANG00], la localisation des vecteurs de support en résolvant un problème d'enveloppe convexe [ZHANG02], ou la sélection des exemples de type *core* à partir de la résolution d'un problème MEB (Minimum Enclosing Ball) [TSANG05].

5.2.1.2 Réduction après entraînement

L'idée principale de ce type d'approche est qu'une fois la phase d'apprentissage d'un SVM effectuée, il est encore possible d'analyser la structure de la fonction de décision produite plus finement pour fusionner plusieurs vecteurs de support en un seul représentant. La fonction de décision produite est moins complexe, mais elle réalise les mêmes

4. Shin et al [SHIN03] montrent également que l'exploitation de la règle *k*-PPV peut être accélérée en utilisant des structures de données et des algorithmes adaptés.

prédictions [BURGES96, ABE05]. Pour obtenir une réduction plus significative, certaines contraintes doivent être relâchées (l'équivalence n'est plus garantie). Burges propose dans une deuxième partie de son étude [BURGES96] un algorithme en deux étapes qui réalise cette relaxation, tout en garantissant une bonne approximation de la fonction de décision initiale.

5.2.1.3 Méthodes incrémentales

Nous entendons par incrémental, une méthode qui ajoute un exemple dans une base à partir d'une mesure de ses performances. Nous nous intéressons en particulier aux méthodes qui exploitent directement l'algorithme des SVM dans ce processus incrémental pour maximiser la marge géométrique tout en réduisant le nombre de vecteurs de support nécessaire à une bonne cohèrence avec les données d'apprentissage. Dans [RALAIV01] est proposée une méthode qui utilise un nombre maximum de vecteurs de support fixé. A chaque fois qu'un exemple est classé à une distance inférieure à celle de la marge actuelle, une phase d'entraînement est réalisée en ajoutant cet exemple à ceux qui sont déjà vecteurs de support. Si le nombre maximum de vecteurs de support est atteint, un critère basé sur la notion de marge géométrique permet de choisir la fonction optimale en utilisant tous les exemples sauf un. La procédure incrémentale garde trace du nombre d'erreurs que fait la fonction de décision en construction pour les exemples déjà vus. Si ce nombre d'erreurs dépasse un certain seuil, le nombre maximum de vecteurs de support fixé augmente afin de limiter le nombre d'erreurs de cette méthode. La taille initiale, le taux d'augmentation de la taille et le seuil constituent les paramètres réglables de l'algorithme. Ils permettent d'agir sur l'importance de la réduction de la fonction de décision produite en acceptant une dégradation des performances en généralisation.

Dans [CAUWEN00] c'est une méthode incrémentale et décrémentale qui est appliquée à l'estimation de l'erreur de *leave-one-out* dans un cadre d'apprentissage en ligne. Cette méthode permet d'ajouter un exemple comme pertinent et de le supprimer plus tard comme non pertinent. Le nombre de vecteurs de support n'est pas fixé à l'avance. Dans [BORDES05], une autre méthode est proposée. Elle a également la possibilité de supprimer des exemples qui étaient précédemment vecteurs de support pendant la phase d'apprentissage. Les expérimentations initiales dans [BORDES05] montrent que cette méthode peut utiliser moins de vecteurs de support qu'un apprentissage classique avec les SVM [CHANG01]. Dans [LOOSLI06B], d'autres résultats montrent que sur une base d'apprentissage de taille conséquente, la croissance linéaire du nombre de vecteurs de support en fonction de la taille de la base n'est plus observée au bout d'un certain nombre d'exemples présentés. Il peut même y avoir une décroissante du nombre de vecteurs de support si de nouveaux exemples sont présentés.

Une méthode d'un principe différent est proposée dans [NGUYEN05]. Elle consiste à réaliser un apprentissage complet avec un SVM sur la base initiale, puis à fusionner itérativement deux vecteurs de support suivant un critère de distance. La localisation du nouveau vecteur de support créé est déterminé selon des considérations géométriques. Un critère d'arrêt est également proposé pour limiter le nombre de fusions.

5.2.2 Approche réduisant le nombre d'attributs utilisés

Nous présentons ici succinctement quelques méthodes utilisées avec les SVM pour réduire le nombre d'attributs. Outre le fait de réduire les riques de malédiction de la dimentionalité, ces méthodes ont également pour conséquence de réduire la complexité des fonctions de décision, comme évoqué précédemment. La plupart des approches actuelles sont de type emballeur [KOHAVI97A] ou de type rang [CHEN05B].

5.2.2.1 Méthode tabou

Dans [KORYCI04] la sélection des attributs est réalisée en utilisant la recherche avec tabous dans la construction d'un classificateur hiérarchique binaire. Pour chaque classificateur binaire utilisé dans ce schéma, la recherche avec tabous est utilisée pour choisir un sous-ensemble pertinent d'attributs. Les expérimentations réalisées soulignent que les performances globales du classificateur sont améliorées.

5.2.2.2 Méthode évolutionnaire

Ces approches à base d'algorithmes évolutionnaires on déjà été utilisées avec différents algorithmes d'apprentissage supervisé [DRÉO03]. Dans le cadre de l'utilisation des SVM, on peut citer les travaux de Fröhlich et al [FRÖH04]. La fonction objectif optimisée dans cette méthode correspond à une borne supérieure sur l'erreur de généralisation. Kuncheva et al [KUNCHE99] proposent une approche à base d'algorithmes génétiques dont le but est de réduire à la fois le nombre d'exemples et le nombre d'attributs utilisés par la régle PPV. Ils proposent un critère qui pénalise les solutions qui comportent un nombre important d'exemples et/ou d'attributs. La fonction à optimiser a la forme suivante:

$$q(\theta) = e_A(\mathrm{ppv}(\theta)) - \alpha \frac{m_\theta + n_\theta}{m + n} \tag{5.1}$$

avec θ la solution indiquant les exemples et attributs utilisés, $e_A(\mathrm{ppv}(\theta))$ l'erreur produite par la règle PPV en utilisant le modèle θ sur la base initiale d'apprentissage, m_θ le nombre d'exemples de la solution θ et n_θ le nombre d'attributs utilisés par la solution θ. α est un coefficient qui permet de règler l'importance de la pénalisation.

5.2.2.3 Méthode optimisant la marge géométrique

Chapelle et al [CHAPEL02] définissent une méthode à base de descente de gradient. La fonction noyau utilisée intègre la possibilité de changer les poids de chaque attribut. Dans cette méthode, les poids les plus faibles en fin d'apprentissage indiquent quels sont les attributs à ne pas sélectionner. Par contre, la valeur d'initialisation de ce type de méthode doit être choisie efficacement pour éviter qu'elle soit piégée dans un minimum local.

5.2.3 Influence des hyper-paramètres des SVM

Un choix efficace des hyper-paramètres des SVM permet de réduire l'erreur en généralisation. Lorsque le problème possède suffisamment d'exemples, la plupart des mé-

thodes réalisant la sélection des hyper-paramètres des SVM utilisent une base de test pour évaluer la qualité d'un modèle. Dans le cas contraire, le chapitre 4 a montré que les techniques de validation croisée sont appropriées pour estimer la qualité d'un modèle. Une autre possibilité est d'utiliser une des bornes supérieures sur l'erreur de généralisation, mais plusieurs résultats expérimentaux montrent que l'utilisation de ces bornes est délicate et que ce n'est pas forcément la meilleure façon de procéder pour la sélection des hyper-paramètres des SVM [AYAT05], en particulier lorsque les données sont fortement bruitées [LEE00, AYAT05].
L'équation (2.52) indique qu'un choix plus efficace de ces hyper-paramètres doit se traduire en moyenne par une réduction du nombre de vecteurs de support. Les résultats d'expérimentations dans [AYAT05] montrent que cet effet est effectivement observé. La sélection efficace des hyper-paramètres des SVM peut donc contribuer à produire des fonctions de décision de complexité réduite. Le chapitre 4 qui traite de techniques de validation croisée a montré l'importance de la sélection des valeurs des hyper-paramètres des SVM. Certaines figures de ce chapitre (*cf.* figures 5.9) ou du précédent (*cf.* figures 4.2) illustrent que les solutions optimales sont situées dans les régions où le nombre de vecteurs de support utilisé est faible, même si elles ne sont pas systématiquement celles qui comportent le minimum de vecteurs de support (*cf.* tableau A.5).

5.2.3.1 Méthode *grille de recherche*

C'est une méthode classique qui discrétise l'espace des modèles [CHANG01]. Cette méthode a déjà été abordée précédemment et nous avons signalé ses limites à des sélections de modèles qui exploitent peu de paramètres (≤ 2). Malgré ce défaut, elle permet de tracer des surfaces correspondant à l'évolution des capacités de généralisation d'un SVM en fonction des valeurs de ces deux hyper-paramètres. Cela permet de mieux appréhender ces évolutions, en plus d'une classique sélection de modèle. Un grand nombre d'expérimentations dans [LEE00] montrent que le paysage de l'erreur en généralisation, à travers l'utilisation de cette "grille", comporte des minima locaux, ce qui illustre que la seule sélection des hyper-paramètres des SVM est un problème difficile en soi. Ces surfaces montrent également que l'évolution de l'erreur en généralisation a des variations peu prononcées lorsque les variations des hyper-paramètres sont faibles. L'optimisation des capacités en généralisation ne nécessite donc pas la recherche de valeurs qui soient très précises pour ces hyper-paramètres, mais d'un ordre de grandeur qui ne soit pas trop grossier. Cela permet de réaliser une recherche efficace avec des *pas* de discrétisation relativement grands.

5.2.3.2 Méthode de recherche avec tabous

Dans cette méthode proposée par Cawley [CAWLEY01], le principe général de la recherche avec tabous est utilisé. Les mouvements possibles correspondent à ajouter ou soustraire ΔC et $\Delta\sigma$ respectivement à la constante de régularisation et à la largeur du noyau. Les valeurs des pas ΔC ou $\Delta\sigma$ sont diminuées ou augmentées suivant que le mouvement choisi a permis d'augmenter ou de diminuer les capacités en généralisation. La procédure s'arrête à partir d'un nombre fixé d'échecs relativement à l'augmentation des capacités de généralisation. Les quelques expérimentations réalisées dans [CAWLEY01]

montrent que la sélection de modèle est aussi efficace et plus rapide que la procédure *grid search*. Un des avantages de cette méthode est qu'elle est facile à étendre à des noyaux comportant plusieurs paramètres libres.

5.2.3.3 Méthode à base de descente de gradient

Dans [AYAT05] est défini un nouveau critère basé sur la notion de probabilité d'erreur mesurée sur un ensemble d'exemples différents (la base de test) de ceux exploités par la phase d'entraînement d'un SVM. Ce critère est utilisé pour minimiser l'erreur de généralisation d'un SVM. Leur méthode exploite le principe proposé par Platt [PLATT99B] (*cf.* section 4.1.5.4) pour estimer la probabilité de classification correcte de chaque exemple de la base de test. Ceci est réalisé à partir de la fonction de décision produite par un SVM pour un modèle donné (les hyper-paramètres de la fonction noyau). A partir de ces valeurs et des multiplicateurs de Lagrange de la fonction de décision est calculé le gradient de l'erreur estimée de généralisation pour les hyper-paramètres. Une technique de descente de gradient classique est ensuite utilisée pour choisir les valeurs optimales des hyper-paramètres de la fonction noyau. Leur méthode sert à l'optimisation des deux paramètres de la fonction noyau KMOD [AYAT02] et la largeur de bande d'un noyau gaussien. Les résultats expérimentaux montrent que si les valeurs d'initialisation des hyper-paramètres d'un noyau sont correctement choisies, la procédure de descente de gradient réduit à la fois la probabilité d'erreur et le nombre de vecteurs de support utilisés par la fonction de décision produite. Il est important de noter que leur méthode ne prend pas en compte l'optimisation de la valeur de C. Pour contourner ce problème, plusieurs optimisations sont réalisées pour différentes valeurs de C et la meilleure solution est conservée.

5.2.3.4 Méthode pour la constante de régularisation d'un SVM

Hastie et al [HASTIE04B] proposent une méthode élégante qui a un coût réduit pour choisir la valeur optimale de C, si c'est le seul paramètre à optimiser d'un SVM. Ils montrent expérimentalement avec un noyau gaussien que la valeur optimale de C dépend fortement de la valeur choisie pour la largeur de bande de ce noyau. Il est important de noter que si la valeur des autres hyper-paramètres (ceux de la fonction noyau en l'occurrence) ne sont pas correctement choisis, l'optimisation de C par cette méthode peut produire une fonction de décision très éloignée de l'optimum. Pour éviter ce problème une piste pourrait être de combiner la méthode de Ayat et al [AYAT05] avec celle-ci.

5.3 Nouvelles approches

5.3.1 Sélection d'exemples pertinents

L'objectif est de définir une méthode qui, à partir d'une base d'apprentissage, ne conserve que les exemples pertinents. Cette notion de pertinence est cependant difficile à définir. Il est possible de faire un parallèle avec la problématique de la sélection d'un sous-ensemble pertinent d'attributs (*cf.* section 5.3.1.1). Dans le cadre qui nous intéresse, il est plus judicieux de parler d'exemple "utile" à un objectif d'apprentissage fixé. Nous

utilisons donc le terme d'exemple "pertinent" dans la suite de ce chapitre en lui accordant la signification précédente "d'utile à". Comme dans le cadre de la sélection d'attributs, la notion de pertinence d'un exemple dans l'absolu, indépendamment du principe d'inférence utilisé et de sa mise en œuvre, n'a pas de sens pour l'apprentissage supervisé. Dans ce chapitre nous désignerons par l'expression "fonction de décision cohérente avec la base d'apprentissage" le fait que la fonction de décision ne réalise aucune erreur pour déterminer la classe de l'ensemble des exemples de cette base.

Plusieurs raisons font qu'un exemple peut être considéré comme non pertinent dans la production de fonctions de décision à partir d'exemples. Nous distinguons 3 types de *non pertinence* :

type 1 : Un exemple n'est pas pertinent, car il est redondant dans la base d'apprentissage. Il y a suffisamment d'exemples proches ou identiques dans la base pour que sa présence dans la base soit inutile.

type 2 : Un exemple n'est pas pertinent, car il n'a aucune influence, par sa présence dans la base, sur la fonction de décision produite par un algorithme d'apprentissage donné. Les exemples qui ne sont pas vecteurs de support pour l'algorithme des SVM ont ce statut. D'un point de vue plus général, un exemple n'est pas pertinent pour un algorithme d'apprentissage, s'il n'a pas d'influence sur les frontières de décision induites par le principe d'inférence réalisé par cet algorithme.

type 3 : Un exemple n'est pas pertinent, car les capacités de généralisation d'un algorithme d'apprentissage diminuent lorsqu'il est présent dans la base d'apprentissage.

La présence d'exemples correspondant aux deux premiers types n'a une influence notable que sur les temps d'apprentissage et de décision. Dans le cadre des SVM, les exemples de type 2 n'ont une influence que sur les temps d'entraînement, puisqu'il ne participeront pas à la fonction de décision. Les exemples de type 1 ont en plus une influence sur les temps de décision, en particulier lorsque la notion de marge molle est utilisée. En effet, si un exemple est dans la marge, ses voisins très proches, même s'ils sont redondants, ont une forte probabilité d'être également dans la marge.

Le troisième cas est plus délicat. Il correspond au fait que tous les exemples dans une base d'apprentissage ne sont pas représentatifs de la même manière du problème d'apprentissage. Plusieurs raisons peuvent être avancées pour expliquer ce fait. La première est qu'une base d'apprentissage n'est qu'un échantillon plus ou moins représentatif d'un problème donné. On suppose que les exemples sont issus d'un tirage *i.i.d*[5], mais ce n'est qu'approximativement vrai. Ces variations peuvent produire le fait qu'un exemple (ou ensemble d'exemples), bien que représentatif du problème, peut changer les frontières de décisions (par rapport à celles idéales) par sa seule présence dans la base d'apprentissage. Les algorithmes d'apprentissage ont pour objectif d'être le moins sensible possible à ce genre de problèmes. Les SVM, par l'optimisation d'un critère global, la marge géométrique, en sont une illustration. Il semble difficile dans ce cas de trouver un critère objectif permettant de supprimer ces exemples de la base sans une connaissance extérieure sur la nature du problème traité.
Mais d'autres raisons existent pour la présence d'exemples non pertinents de type 3 (cf. 2.2.3). Etant donné le caractère faillible des oracles et ou différentes imprécisions entre un

5. Tirage indépendant et identiquement distribué

objet et sa caractérisation, il peut exister dans les bases d'apprentissage des exemples non représentatifs d'un problème donné. Ceci peut être traduit par :

$$\exists\,(x,y) \in Z, \hat{y} \neq y : x = f_d(o), p(y|o) << p(\hat{y}|o) \tag{5.2}$$

avec $x = f_d(o)$ (*cf.* section 2.2) qui désigne que la description x est relative à l'objet o et que $\hat{y}$ est la vraie classe de l'objet o. Idéalement toute base d'apprentissage devrait avoir des exemples qui sont tous de la forme $(x,\hat{y})$. Si l'écart entre $p(\hat{y}|o)$ et $p(y|o)$ est faible, il semble difficile de découvrir ces exemples aberrants. Il est préférable de laisser un algorithme d'apprentissage tel que les SVM faire au mieux. Par contre, lorsque cet écart augmente, il semble logique que les exemples aberrants soient plus faciles à identifier. Le fait de pouvoir les éliminer[6]de la base d'apprentissage avant la phase d'entraînement d'un algorithme d'apprentissage ne peut être que bénifique. Ceci devrait permettre à un algorithme d'apprentissage d'être "moins perturbé" dans la production d'une hypothèse valide et donc d'augmenter ses capacités de généralisation. Des résultats dans le cadre de l'apprentissage actif montrent qu'un sous-ensemble d'exemples choisis pour leur représentativité à partir d'une base d'apprentissage initiale permet de produire des fonctions de décision plus performantes que l'application directe de l'algorithme d'apprentissage avec la base initiale [SCHOHN00].

Les méthodes qui sont proposées dans les sections suivantes ont pour objectif dans un premier temps de supprimer les données non pertinentes, essentiellement celles de type 1 et 2, puis d'améliorer les capacités de ces méthodes à supprimer les données non pertinentes de type 3.

5.3.1.1 Sélection d'attributs et pertinence

Pour mettre en évidence le parallèle entre la recherche d'un sous-ensemble pertinent d'exemples et un sous-ensemble pertinent d'attributs, nous présentons ici cette problèmatique en insistant sur le fait que la notion de pertinence des attributs n'est pas complétement indissociable de l'algorithme réalisant cette sélection.

La problématique

Il est courant d'avoir un grand nombre d'attributs comme descripteurs des objets. Cela conduit à rechercher une représentation plus compacte de la description des exemples qui utilise le moins d'attributs possibles, tout en conservant le même pouvoir expressif. Le premier intérêt de la réduction de la dimensionalité est que la majorité des algorithmes d'apprentissage supervisés ou non-supervisés sont sensibles à l'augmentation du nombre d'attributs. Cette sensibilité se traduit généralement par une dégradation des performances des algorithmes. Cependant, d'un point de vue théorique la probabilité d'erreur de classification ne doive pas augmenter quand on ajoute de nouveaux attributs, ce résultat théorique suppose que la densité de probabilité est parfaitement connue, or cette estimation de la densité est d'autant plus difficile que le nombre d'attributs augmente. Ce phénomène

6. Si la méthode utilisée pour la détection des données aberrantes à une certitude forte sur la vraie classe d'un objet, il semble préférable de changer sa classe dans la base en lui attribuant la vraie classe prédite, plutôt que de supprimer l'exemple de la base. Bien que cette perspective soit intéressante, nous ne chercherons pas à l'étudier dans ce document, laissant cette possibilité à de futurs travaux.

est appelé malédiction de la dimensionnalité. Un grand nombre d'expériences ont permis d'observer cet effet [JAIN97, SUN02] et plusieurs résultats théoriques [TRUNK79, VAPNIK98] permettent de le comprendre. Le second intérêt de la réduction de la dimensionalité est que l'augmentation du nombre d'attributs conduit à une augmentation à la fois des temps de traitements, de l'occupation en mémoire des données à traiter et de l'espace de stockage dans des bases de données. L'augmentation des temps de traitement a essentiellement deux causes: 1) l'obtention des attributs peut nécessiter une phase de prétraitement qui a un coût calculatoire important, 2) la phase d'entraînement des classificateurs a une durée qui augmente linéairement avec le nombre d'attributs. Une fonction de décision prendra donc d'autant plus de temps pour déterminer la classe d'un objet qu'elle utilisera un grand nombre d'attributs. La réduction du nombre d'attributs utilisés, tout en conservant au maximum l'information significative, est donc une étape indispensable pour améliorer les temps de traitement et réduire les effets néfastes de la malédiction de la dimensionalité. Pour cela un grand nombre d'algorithmes ont été proposés pour réaliser cette réduction [LIU98].

Le problème de la recherche d'un sous-ensemble pertinent d'attributs (ou variables) peut avoir deux formulations suivant que le nombre d'attributs à sélectionner est fixé ou ne l'est pas [JAIN00, KOHAVI97B, GUYON03]. Dans les deux cas, la sélection nécessite la définition d'un critère de qualité q caractérisant la pertinence de ce sous-ensemble. Sans perte de généralité, nous supposerons que la valeur de q croît avec l'augmentation de la qualité et que le cas où le nombre d'attributs du sous-ensemble est fixé est un cas particulier de celui où il est libre. Notons $N = \{1, \ldots, n\}$, l'ensemble contenant tous les indices des attributs exploitables, $N_s \subset N$, un sous-ensemble contenant les indices des attributs sélectionnés et $q(N_s)$ la valeur du critère de qualité q lorsque seulement les attributs N_s des exemples sont utilisés. Dans le cas général où le nombre d'attributs à sélectionner n'est pas fixé, il est possible d'envisager que plusieurs solutions puissent avoir la même qualité maximale, dans ce cas la solution utilisant le moins d'attributs doit être sélectionnée (principe de parcinomie), si plusieurs solutions existent toujours, elles sont alors considérées comme autant de solutions optimales. Les définitions des deux types de sélection d'attributs sont les suivantes :

Définition (Sélection du meilleur sous-ensemble d'attributs) Soit N_s^* un ensemble d'attributs, N_s^* est optimal ssi $N_s^* = \underset{N_s \subset N \wedge q(N_s)=q^*}{\arg\min} (|N_s|)$ avec $q^* = \max_{N_s \subset N} (q(N_s))$

Le problème est alors de définir le critère q qui mesure la pertinence d'un sous-ensemble d'attributs afin de sélectionner N_s[7]. Idéalement, on souhaiterait pouvoir mesurer la pertinence d'un sous-ensemble d'attributs à partir de la pertinence de chaque attribut. Nous allons voir que cela n'est pas possible et que la notion de pertinence des attributs est dans l'absolu un concept difficile à définir.

Notion de pertinence d'attributs selon Kohavi et al

Kohavi et al [KOHAVI97B] explorent plusieurs définitions théoriques de cette notion de pertinence, proposées par différents auteurs dans les années 1990. Elles se situent dans un cadre probabiliste et elles supposent que la distribution $\mathcal{Z}$, correspondant aux variables

7. Un autre problème est la complexité de la méthode réalisant la sélection de N_s à partir de cette mesure, mais nous ne nous intéresserons pas dans cette section à ce problème, même si en pratique il est essentiel.

d'entrées X et à la variable cible Y, est parfaitement connue. Une définition possible, sous ces conditions, est qu'un attribut X_i est pertinent s'il satisfait la relation suivante:

$$\exists(x,y) \in \mathcal{Z} : p(Y = y|X_i = x) \neq p(Y = y) \tag{5.3}$$

avec $p(X_i = x) > 0$. Un attribut est donc pertinent si sa connaissance change la probabilité d'observer y et ceci indépendamment de la connaissance des autres attributs. La détermination des attributs pertinents devient donc triviale relativement à cette définition, mais n'est pas sans poser de problème pour la sélection du sous-ensemble N_s^*. Kohavi et al [KOHAVI97B] relèvent plusieurs contradictions, nous n'en citerons que deux relatives à (5.3):

- Supposons que deux variables X_i et X_j sont totalement corrélées. Elles sont toutes les deux pertinentes, si l'une d'elles satisfait (5.3). Pour autant, il est logique de n'utiliser qu'une seule de ces deux variables pour construire N_s^*.
- Supposons maintenant, que $X_1 \in [0,1]$ est une variable aléatoire correspondant à une distribution uniforme indépendante des valeurs de X_2 et Y. Définissons la variable X_2, conditionnellement à X_1 et Y, de la façon suivante : $X_2 = \begin{cases} 1 - X_1 & \text{si } Y = 1 \\ X_1 & \text{si } Y = 0 \end{cases}$
  On peut remarquer que ces deux variables, prises individuellement, ne sont pas pertinentes par rapport à la définition précédente. Pour autant, la connaissance globale de ces deux variables permet d'inférer avec une probabilité d'erreur nulle la valeur de Y.

Kohavi et al relevant de telles contradictions dans ces définitions (voir [KOHAVI97B] pour plus de détails) arrivent à la conclusion qu'il n'est pas possible de classer les attributs en totalement pertinent ou non pertinent même lorsque la distribution $\mathcal{Z}$ est parfaitement connue. Ils proposent alors d'utiliser la notion de degré de pertinence : fortement, faiblement ou non pertinent. Ces trois notions peuvent être traduites de la façon suivante (la distribution $\mathcal{Z}$ est toujours considérée comme parfaitement connue):

- Un attribut est fortement pertinent, si l'ajout de cet attribut à un ensemble quelconque N_i d'autres attributs change la probabilité associée ($\forall N_i : P(Y,N_i|X_i) \neq P(Y,N_i)$).
- Un attribut est faiblement pertinent, s'il n'est pas fortement pertinent et qu'il existe au moins un sous-ensemble non vide d'attributs pour lequel l'ajout de cet attribut change la probabilité associée ($\exists N_i : P(Y,N_i|X_i) \neq P(Y,N_i)$).
- Un attribut est non pertinent s'il n'est ni fortement ni faiblement pertinent.

Par rapport à cette définition, il est évident que les attributs fortements pertinents doivent être sélectionnés dans N_s^* et ceux non pertinents doivent être rejetés, mais pour les faiblements pertinents c'est plus problématique. En supposant que le critère q ne prenne pas en compte la capacité de généralisation, le choix des attributs faiblement pertinents à retenir dans N_s est un problème combinatoire complexe et cela même lorsque $\mathcal{Z}$ est parfaitement connue. De plus d'un point de vue pratique beaucoup d'attributs sont de type faiblement pertinent. Nous considérons par la suite que tous les attributs utilisés sont de type faiblement pertinent (même si parmi ceux-ci certains peuvent être plus informatif que d'autres).

Attributs utiles à un algorithme d'apprentissage

Dans la majorité des cas, la distribution $\mathcal{Z}$ est inconnue et le problème de sélection des attributs pertinents en devient encore plus difficile. C'est typiquement le cas lorsque la connaissance $\mathcal{Z}$ n'est accessible uniquement à travers un ensemble réduit d'exemples Z. A partir de la donnée Z, d'une mesure de performance M et d'un algorithme d'apprentissage A, l'objectif est alors de sélectionner un sous-ensemble d'attributs N_s tel que la mesure de la performance de l'algorithme d'apprentissage à partir de ces attributs soit optimale. Notons $Z \downarrow N_s$ la sélection des N_s attributs présents dans Z. Plusieurs approches existent pour réaliser cette recherche. Elles sont généralement regroupées en trois catégories :

1. Emballeur (*wrapper*) : Cette approche utilise directement la mesure des performances de l'algorithme A choisi comme critère de qualité (*i.e.* $q(N_s) = M(Z,A(Z \downarrow N_s))$),
2. Filtre (*filter*) : Cette approche utilise une mesure qui ne dépend pas des performances de l'algorithme d'apprentissage A avec les données $Z \downarrow N_s$ (*i.e.* $q(N_s) = M(Z \downarrow N_s)$), mais qui est supposé représentatif de la performance de l'algorithme A (*i.e.* $M(N_{s_1}) > M(N_{s_2}) \rightarrow M(Z,A(Z \downarrow N_{s_1})) > M(Z,A(Z \downarrow N_{s_2}))$),
3. Incorporée (*embedded*) : Cette approche n'utilise pas un critère extérieur à l'algorithme A pour réaliser la sélection, mais c'est l'algorithme A qui pendant sa phase d'apprentissage réalise cette sélection (*i.e.* $N_s^* \leftarrow A(Z, M)$).

Au regard de ces trois types d'approche, la sélection produite par une approche *filtre* a un grand avantage car elle est réalisée en amont de l'algorithme d'apprentissage. S'il est possible de réaliser un algorithme de type *filtre* qui est capable de sélectionner un sous-ensemble optimal d'attributs pertinents tel que défini précédemment, et ceci indépendamment de l'algorithme d'apprentissage utilisé, le problème d'apprentissage sera grandement simplifié. Tsamardinos et al [TSAMAR03] montrent que cette recherche est vaine, car la définition d'un ensemble optimal d'attributs N_s^* est dépendante du choix de A et de M et ne correspondra pas à la définition d'un sous-ensemble optimal d'attributs pertinents tels que défini dans un cadre purement probabiliste. Kohavi et al [KOHAVI97B] donnent des exemples où le simple fait d'ajouter un attribut pertinent (relativement à une distribution connue) peut réduire de façon significative les performances d'un algorithme d'apprentissage, alors que paradoxalement l'ajout d'un attribut non pertinent peut améliorer ces performances, ce qui illustre à nouveau la dépendance à un algorithme d'apprentissage. Blum et al [BLUM97] proposent différentes définitions de la notion d'attributs pertinents ou d'ensemble d'attributs pertinents suivant l'objectif de la méthode de sélection, en particulier une nommée *Incremental usefulness* qui définit la pertinence d'un attribut i relativement à un ensemble d'attributs N_s ($i \notin N_s$) comme la possibilité d'augmenter les performances en généralisation d'un algorithme d'apprentissage particulier A lorsque l'ensemble d'attributs $\{i\} \cup N_s$ est utilisé au lieu de l'ensemble d'attributs N_s. Guyon et al [GUYON03] utilise la notion d'attributs pertinents comme signifiant que le ou les attributs sont "utiles" à l'augmentation des performances d'un algorithme d'apprentissage A particulier (la définition *Incremental usefulness* à déjà cette connotation).

Le problème de la sélection d'attributs peut alors être reformulé comme la recherche d'un ensemble d'attributs utiles à la réalisation d'un processus d'inférence particulier afin qu'il soit de qualité maximum, plutôt que la recherche d'un ensemble d'attributs pertinents par rapport à un processus d'inférence idéal. Les approches de type *emballeur* et *incorporée* sont les plus succeptibles de réaliser la sélection de N_s^*, mais elles ont généralement un

coût calculatoire important, en particulier les approches *emballeur* car elles nécessitent de mutliples phases d'entraînement. Les approches *filtre* ne sont pas pour autant à banir, elles permettent généralement de choisir un ensemble N_s efficace avec un coût calculatoire plus faible , même s'il n'est pas optimal, pour un grand nombre d'algorithmes d'apprentissage. De plus la connaissance des avantages et inconvénients d'un algorithme d'apprentissage permet de choisir une méthode de type *filter* qui lui est plus appropriée.

5.3.1.2 Ensemble d'exemples pertinents et règle PPV

Comme précisé précédemment, l'objectif est de définir une méthode qui ne conserve que les exemples pertinents, dans une base donnée, pour la production d'une fonction de décision de qualité. Ces exemples pertinents sont ensuite utilisés pour réaliser la construction d'une fonction de décision efficace avec l'algorithme des SVM. La pertinence d'un ensemble d'exemples est mesurée par l'efficacité de cet ensemble à être cohérent avec la base d'apprentissage tout en ayant le moins d'exemples possible. La règle PPV est utilisée pour mesurer cette cohérence. Si l'on réalise un parallèle avec la sélection d'attributs, la règle PPV est une approche de type *filtre* relativement aux SVM pour la sélection d'exemples pertinents. Le principe d'inférence mis en œuvre correspond donc à la construction d'un schéma de compression efficace avec la règle PPV. La section 2.4.1.4 a montré que cela correspondait à réaliser un principe d'inductif basé sur la minimisation du risque structurel. Comme le noyau utilisé est un noyau gaussien, les mesures de distance sont réalisées dans l'espace initial. La raison de ce choix est que la relation suivante est vraie lorsque la fonction noyau utilisée est un noyau Gaussien:

$$\forall i,j,k : d(\mathbf{x}_i,\mathbf{x}_j) > d(\mathbf{x}_i,\mathbf{x}_k) \Leftrightarrow d(\Phi(\mathbf{x}_i),\Phi(\mathbf{x}_j)) > d(\Phi(\mathbf{x}_i),\Phi(\mathbf{x}_k)) \tag{5.4}$$

La règle PPV donne donc le même résultat dans l'espace inititial et dans l'espace de redescription avec un noyau Gaussien. L'extension de notre méthode à d'autres noyaux peut par contre être réalisée grâce à l'utilisation de l'expression (4.58)[8]. Comme la notion de pertinence (ou non pertinence) individuelle d'un exemple est multiple et dépendante des autres exemples connus, nous allons définir plusieurs critères qui mesurent l'importance de la pertinence d'un exemple relativement à une base d'exemples. Ces mesures seront basées sur l'utilisation de marges de confiance (*cf.* section 5.3.1.3).

Avant d'aborder la définition de ces critères, commençons par donner le synopsis général de notre méthode. Notons Z_m^A une base d'apprentissage initiale qui comporte m exemples et Z_i^p la base qui contient les i exemples les plus pertinents (le p en exposant est une contraction de pertinence). Supposons qu'une mesure de pertinence q_p permet de choisir le $i+1^{\text{ème}}$ exemple le plus pertinent dans Z_m^A sachant que Z_i^p contient les i précédents. Le synopsis général de notre méthode est le suivant :

1. $Z_0^p = \emptyset$ et $i = 0$,
2. Sélectionner l'exemple j dont la valeur de $q_p(z_j \in Z_m^A | Z_i^p)$ est maximale,
3. Ajouter z_j à Z_i^p pour créer Z_{i+1}^p et incrémenter i,
4. Répéter les étapes 2 et 3 jusqu'à ce que la règle *1-ppv* avec Z_i^p soit cohérente avec Z_m^A.
5. Retourner $Z^p = Z_i^p$

8. C'est d'ailleurs l'utilisation de cette expression qui permette de justifier la relation (5.4) .

Le schéma de compression correspondant est d'autant plus performant que la la taille de Z^p est faible. Il garantit par la même occasion une borne supérieure efficace sur l'erreur en généralisation (*cf.* expression 2.51). L'efficacité de la méthode dépend de la définition de q_p et de sa relation avec Z_i^p.

La définition de notre critère de pertinence q_p relativement à l'utilisation de la règle PPV utilise deux notions. L'une est directement dépendante de Z_i^p et l'autre en est indépendante. Ces deux notions sont:

1. Nous considérons que tout exemple qui est dans la base d'apprentissage Z_m^A et qui est correctement classé par la règle PPV à partir de Z_i^p n'est pas pertinent relativement à cette règle, car cet exemple est redondant pour cette règle. Un sous-ensemble d'exemples pertinents candidats, que nous notons Z_i^c (le c en exposant signifiant candidat), peut être construit à partir de Z_i^p et de la règle PPV. Il est défini comme:
$$Z_i^c = \{z | z = (\mathbf{x},y) \in Z_m^A \wedge z \notin Z_i^p \wedge \text{1-ppv}(\mathbf{x},Z_i^p) \neq y\} \tag{5.5}$$
avec 1-ppv$(\mathbf{x},Z_i^p)$ la classe de l'exemple le plus proche de $\mathbf{x}$ dans l'ensemble Z_i^p. Cette définition permet de ne pas sélectionner les exemples non pertinents de type 1 et 2 relativement à la règle PPV.
2. Nous utilisons une mesure globale de pertinence q_p, dont les détails seront explicités ultérieurement (*cf.* section 5.3.1.3). Elle est soit mesurée pour un exemple z donné, soit mesurée pour un couple d'exemples (z_1,z_2) donné à partir de la base initiale Z_m^A et elle est notée respectivement $q_p(z,Z_m^A)$ et $q_p((z_1,z_2),Z_m^A)$. Dans le cas où la mesure est réalisée à partir d'un couple d'exemples, les deux exemples formant ce couple sont obligatoirement de classes différentes. Dans le cas où la mesure concerne un couple d'exemples, c'est le couple qui est ajouté et non un simple exemple[9]. Au-delà de ces considérations techniques (ajouter un ou deux exemples), cela ne retire rien au principe général du synopsis ci-dessus.

L'utilité de ces deux notions est de permettre de représenter que la pertinence globale d'un exemple (ou d'un couple d'exemples) peut être remise en cause s'il est redondant avec l'ensemble des exemples les plus pertinents (relativement à la mesure globale) déjà sélectionnés. La formulation de pertinence relative (à Z_i^p) pour un exemple à partir des deux notions précédentes est la suivante:
$$q_p(z,Z_m^A,Z_i^p) = \begin{cases} q_p(z,Z_m^A) \text{ si } z \in Z_i^c \\ -\infty \text{ sinon} \end{cases} \tag{5.6}$$

Dans le cas où la mesure de pertinence relative est réalisée à partir d'un couple d'exemples, elle devient:
$$q_p((z_1,z_2),Z_m^A,Z_i^p) = \begin{cases} q_p((z_1,z_2),Z_m^A) \text{ si } z_1 \in Z_i^c \vee z_2 \in Z_i^c \\ -\infty \text{ sinon} \end{cases} \tag{5.7}$$

Sélectionner le prochain exemple z_j (ou couple d'exemples) le plus pertinent dans une base d'apprentissage en tenant compte des précédents (*cf. étape 2 du synopsis*) est maintenant clairement défini comme :
$$z_j = \arg\max_{z \in Z_m^A \wedge Z_i^p} q_p(z,Z_m^A,Z_i^p) \tag{5.8}$$

9. Dans le cas où un des exemples d'un couple est déjà présent dans Z_i^p, c'est seulement celui qui n'y est pas qui est ajouté, ceci afin d'éviter de créer des doublons inutiles.

5.3.1.3 Quatre critères de pertinence

Dans la méthode que nous proposons, il reste à définir la notion de pertinence globale d'un exemple $q_p(z,Z_m^A)$ ou d'un couple d'exemples $q_p((z_1,z_2),Z_m^A)$. L'objectif principal est de produire un schéma de compression efficace. Pour arriver à ce résultat, il est important d'ajouter à chaque étape de notre méthode un exemple (ou couple d'exemples) qui permettra de classer correctement un maximum d'exemples dans la base afin de garantir une convergence rapide de la méthode. Nous proposons quatre critères pour réaliser cette estimation de pertinence globale. Deux sont basés sur une mesure qui fait intervenir des couples d'exemples et les deux autres sont basés sur une mesure estimée pour chaque exemple. Pour arriver à ce résultat, il est important que la sélection soit basée sur des mesures de marge de confiance. Trois précédentes études ont particulièrement attiré notre attention: [KARACA03] (voir section 2.4.1.4), [GILAD-04] et [SHIN02] pour la définition de ces critères.

Critères 1 et 2 basés sur les couples d'exemples

Dans [KARACA03], l'idée proposée est dans un premier temps de calculer toutes les distances entre couples d'exemples de classes différentes, puis d'ajouter itérativement ces couples d'exemples dans une base initialement vide en commençant par ceux les plus proches. L'algorithme s'arrête lorsque la base produite ainsi est cohérente avec les données en utilisant la règle PPV. Nous avons vu que cela correspondait à contrôler la VC dimension de l'espace d'hypothèse en section 2.4.1.4. Par rapport à notre notion de pertinence, cela correspond essentiellement à réduire le nombre d'exemples de type 2, en particulier pour les SVM. Cette notion de marge peut être utilisée pour définir notre premier critère $q_p((z_1,z_2),Z_m^A)$. Il y a par contre une différence notable entre la méthode de Karaçali et al et notre synopsis général : l'utilisation d'un ensemble candidat Z_i^c. Notre méthode est donc à même d'utiliser moins d'exemples que celle de Karaçali et al pour être cohérente avec la base de donnée initiale.
L'utilisation de ce critère correspond donc à considérer comme globalement pertinent les exemples proches de la frontière de décision. Pour les bases de données comportant des données faiblement bruitées, ce critère est un choix intéressant. Par contre lorsque la base contient des exemples non pertinents de type 3, ils seront intégrés dans Z_i^p dès les premières itérations de l'algorithme. Pour éviter ce problème , il est possible de considérer l'utilisation inverse de ce critère, à savoir commencer par les exemples de classes différentes les plus éloignés. Utilisée telle quelle dans la méthode de Karaçali et al, cette façon de procéder serait non appropriée, car le schéma de compression ne serait pas efficace. Par contre, notre méthode ne considère pas les exemples bien classés par la règle PPV comme candidats. L'ajout d'exemples éloignés de la frontière de décision permet alors de classer correctement un grand nombre d'exemples qui sont eux aussi éloignés de cette frontière. La méthode se rapprochera donc rapidement de la frontière de décision lors de l'ajout de nouveaux couples grâce à l'utilisation de Z_i^c. Il y aura forcément quelques exemples, ajoutés pendant les premières itérations de la méthode, qui seront trop éloignés de la frontière de décision pour être utiles aux SVM, mais la proportion du nombre d'exemples de ce type reste assez faible alors cela ne sera pas problèmatique[10]. La méthode ajoutera par

10. De plus, ils seront ignorés par la suite lors de la construction de la fonction de décision lorsque l'algorithme des SVM est utilisé. Ceci aura juste pour effet d'augmenter les temps d'apprentissage.

contre les exemples aberrants, s'ils existent, dans les dernières itérations[11], ce qui laisse la possibilité de les éliminer en proposant un critère d'arrêt prématuré pour notre synopsis global. A partir de ces deux approches nous avons défini le critère 1 comme :

$$q_p((z_1,z_2),Z_m^A)[\gamma \nearrow] = -d(\mathbf{x}_1,\mathbf{x}_2) \tag{5.9}$$

et le critère 2 comme :

$$q_p((z_1,z_2),Z_m^A)[\gamma \searrow] = d(\mathbf{x}_1,\mathbf{x}_2) \tag{5.10}$$

$\gamma \nearrow$ et $\gamma \searrow$ signifiant respectivement que la mesure de l'importance de la pertinence, indépendamment de Z_i^p, est basée sur une approximation de la notion de marge géométrique croissante ou décroissante. Nous rappelons que les exemples $\mathbf{x}_1$ et $\mathbf{x}_2$ sont forcément de classes différentes.

Critères 3 et 4 basés sur des singletons d'exemples

Le critère 3 a pour inspiration les travaux de Shin et al [SHIN02] et permet de mesurer autrement la pertinence d'un exemple. Shin et al utilisent la règle *k-ppv* pour définir une mesure de probabilité locale de chaque classe. Elle correspond, pour une valeur de k fixé, à déterminer pour un exemple donné ses k plus proches voisins, puis à calculer la probabilité de chaque classe à partir de celles de ces k voisins. Cette mesure caractérise la confiance dans la classe de l'exemple considéré. En effet, si la probabilité estimée est élevée pour la classe correspondant à celle de l'exemple et que la valeur de k est grande, alors la probabilité que l'exemple ne soit pas de cette classe est faible. Cela permet de caractériser en particulier un risque faible de non pertinence de type 3. Pour obtenir une mesure qui corresponde à une marge d'hypothèse et ne pas être dépendant du fait de devoir fixer la valeur de k, nous considérons qu'un exemple est d'autant plus pertinent que les k voisins les plus proches sont de la même classe que l'exemple considéré. La valeur maximale de k définit la valeur de la marge. Cette valeur caractérise la confiance dans la classe de l'exemple et par la même sa pertinence à être utilisé dans la règle PPV. Elle caractérise également le fait que l'exemple est dans une zone à forte densité d'exemples de la même classe. Soit nombre-ppv$_{\text{mc}}(z,Z)$ la fonction qui retourne le nombre maximum de voisins dans Z qui soient à la fois les plus proches de z et tous de la même classe que z. Ce dernier critère de marge d'hypothèse est alors le suivant :

$$q_p((z),Z_m^A)[\gamma_{\odot} \searrow] = \text{ nombre-ppv}_{\text{mc}}(z,Z_m^A) \tag{5.11}$$

Le symbole $\odot$ illustre que la pertinence d'un exemple relativement à une base d'apprentissage est d'autant plus grande qu'il est entouré d'un grand nombre d'exemples (ses plus proches voisins) de la même classe que lui.

La définition du critère 4 a pour inspiration une mesure de marge d'hypothèse, similaire à (2.61), utilisée dans une méthode de sélection d'attributs [GILAD-04]. Pour un exemple pris aléatoirement dans une base d'apprentissage, la méthode recherche l'exemple qui lui est le plus proche avec la même classe ainsi que l'exemple qui lui est le plus proche avec une classe différente. L'importance de chaque attribut est augmentée ou diminuée

11. Il est aussi possible d'avoir des exemples aberrants éloignés de la frontière de décision, mais il est logique de penser que, pour la majorité des bases d'apprentissage, la probabilité qu'un exemple z soit aberrant diminue avec la distance qui le sépare de la frontière de décision.

suivant que pour cet attribut la règle PPV réalise une classification correcte ou incorrecte. Nous reprenons ce principe sans définir une procédure stochastique et le transposons à la notion d'exemple pertinent. L'idée directrice pour la définition de ce nouveau critère est que les exemples qui permettent de réaliser une classification correcte à partir de la règle PPV ont des valeurs de marge d'hypothèse importantes et ceux conduisant à produire des erreurs ont des valeurs de marge faibles ou négatives. Nous utilisons la marge d'hypothèse (2.61) que nous rappelons ici:

$$\gamma_h(z) = \gamma_h(\mathbf{x},y) = \frac{1}{2}\left(\|\mathbf{x}_i - \mathbf{x}\| - \|\mathbf{x}_j - \mathbf{x}\|\right)$$

avec $\mathbf{x}_i$ et $\mathbf{x}_j$ les vecteurs caractérisant respectivement l'exemple z_i le plus proche z de classe différente et l'exemple z_j le plus proche z de même classe.

Définissons deux propriétés qui sont utiles à la sélection des exemples z_i et z_j pour le calcul de la marge d'hypothèses $\gamma_h(z)$:

1. Le plus proche voisin (PPV) de la même classe (mc) d'un exemple donné :

$$ppv_{\text{mc}}(\mathbf{x},y) = \underset{\substack{(\mathbf{x}',y')\in Z_m^A \\ y'=y \,\wedge\, \mathbf{x}'\neq\mathbf{x}}}{\arg\min} \; (d(\mathbf{x},\mathbf{x}')) \tag{5.12}$$

2. Le plus proche voisin d'une classe différente (diff) d'un exemple donné :

$$ppv_{\text{diff}}(\mathbf{x},y) = \underset{\substack{(\mathbf{x}',y')\in Z_m^A \\ y'\neq y}}{\arg\min} \; (d(\mathbf{x},\mathbf{x}')) \tag{5.13}$$

A partir de ces deux notions, nous définissons l'ensemble des marges d'hypothèse relatives à un exemple z:

$$\Gamma_h(z) = \{\gamma_h(z') : z' \in Z_m^A, ppv_{\text{mc}}(z') = z \vee \; ppv_{\text{diff}}(z') = z\} \tag{5.14}$$

Un exemple z de la base d'apprentissage est d'autant plus pertinent que les marges d'hypothèse dans $\Gamma_h(z)$ sont élevées. La figure 5.1 illustre l'application du critère (5.14). Plus les valeurs dans $\Gamma_h(z)$ seront élevées et positives, et plus cet exemple sera un bon candidat pour son exploitation par la règle PPV. Lorsque toutes les valeurs de $\Gamma_h(z)$ sont négatives, il semble préférable de considérer cet exemple comme un exemple non pertinent de type 3 et donc de ne pas prendre cet exemple dans la base Z^p même si cela conduit à ne pas pouvoir produire une règle *1-ppv* totalement cohérente avec la base d'apprentissage. Nous reviendrons sur cette possibilité dans la section 5.3.1.5. Le critère que nous avons utilisé pour déterminer la pertinence d'un exemple à partir de $\Gamma_h(z)$ est la valeur moyenne de ces marges [12]. Ce qui nous donne le critère de pertinence suivant:

$$q_p(z,Z_m^A)[\gamma_{\bar{\Gamma}} \searrow] = \bar{\Gamma}_h(z) \tag{5.15}$$

avec $\bar{\Gamma}_h(z)$ désignant la valeur moyenne de $\Gamma_h(z)$ (par convention la valeur moyenne d'un ensemble vide est égale 0). Le symbole $\gamma_{\bar{\Gamma}} \searrow$ est utilisé pour représenter le fait que le critère est basé sur l'utilisation d'une marge d'hypothèse. Plus la valeur de ce critère est faible et plus la pertinence est considérée comme faible.

Le tableau 5.1 résume les informations sur les quatre critères de confiance utilisés pour la sélection des exemples les plus pertinents.

12. C'est une façon simple et directe d'utiliser l'ensemble de ces valeurs, même si nous pensons qu'il est possible d'en avoir une utilisation plus fine. Nous laissons cette possibilité pour de futurs travaux.

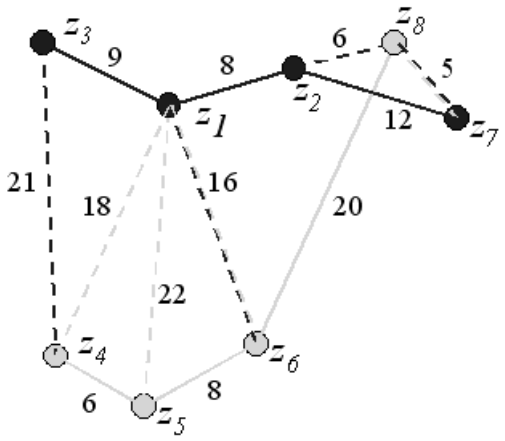


FIG. 5.1 – *Illustration de l'application du critère 3 avec une base simple de 8 exemples (les couleurs noire et grise représentent les classes des exemples). Les traits en plein relient un exemple z à l'exemple de la même classe le plus proche de lui. Les traits en pointillés relient un exemple z à l'exemple le plus proche de classe différente. La longueur de ces traits, indiquée par un chiffre, correspond aux valeurs de $ppv_{\rm mc}$ et $ppv_{\rm diff}$ pour ces exemples. Pour les exemples de z_1 à z_8 les valeurs de γ_h sont respectivement de 4 ; -1 ; 6 ; 6 ; 8 ; 4 ; -3,5 ; -7,5. Prenons pour exemple le calcul de $\bar{\Gamma}_h(z_1)$. Les exemples qui ont pour plus proche voisin de même classe z_1 sont z_2 et z_3 ($ppv_{\rm mc}(z') = z_1$ dans l'expression 5.14). Les exemples qui ont un plus proche voisin de classe différente z_1 sont z_4, z_5 et z_6 ($ppv_{\rm diff}(z') = z_1$ dans l'expression 5.14). On en déduit que $\Gamma_h(z_1) = \{-1; 6; 6; 6; 8; 4\}$ et $\bar{\Gamma}_h(z_1) =$ 4,6. De la même façon on obtient que $\bar{\Gamma}_h(z_8) =$ -2,25. Par rapport à ce critère z_1 et globalement plus pertinent que z_8.*

Numéro	Symbole	q_p
1	$\gamma \nearrow$	$-d(\mathbf{x}_1,\mathbf{x}_2)$ avec $y_1 \neq y_2$
2	$\gamma \searrow$	$d(\mathbf{x}_1,\mathbf{x}_2)$ avec $y_1 \neq y_2$
3	$\gamma_\odot \searrow$	nombre-ppv$_{\rm mc}(z, Z_m^A)$
4	$\gamma_{\bar{\Gamma}} \searrow$	$\bar{\Gamma}_h(z)$

TAB. 5.1 – *Résumé d'informations sur les 4 critères de confiance. La désignation par un symbole de ces 4 critères est utilisée dans les tableaux de résultats expérimentaux de ce chapitre et la désignation par numéro est utilisée dans les légendes des graphiques de ce chapitre.*

5.3.1.4 Expérimentation des 4 critères de pertinence

Nous avons testé les quatre critères de pertinence (5.9), (5.10), (5.15) et (5.11) avec notre méthode de sélection d'un sous-ensemble réduit d'exemples pertinents. Nous avons utilisé plusieurs bases d'entraînement dont les caractéristiques sont décrites dans les tableaux A.1 et A.3. Comme l'implémentation de notre méthode n'utilise pas de techniques d'accélération de la règle PPV qui permettent la construction d'une structure de données adaptée à une recherche rapide des plus proches voisins, sa complexité est relativement importante, même si elle reste polynomiale. Cela nous limite à des bases de tailles ré-

duites ($m < 1000$). Malgré tout, l'ensemble de ces tests est suffisant pour observer le comportement de notre méthode avec ces différents critères.
Les colonnes $Z\%$ et $VS\%$ des tableaux de cette section (et des sections suivantes) représentent respectivement le pourcentage d'exemples utilisé par la règle PPV et le pourcentage de vecteurs de support dans la fonction de décision d'un SVM par rapport au nombre d'exemples dans la base d'apprentissage.

Les résultats décrits par les tableaux 5.2 et 5.3 montrent que pour la règle PPV la réduction du nombre d'exemples peut être très importante, sans pour autant réduire significativement les performances en généralisation (**Pendigits-2c** par exemple). Pour certaines bases de données, cette réduction s'accompagne d'amélioration des performances en généralisation (**QIjpeg2M** $\approx -5\%$, **Bupa** $\approx -5\%$, **Ionosphere** $\approx -2\%$, **Pima** $\approx -3\%$).

		PPV		SVM	
Base	critère	e_V	Z %	e_V	VS %
QIjpep2M	$\gamma \searrow$	*30,63*%	32,76%	21,62%	31,03%
QIjpep2M	$\gamma_{\odot} \searrow$	29,73%	**34,48**%	22,52%	32,76%
QIjpep2M	$\gamma_{\bar{\Gamma}} \searrow$	**21,62** %	37,93%	19,82%	34,48%
QIjpep2M	$\gamma \nearrow$	**20,72**%	**31,9**%	23,42%	31,03%
QIjpep2M-2c	$\gamma \searrow$	0,9009%	**11,21**%	1,802%	7,759%
QIjpep2M-2c	$\gamma_{\odot} \searrow$	0,9009%	**11,21**%	1,802%	8,621%
QIjpep2M-2c	$\gamma_{\bar{\Gamma}} \searrow$	0,9009%	**12,07**%	1,802%	8,621%
QIjpep2M-2c	$\gamma \nearrow$	0,9009%	**6,034**%	0,9009%	6,034%
Iris	$\gamma \searrow$	0%	**16,67**%	0%	**12,5**%
Iris	$\gamma_{\odot} \searrow$	*3,333*%	15,83%	0%	13,33%
Iris	$\gamma_{\bar{\Gamma}} \searrow$	0%	**15,83**%	0%	**14,17**%
Iris	$\gamma \nearrow$	*3,333*%	11,67%	3,333%	10,83%
Iris-2c	$\gamma \searrow$	0%	**2,5**%	0%	**2,5**%
Iris-2c	$\gamma_{\odot} \searrow$	0%	**2,5**%	0%	**2,5**%
Iris-2c	$\gamma_{\bar{\Gamma}} \searrow$	0%	**2,5**%	0%	**2,5**%
Iris-2c	$\gamma \nearrow$	0%	**1,667**%	0%	**1,667**%
Wine	$\gamma \searrow$	*11,76*%	22,92%	0%	17,36%
Wine	$\gamma_{\odot} \searrow$	*11,76*%	20,14%	0%	**14,58**%
Wine	$\gamma_{\bar{\Gamma}} \searrow$	*8,824*%	**16,67**%	0%	**14,58**%
Wine	$\gamma \nearrow$	*8,824*%	17,36%	2,941%	16,67%
Wine-2c	$\gamma \searrow$	*11,76*%	22,22%	2,941%	13,89%
Wine-2c	$\gamma_{\odot} \searrow$	*11,76*%	20,14%	0%	**12,5**%
Wine-2c	$\gamma_{\bar{\Gamma}} \searrow$	*8,824*%	**16,67**%	0%	**11,81**%
Wine-2c	$\gamma \nearrow$	*8,824*%	17,36%	0%	13,89%
Glass	$\gamma \searrow$	**17,95**%	**51,43**%	30,77%	50,29%
Glass	$\gamma_{\odot} \searrow$	*28,21*%	43,43%	25,64%	**42,86**%
Glass	$\gamma_{\bar{\Gamma}} \searrow$	20,51%	**49,71**%	28,21%	**48,57**%
Glass	$\gamma \nearrow$	*25,64*%	40,57%	35,9%	40,57%
Glass-2c	$\gamma \searrow$	**12,82**%	**38,29**%	**10,26**%	**30,29**%
Glass-2c	$\gamma_{\odot} \searrow$	15,38%	**40**%	**10,26**%	**29,71**%
Glass-2c	$\gamma_{\bar{\Gamma}} \searrow$	15,38%	**39,43**%	**10,26**%	**30,86**%
Glass-2c	$\gamma \nearrow$	*20,51*%	29,14%	17,95%	26,86%
Segment	$\gamma \searrow$	*8,571*%	30,29%	5,714%	29,71%
Segment	$\gamma_{\odot} \searrow$	*11,43*%	22,29%	11,43%	21,71%
Segment	$\gamma_{\bar{\Gamma}} \searrow$	*5,714*%	**29,14**%	11,43%	29,14%
Segment	$\gamma \nearrow$	*5,714*%	**26,29**%	**2,857**%	**25,14**%
Segment-2c	$\gamma \searrow$	*8,571*%	21,71%	8,571%	16,57%
Segment-2c	$\gamma_{\odot} \searrow$	*8,571*%	20,57%	11,43%	14,86%
Segment-2c	$\gamma_{\bar{\Gamma}} \searrow$	*8,571*%	21,71%	14,29%	17,71%
Segment-2c	$\gamma \nearrow$	*5,714*%	**17,71**%	**5,714**%	**15,43**%

TAB. 5.2 – *Taux d'erreur avec la base de validation (e_V) et pourcentage d'exemples utilisés par la fonction de décision d'un ppv (Z %) ou d'un SVM (VS %),* $1^{ère} partie$.

Pour les SVM l'apprentissage est réalisé sur la base simplifiée produite par notre méthode. Les hyper-paramètres des SVM sont les mêmes que ceux relatifs à la méthode directe. La réduction du nombre de vecteurs de support est moins importante que celle du nombre d'exemples avec la règle PPV, mais elle reste significative. Par exemple, pour les bases **Qijpeg2M** et **Segment** le nombre de vecteurs de support utilisés est presque divisé par 2. Les capacités en généralisation sont dans la majorité des cas identiques ou faiblement dégradées et pour certaines bases améliorées. C'est le cas pour les bases **Qijpeg2M** et **Segment**, même si ce n'est pas le même critère qui permet cette amélioration.

Si l'on compare les fonctions de décision des PPV et des SVM, on remarque que l'écart entre le nombre d'exemples utilisés par les deux méthodes est faible, même si les SVM en utilisent moins. Dans certains cas la fonction de décision PPV est meilleure que celle d'un SVM (**Glass** et **Pendigits-2c** par exemple). La procédure de sélection des hyper-paramètres des SVM étant simple, cela peut expliquer ces différences de performances.

		ppv		SVM	
Base	critère	e_V	Z %	e_V	VS %
Bupa	$\gamma \searrow$	*43,28*%	62,23%	**26,87**%	**50**%
Bupa	$\gamma_{\odot} \searrow$	38,81%	**57,19**%	**28,36**%	**44,24**%
Bupa	$\gamma_{\bar{\Gamma}} \searrow$	38,81%	**58,99**%	**31,34**%	**47,12**%
Bupa	$\gamma \nearrow$	**29,85**%	**54,68**%	**29,85**%	**46,04**%
Ionosphere	$\gamma \searrow$	24,64%	39,5%	**2,899**%	**24,2**%
Ionosphere	$\gamma_{\odot} \searrow$	**18,84**%	**35,23**%	**2,899**%	**20,64**%
Ionosphere	$\gamma_{\bar{\Gamma}} \searrow$	21,74%	**30,6**%	4,348%	**21,71**%
Ionosphere	$\gamma \nearrow$	26,09%	27,4%	4,348%	**17,79**%
Breast-cancer	$\gamma \searrow$	7,407%	13,87%	2,222%	8,394%
Breast-cancer	$\gamma_{\odot} \searrow$	8,148%	14,05%	**1,481**%	**8,212**%
Breast-cancer	$\gamma_{\bar{\Gamma}} \searrow$	8,148%	12,77%	2,222%	8,394%
Breast-cancer	$\gamma \nearrow$	5,926%	**10,22**%	2,222%	7,482%
OptDigits-2c	$\gamma \searrow$	0,6579%	1,794%	0%	1,794%
OptDigits-2c	$\gamma_{\odot} \searrow$	0%	**2,121**%	0%	1,958%
OptDigits-2c	$\gamma_{\bar{\Gamma}} \searrow$	0%	**2,121**%	0%	1,958%
OptDigits-2c	$\gamma \nearrow$	0%	**0,9788**%	0%	**0,9788**%
Pima	$\gamma \searrow$	**31,58**%	53,25%	23,03%	**42,37**%
Pima	$\gamma_{\odot} \searrow$	**30,92**%	53,41%	23,68%	**42,37**%
Pima	$\gamma_{\bar{\Gamma}} \searrow$	**32,24**%	51,14%	24,34%	41,07%
Pima	$\gamma \nearrow$	**28,29**%	44,16%	26,32%	38,15%
Vehicle	$\gamma \searrow$	36,53%	46,83%	17,37%	40,8%
Vehicle	$\gamma_{\odot} \searrow$	35,93%	50,81%	17,37%	43,45%
Vehicle	$\gamma_{\bar{\Gamma}} \searrow$	37,72%	47,13%	19,16%	40,35%
Vehicle	$\gamma \nearrow$	35,93%	41,09%	20,96%	37,11%
Vehicle-2c	$\gamma \searrow$	9,581%	21,06%	2,395%	12,96%
Vehicle-2c	$\gamma_{\odot} \searrow$	8,383%	16,94%	2,395%	11,05%
Vehicle-2c	$\gamma_{\bar{\Gamma}} \searrow$	8,383%	21,94%	2,395%	13,84%
Vehicle-2c	$\gamma \nearrow$	10,18%	16,05%	3,593%	11,78%
Letter-2c	$\gamma \searrow$	1,613%	2,811%	1,29%	1,847%
Letter-2c	$\gamma_{\odot} \searrow$	0,9677%	3,213%	0,6452%	2,169%
Letter-2c	$\gamma_{\bar{\Gamma}} \searrow$	1,29%	3,133%	0,9677%	2,088%
Letter-2c	$\gamma \nearrow$	1,613%	2,41%	0,6452%	1,847%
Pendigits-2c	$\gamma \searrow$	1,238%	0,7056%	2,751%	0,6414%
Pendigits-2c	$\gamma_{\odot} \searrow$	2,613%	**0,1283**%	2,613%	0,1283%
Pendigits-2c	$\gamma_{\bar{\Gamma}} \searrow$	**0,1376**%	**0,8339**%	0,8253%	0,7056%
Pendigits-2c	$\gamma \nearrow$	1,788%	0,5131%	5,502%	0,449%

TAB. 5.3 – *Taux d'erreur avec la base de validation (e_V) et pourcentage d'exemples utilisés par la fonction de décision d'un ppv (Z %) ou d'un SVM (VS %),* $2^{ème} partie$.

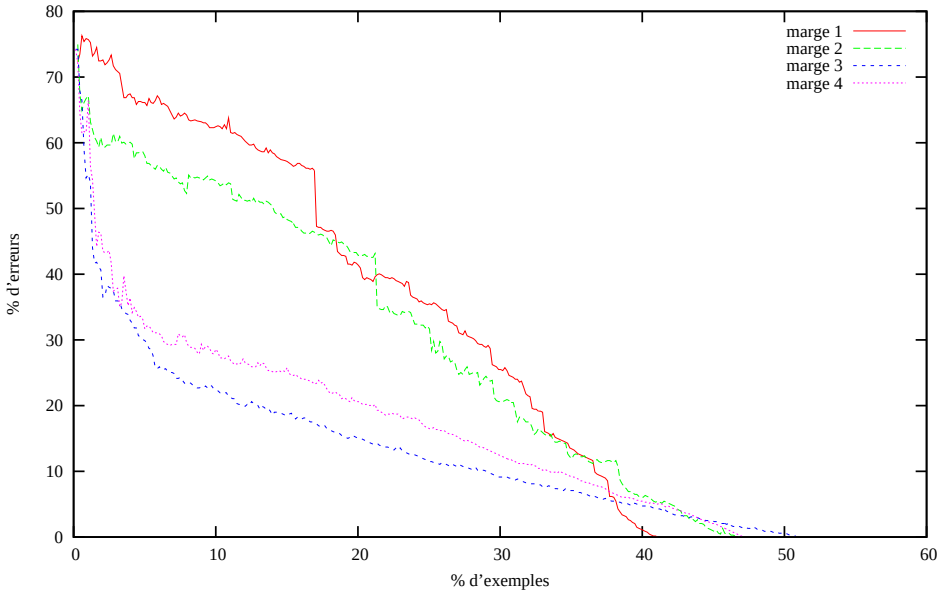
En ce qui concerne la réduction du nombre d'exemples utilisés, la méthode $[\gamma \nearrow]$ est globalement plus performante, même si cela ne se traduit pas forcément par une capacité de généralisation accrue. Le risque d'incorporer des exemples peu pertinents dans les premières itérations peut réduire les capacités en généralisation. La méthode $[\gamma \searrow]$ utilise systématiquent plus d'exemples que la méthode $[\gamma \nearrow]$, mais peut par contre produire des fonctions de décision qui ont de meilleures capacités de généralisation comme avec les bases **Glass** et **Glass-2c** pour les raisons précédemment évoquées.

Il est donc difficile de dire qu'un de ces 4 critères est globalement plus performant que les autres. Une des raisons est que pour définir une méthode de sélection qui s'appuie sur un principe SRM, nous avons choisi de produire des règles PPV qui soient systématiquement cohérentes avec la base d'apprentissage. Suivant ce principe, toute règle PPV qui utilise peu d'exemples pour être cohérente avec la base initiale a une borne supérieure sur son erreur en généralisation d'autant plus faible que le nombre d'exemples utilisés est réduit (*cf.* équation (2.51)). Ce résultat théorique correspond à une borne plus où moins pessimiste de l'erreur réelle et elle est définie uniquement pour un seuil de confiance δ donné. En pratique, même si l'utilisation d'un tel principe permet de contrôler les risques de sur-apprentissage (c-à-d réduire les effets de la variance), il ne permet pas de choisir suffisamment finement la meilleure solution. On peut vérifier dans les tableaux 5.2 et 5.3 que les solutions les plus performantes en généralisation ne sont pas systématiquement celles qui utilisent le moins d'exemples parmi toutes celles qui produisent une règle PPV cohérente avec la base d'apprentissage. Un autre effet à prendre en compte est que pour pouvoir obtenir un taux d'erreur nul sur la base d'apprentissage, l'algorithme ajoute les derniers exemples difficiles à classer correctement, même si les marges associées sont faibles [13]. Dans la plupart des cas cela correspond à ajouter des exemples non représentatifs du problème d'apprentissage. Il paraît alors plus logique de s'arrêter avant d'atteindre la cohérence avec la base d'apprentissage. Le problème est alors de choisir un critère d'arrêt adapté pour chacune des 4 approches proposées.

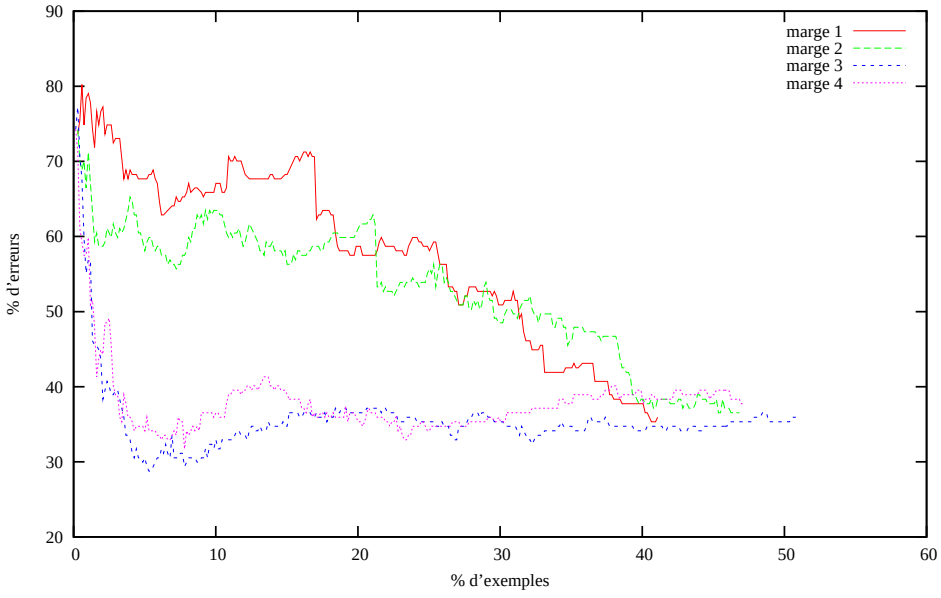
Effet du nombre d'itérations de la méthode

Avant de réaliser la définition d'un tel critère, il est important d'observer le comportement de notre méthode au cours de ces itérations avec les 4 critères proposés. Nous allons pour cela mesurer le taux d'erreur en fonction du pourcentage d'exemples sélectionnés par notre méthode pour chacun des quatre critères de marge. Cette mesure est réalisée à la fois sur la base d'apprentissage et sur la base de validation. La base de validation est dans ce cas particulièrement utile à l'observation du pouvoir de généralisation avec peu d'exemples, ce qui permet d'illustrer les capacités de ces critères à guider notre méthode dans la sélection des exemples les plus pertinents. Plus cette sélection sera efficace et plus l'utilisation d'un critère d'arrêt avant la cohérence permettra d'avoir un compromis efficace entre réduction du nombre d'exemples et capacité de généralisation. Les figures 5.2(a) et 5.2(b) sont une illustration de la variation du taux d'erreur repectivement avec la base d'apprentissage et de validation pour une des bases de données. Des résultats suplémentaires sont donnés dans l'annexe C.

13. Sauf pour le premier critère qui est plus susceptible de réaliser l'ajout d'exemples aberrants dès ces premières itérations.



(a) apprentissage



(b) validation

FIG. 5.2 – *Taux d'erreur des 4 critères de marge (1 :* $[\gamma \nearrow]$*, 2 :* $[\gamma \searrow]$*, 3 :* $[\gamma_{\odot} \searrow]$ *et 4 :*$[\gamma_{\bar{\Gamma}} \searrow]$*) suivant le pourcentage d'exemples utilisé pour la base* **vehicle**.

Globalement le fait qu'un exemple soit plus proche de ceux de sa classe que de ceux des autres classes permet de produire une mesure de confiance plus efficace pour la sélection d'exemples pertinents comparativement à l'utilisation de la proximité d'un couple d'exemples proche de la frontière de décision. Ceci est d'autant plus vrai pour les bases contenant des données qui sont bruitées. A partir de ces différentes observations, il semble important de modifier le critère d'arrêt de notre méthode lorsque la marge de confiance utilisée est $[\gamma_{\odot} \searrow]$ ou $[\gamma_{\overline{\Gamma}} \searrow]$ afin de réduire l'effet indésirable précédemment évoqué et d'améliorer la qualité du compromis entre la réduction de la complexité et l'augmentation des capacités de généralisation.

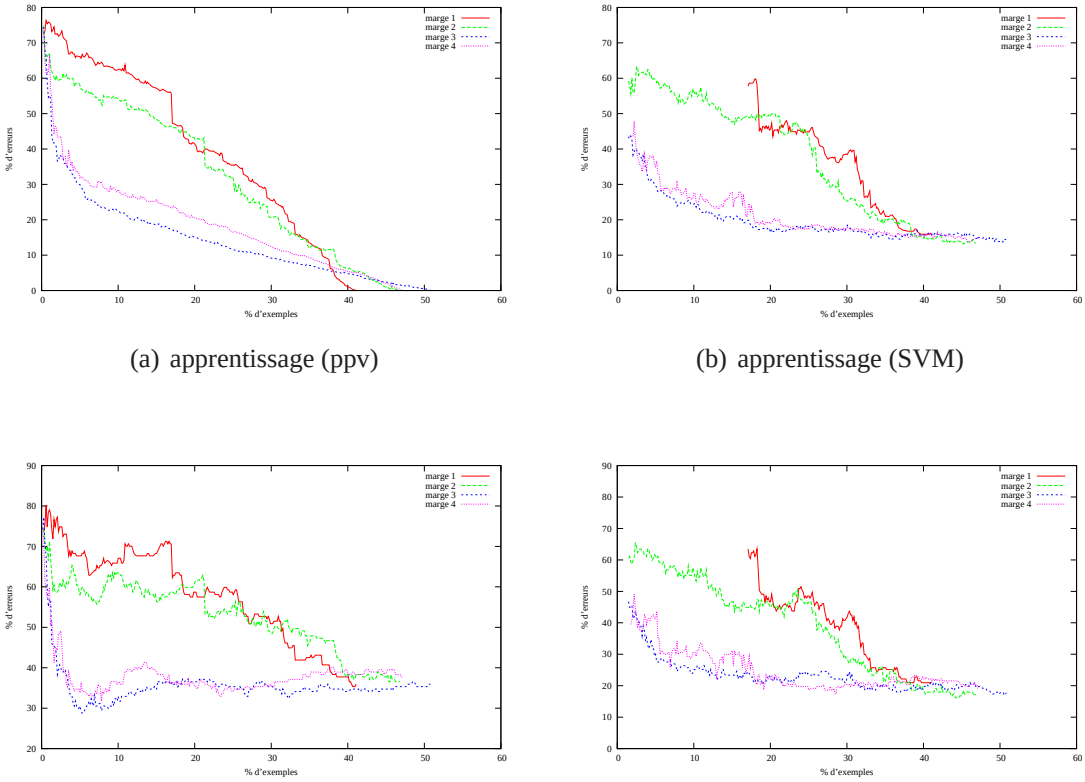
Lien entre PPV et SVM

Pour illustrer le lien entre la performance en généralisation de la règle PPV avec un sous-ensemble d'exemples Z_i^p et celle avec un SVM entraîné à partir du même sous-ensemble d'exemples Z_i^p, nous avons systématiquement entraîné un SVM sur chaque ensemble d'exemples Z_i^p produit par notre méthode. Les valeurs des hyper-paramètres d'un SVM sont constantes pour une base de données fixée et elles correspondent aux valeurs dans le tableau A.5 de la procédure de sélection *directe*. Les figures 5.3 et 5.4 illustrent la corrélation importante qui existe entre les taux d'erreur de ces deux types de fonctions de décision avec la base *Vehicle*. Les mêmes corrélations ont été observées avec les autres bases de données utilisées lors de ces expérimentations. Ces résultats montrent que la règle PPV peut être utilisée pour prévoir avec une précision suffisamment grande les capacités de généralisation d'un SVM avec un sous-ensemble d'exemples. Ces résultats montrent également la supériorité des deux critères d'hypothèses $[\gamma_{\odot} \searrow]$ et $[\gamma_{\overline{\Gamma}} \searrow]$ par rapport aux deux autres ($[\gamma \nearrow]$ et $[\gamma \searrow]$) pour sélectionner un sous-ensemble réduit d'exemples pertinents qui permet de produire des fonctions de décision efficaces avec des SVM.

5.3.1.5 Critères d'arrêt avant cohérente

L'objectif est maintenant de définir un critère d'arrêt qui permette à notre méthode de sélectionner suffisamment d'exemples pertinents pour produire une fonction de décision performante en généralisation, mais qui s'arrête suffisamment tôt pour ne pas ajouter d'exemples inutiles à l'amélioration des capacités de généralisation. L'arrêt prématuré est encore plus important si l'ajout d'exemples avec l'un des deux critères de marge d'hypothèses peut produire des dégradations des performances en généralisation. Globalement nous avons choisi que l'ajout d'un exemple n'est pas réalisé si la valeur de la marge d'hypothèses devient trop faible, même si cela conduit à produire une règle PPV qui ne soit pas cohérente avec la base d'apprentissage. Notre méthode étant basée sur l'exploitation de l'ensemble candidat Z^c, lorsque cet ensemble est vide l'arrêt de celle-ci est obligatoire. Nous conservons donc comme critère d'arrêt, la cohérence de la règle PPV avec la base d'apprentissage, même si les valeurs des marges d'hypothèses des exemples restants sont encore importantes.

Pour le critère $[\gamma_{\odot} \searrow]$, nous avons considéré qu'une valeur trop faible de ce critère caractérise le fait que l'exemple correspondant a une probabilité importante d'être un mauvais représentant du problème d'apprentissage. Nous avons alors choisi le fait qu'un exemple doit avoir au minimum k exemples de sa classe comme plus proches voisins, la



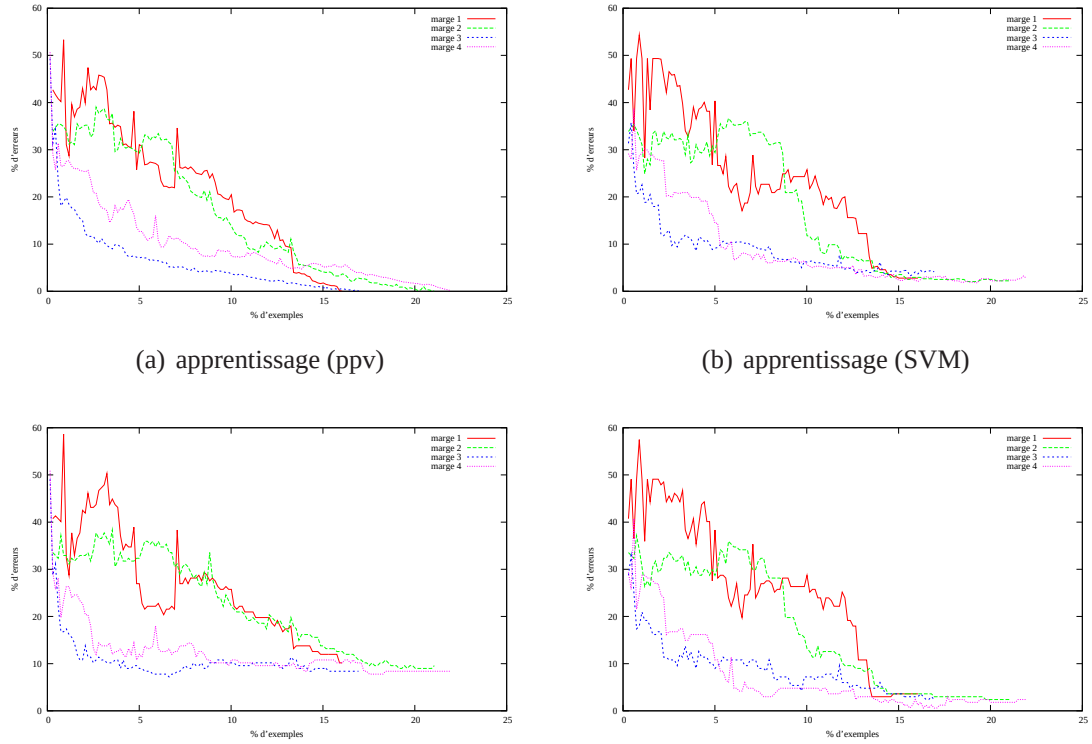
(a) apprentissage (ppv)

(b) apprentissage (SVM)

(c) validation (ppv)

(d) validation (SVM)

FIG. 5.3 – *Comparaison entre les performances d'un ppv et d'un SVM à partir d'un même sous-ensemble d'exemples pertinents extrait de la base* **Vehicle**. *Pour les SVM les courbes (b et d) ne commencent que lorsque l'ensemble* Z^p *a au moins un exemple de chaque classe.*



(a) apprentissage (ppv) (b) apprentissage (SVM)

(c) validation (ppv) (d) validation (SVM)

FIG. 5.4 – *Comparaison entre les performances d'un ppv et d'un SVM à partir d'un même sous ensemble d'exemples pertinents extrait de la base* **Vehicle-2c**.

valeur de k devant être fixée au préalable. Le critère d'arrêt lié à $[\gamma_{\odot} \searrow]$ peut être formulé de la façon suivante:

$$\left(\forall z \in Z_i^c : \text{ nombre-ppv}_{\text{mc}}(z, Z_m^A) < k\right) \rightarrow \text{Arrêt} \tag{5.16}$$

Pour le critère $[\gamma_{\bar{\Gamma}} \searrow]$, nous avons choisi que l'arrêt prématuré soit réalisé si la sélection d'un exemple de Z^c conduit forcément à ajouter un exemple de marge d'hypothèse $\bar{\Gamma}$ inférieure ou égale à zéro. L'utilisation de ce critère d'arrêt correspond à accepter seulement les exemples qui ont un effet positif (en moyenne) avec la règle PPV, effet positif mesuré sur la base initiale. Il existe également dans une base de données, des exemples qui ne sont l'exemple le plus proche d'aucun autre exemple (mis à part eux-mêmes). Ces exemples ont une marge d'hypothèse $\bar{\Gamma}_h$ nulle. Ils ne seront pas pris en compte par notre critère d'arrêt prématuré, car ils sont considérés comme non productifs pour la règle PPV (relativement à la base d'apprentissage). Le critère d'arrêt lié à $[\gamma_{\bar{\Gamma}} \searrow]$ peut être formulé de la façon suivante:

$$\left(\forall z \in Z_i^c : \bar{\Gamma}_h(z) \leq 0\right) \rightarrow \text{Arrêt} \tag{5.17}$$

5.3.1.6 Expérimentations des deux critères d'arrêts prématurés

Nous avons réalisé un ensemble d'expérimentations avec les mêmes bases de données que celles utilisées dans la section 5.3.1.4. Nous avons utilisé les critères d'arrêts prématurés (CAP) précédemment évoqués et avons choisi une valeur de $k = 2$ pour le CAP (5.16). L'utilisation conjointe du critère de pertinence $[\gamma_{\odot} \searrow]$ et du CAP (5.16) est notée $[\gamma_{\odot} \searrow \ominus]$ avec $\ominus$ représentant l'utilisation d'un CAP. De la même façon l'utilisation conjointe du critère de pertinence $[\gamma_{\bar{\Gamma}} \searrow]$ et du CAP (5.17) est notée $[\gamma_{\bar{\Gamma}} \searrow \ominus]$.

Les valeurs dans le tableau 5.4 lorsqu'elles sont comparées à celles des tableaux 5.2 et 5.3 montrent une baisse significative du nombre d'exemples utilisés par chaque fonction de décision produite. Ceci est particulièrement vrai avec le critère (5.16). La diminution du nombre d'exemples s'accompagne pour la plupart des bases d'une diminution ou du maintien du nombre d'erreurs réalisées avec la base de validation. L'exemple de la base *QIjpeg2M* où l'on passe d'un taux d'erreur de 39,8% avec 34,5% d'exemples utilisés à 22,5% d'erreurs avec seulement 8,6% d'exemples utilisés pour la règle PPV est particulièrement évocateur. Il en va de même, pour cette base, avec la fonction de décision produite avec des SVM où l'on passe de 21,6% d'erreurs avec 31% d'exemples utilisés à 18,9% d'erreurs avec seulement 8,6% d'exemples.

Globalement les SVM sont plus performants que la règle PPV à la fois en nombre d'exemples utilisés et en nombre d'erreurs réalisées sur la base de validation, mais cela n'est pas systématique et peut dépendre de la bonne adéquation des hyper-paramètres choisis pour un SVM en fonction des changements de la nature de la base d'apprentissage par sa simplification. Nous reviendrons plus tard sur cette possibilité. Le fait que les fonctions de décision produites avec la règle PPV soient globalement moins performantes que les SVM, alors que ce sont elles qui ont servi au choix des exemples pertinents, est essentiellement dû à la nature discontinue de ces fonctions (voir [VINCEN01] pour plus de détails sur la nature de ce problème). Cet effet est accentué par la réduction du nombre d'exemples utilisés par la règle PPV. Le cas de la base *Bupa* dans le tableau 5.4 avec $[\gamma_{\odot} \searrow \ominus]$ est caractéristique de ce phénomène, car on passe d'un taux d'erreur de 47,8% avec 29,1%

		1-ppv		SVM	
Base	critère	e_V	Z %	e_V	VS %
QIjpeg2M	$\gamma_{\odot} \searrow \ominus$	22,52%	8,621%	18,92%	8,621%
QIjpeg2M	$\gamma_{\bar{\Gamma}} \searrow \ominus$	27,93%	25%	24,32%	25%
QIjpeg2M-2c	$\gamma_{\odot} \searrow \ominus$	0%	3,448%	0%	2,586%
QIjpeg2M-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	2,703%	9,483%	2,703%	6,897%
Iris	$\gamma_{\odot} \searrow \ominus$	0%	5%	0%	5%
Iris	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	14,17%	0%	12,5%
Iris-2c	$\gamma_{\odot} \searrow \ominus$	0%	2,5%	0%	2,5%
Iris-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	2,5%	0%	2,5%
Wine	$\gamma_{\odot} \searrow \ominus$	11,76%	10,42%	2,941%	9,028%
Wine	$\gamma_{\bar{\Gamma}} \searrow \ominus$	8,824%	15,28%	0%	12,5%
Wine-2c	$\gamma_{\odot} \searrow \ominus$	11,76%	10,42%	5,882%	8,333%
Wine-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	8,824%	15,28%	0%	10,42%
Glass	$\gamma_{\odot} \searrow \ominus$	23,08%	8,571%	26,38%	8,571%
Glass	$\gamma_{\bar{\Gamma}} \searrow \ominus$	28,21%	31,43%	30,77%	30,86%
Glass-2c	$\gamma_{\odot} \searrow \ominus$	12,82%	5,714%	10,26%	5,143%
Glass-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	17,95%	25,14%	17,95%	20%
Segment	$\gamma_{\odot} \searrow \ominus$	8,571%	8%	11,43%	8%
Segment	$\gamma_{\bar{\Gamma}} \searrow \ominus$	5,714%	24%	11,43%	23,43%
Segment-2c	$\gamma_{\odot} \searrow \ominus$	14,29%	8,571%	11,43%	7,429%
Segment-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	5,714%	20%	14,29%	16%
bupa	$\gamma_{\odot} \searrow \ominus$	38,81%	9,353%	34,33%	6,475%
bupa	$\gamma_{\bar{\Gamma}} \searrow \ominus$	47,76%	29,14%	26,87%	25,18%
ionosphere	$\gamma_{\odot} \searrow \ominus$	26,09%	6,05%	15,94%	4,982%
ionosphere	$\gamma_{\bar{\Gamma}} \searrow \ominus$	20,29%	17,08%	7,246%	14,59%
breast-cancer	$\gamma_{\odot} \searrow \ominus$	2,222%	2,372%	1,481%	1,46%
breast-cancer	$\gamma_{\bar{\Gamma}} \searrow \ominus$	9,63%	10,58%	1,481%	6,934%
OptDigits-2c	$\gamma_{\odot} \searrow \ominus$	0%	2,121%	0%	1,958%
OptDigits-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	2,121%	0%	1,958%
Pima	$\gamma_{\odot} \searrow \ominus$	25%	6,169%	23,68%	4,058%
Pima	$\gamma_{\bar{\Gamma}} \searrow \ominus$	35,53%	26,46%	23,03%	23,21%
vehicle	$\gamma_{\odot} \searrow \ominus$	29,94%	7,953%	25,15%	6,922%
vehicle	$\gamma_{\bar{\Gamma}} \searrow \ominus$	34,73%	25,63%	19,16%	23,12%
vehicle-2c	$\gamma_{\odot} \searrow \ominus$	8,982%	7,658%	8,982%	5,007%
vehicle-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	7,784%	17,82%	1,796%	12,22%
letter-2c	$\gamma_{\odot} \searrow \ominus$	0,6452%	2,811%	0,9677%	1,847%
letter-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	1,613%	2,972%	0,9677%	2,008%
pendigits-2c	$\gamma_{\odot} \searrow \ominus$	2,613%	0,1283%	2,613%	0,1283%
pendigits-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0,1376%	0,8339%	0,8253%	0,7056%

TAB. 5.4 – *Taux d'erreur et pourcentage d'exemples utilisés avec la fonction de décision d'un ppv ou d'un SVM lorsqu'un critère d'arrêt prématuré est utilisé.*

des exemples pour la règle PPV à un taux d'erreur de 26,9% avec 25,2% d'exemples utilisés avec les SVM. Le nombre d'exemples utilisés est très proche dans les deux cas, mais le taux d'erreur est presque divisé par deux avec l'utilisation d'un SVM, ce qui illustre leur pouvoir de généralisation important lorsque les hyper-paramètres sont correctement choisis. La lecture des valeurs du tableau 5.4 montre que ce n'est pas systématiquement le même critère qui conduit à produire à la fois une réduction du nombre d'erreurs et une réduction du nombre d'exemples, lorsque la base d'apprentissage change.

Une autre remarque concerne le critère d'arrêt (5.17) qui semble agir trop tardivement au regard des courbes précédentes (*cf.* les figures C.1 à C.10 en prenant les taux d'exemples utilisés dans le tableau 5.4). L'utilisation de la valeur moyenne de Γ semble ne pas être assez fine et cette mesure devra être modifiée par la suite pour améliorer l'efficacité d'un critère d'arrêt prématuré à partir de sa valeur.

Influence des hyper-paramètres

Nous avons dit précédemment qu'il ne faut pas négliger l'importance du choix des hyper-paramètres des SVM. En effet, le choix initial des hyper-paramètres réalisé par la méthode directe (*cf.* section A.2) avec une base de données complète peut être moins approprié avec la version réduite à cause de la modification de sa constitution (voir les observations réalisées avec la seconde approche en section 5.3.2.1 et 5.3.2.2 qui corroborent ce phénomène, ainsi que celles en section 4.2.4.2). Nous avons alors réalisé deux procédures de sélection des hyper-paramètres des SVM à partir de la base d'apprentissage simplifiée pour vérifier l'influence de ces paramètres sur la capacité de généralisation des SVM. La première est identique à celle de la section A.2 qui consiste à rechercher la solution qui comporte le moins de vecteurs de support. L'idée est toujours de produire la fonction de décision la plus simple possible en minimisation de la borne supérieure (2.52) afin d'amplifier l'effet de compression. Les résultats du tableau 5.5 correspondent à la sélection des hyper-paramètres avec cette procédure. La deuxième est une procédure classique de validation croisée en 10 parties à partir des bases d'apprentissage simplifiées produites par les deux méthodes $[\gamma_{\odot} \searrow \ominus]$ et $[\gamma_{\bar{\Gamma}} \searrow \ominus]$. Les résultats du tableau 5.6 correspondent à la sélection des hyper-paramètres avec cette procédure. Pour ces deux procédures l'ensemble des couples (C,σ) testés sont les mêmes que ceux de la section A.2 avec la technique de *grid search*.

Les résultats du tableau 5.5 illustrent que la sélection par minimisation du nombre de vecteurs de support peut produire la sélection d'un modèle (C,σ) qui réduit à la fois le taux d'erreur et le nombre de vecteurs de support utilisés (base *Segment* par exemple), mais ce principe peut également produire une augmentation significative du taux d'erreur (base **Glass** par exemple). Ces résultats illustrent également que le gain sur la réduction du nombre de vecteurs de support est globalement assez faible car la majorité des exemples pertinents sélectionnés sont des vecteurs de support. Les résultats du tableau 5.6 montrent que la sélection par validation croisée réduit dans la majorité des cas le taux d'erreur et ne l'augmente jamais. Ces résultats illustrent également que si cette méthode de sélection peut augmenter le nombre de vecteurs de support, cette augmentation n'est jamais très importante.

A partir de ces résultats, il semble évident qu'il est préférable d'utiliser la sélection d'un modèle efficace après simplification de la base d'apprentissage par la méthode de validation croisée quel que soit le critère de simplification utilisé.

5.3.1.7 Pré-filtrage par SVM

Pour pallier les difficultés de notre méthode à réaliser une sélection des exemples les plus pertinents sur des bases d'apprentissage de taille plus conséquente, il est possible d'utiliser préalablement une technique de simplification qui soit plus rapide, même si elle n'est pas aussi performante dans sa sélection d'exemples pertinents, cela permet de supprimer un maximum d'exemples parmi les moins utiles à notre méthode et par la suite d'appliquer notre méthode à cette base pré-simplifiée, si la réduction est suffisante. Nous nommons cette technique *pré-filtrage*. Au regard du pourcentage moyen d'exemples utilisés par les SVM (*cf.* tableau A.5) et en s'inspirant de la méthode de Dong et al [DONG05], nous avons choisi d'utiliser directement la fonction de décision produite par un SVM pour

		Avant		Après sélection avec minSV			
Base	critère	e_V	VS %	e_V	VS %	C	σ
QIjpeg2M	$\gamma_{\odot} \searrow \ominus$	18,92%	8,621%	23,96%	7,759%	0,01	71
QIjpeg2M	$\gamma_{\bar{\Gamma}} \searrow \ominus$	24,32%	25%	26,13%	24,14%	100	2,2
QIjpeg2M-2c	$\gamma_{\odot} \searrow \ominus$	0%	2,586%	0%	**1,724**%	10	2,2
QIjpeg2M-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	2,703%	6,897%	**1,802**%	**6,034**%	100	2,2
Iris	$\gamma_{\odot} \searrow \ominus$	0%	5%	0%	**4,167**%	100	7,1
Iris	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	12,5%	0%	**9,167**%	1000	7,1
Iris-2c	$\gamma_{\odot} \searrow \ominus$	0%	2,5%	0%	**1,667**%	10	7,1
Iris-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	2,5%	0%	**1,667**%	10	7,1
Wine	$\gamma_{\odot} \searrow \ominus$	2,941%	9,028%	**0**%	**7,639**%	1000	7,1
Wine	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	12,5%	2,941%	10,42%	100	2,2
Wine-2c	$\gamma_{\odot} \searrow \ominus$	5,882%	8,333%	5,882%	**6,944**%	100	2,2
Wine-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	10,42%	0%	**7,639**%	1000	7,1
Glass	$\gamma_{\odot} \searrow \ominus$	26,38%	8,571%	*61,54*%	7,429%	0,01	71
Glass	$\gamma_{\bar{\Gamma}} \searrow \ominus$	30,77%	30,86%	*46,15*%	28,57%	1000	2,2
Glass-2c	$\gamma_{\odot} \searrow \ominus$	10,26%	5,143%	*17,95*%	4%	1000	7,1
Glass-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	17,95%	20%	*23,08*%	16%	1000	2,2
Segment	$\gamma_{\odot} \searrow \ominus$	11,43%	8%	**2,857**%	**7,429**%	1000	7,1
Segment	$\gamma_{\bar{\Gamma}} \searrow \ominus$	11,43%	23,43%	**8,571**%	**22,29**%	1000	7,1
Segment-2c	$\gamma_{\odot} \searrow \ominus$	11,43%	7,429%	20%	6,286%	1000	7,1
Segment-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	14,29%	16%	14,29%	**12**%	1000	7,1
bupa	$\gamma_{\odot} \searrow \ominus$	34,33%	6,475%	34,33%	**3,597**%	1000	2,2
bupa	$\gamma_{\bar{\Gamma}} \searrow \ominus$	26,87%	25,18%	41,79%	23,74%	1000	0,71
ionosphere	$\gamma_{\odot} \searrow \ominus$	15,94%	4,982%	20,29%	4,27%	1000	7,1
ionosphere	$\gamma_{\bar{\Gamma}} \searrow \ominus$	7,246%	14,59%	10,14%	9,964%	1000	7,1
breast-cancer	$\gamma_{\odot} \searrow \ominus$	1,481%	1,46%	1,481%	**1,277**%	100	2,2
breast-cancer	$\gamma_{\bar{\Gamma}} \searrow \ominus$	1,481%	6,934%	1,481%	**6,204**%	1000	7,1
OptDigits-2c	$\gamma_{\odot} \searrow \ominus$	0%	1,958%	0%	**1,468**%	100	7,1
OptDigits-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	1,958%	0%	**1,305**%	100	71
Pima	$\gamma_{\odot} \searrow \ominus$	23,68%	4,058%	25,66%	2,76%	1000	7,1
Pima	$\gamma_{\bar{\Gamma}} \searrow \ominus$	23,03%	23,21%	33,55%	20,45%	1000	0,71
vehicle	$\gamma_{\odot} \searrow \ominus$	25,15%	6,922%	**24,55**%	**5,155**%	1000	7,1
vehicle	$\gamma_{\bar{\Gamma}} \searrow \ominus$	19,16%	23,12%	23,35%	18,7%	1000	7,1
vehicle-2c	$\gamma_{\odot} \searrow \ominus$	8,982%	5,007%	9,581%	3,093%	1000	7,1
vehicle-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	1,796%	12,22%	1,796%	**5,891**%	1000	2,2
letter-2c	$\gamma_{\odot} \searrow \ominus$	0,9677%	1,847%	1,29%	1,205%	1000	7,1
letter-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0,9677%	2,008%	2,581%	1,285%	1000	2,2
pendigits-2c	$\gamma_{\odot} \searrow \ominus$	2,613%	0,1283%	2,613%	0,1283%	0,01	71
pendigits-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0,8253%	0,7056%	0,9313%	0,7697%	0,01	71

TAB. 5.5 – *Sélection des hyperparamètres des SVM avec la technique de minimisation du nombre de vecteurs de support pour les bases d'apprentissage simplifiées par notre méthode.*

		Avant		Après sélection par CV10			
Base	critère	e_V	VS %	e_V	VS %	C	σ
QIjpeg2M	$\gamma_{\odot} \searrow \ominus$	18,92%	8,621%	**17,12**%	8,621%	10	2,2
QIjpeg2M	$\gamma_{\bar{\Gamma}} \searrow \ominus$	24,32%	25%	24,32%	25%	10	0,71
QIjpeg2M-2c	$\gamma_{\odot} \searrow \ominus$	0%	2,586%	0%	3,448%	10	0,71
QIjpeg2M-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	2,703%	6,897%	**1,802**%	9,483%	10	0,71
Iris	$\gamma_{\odot} \searrow \ominus$	0%	5%	0%	4,167%	10	7,1
Iris	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	12,5%	0%	13,33%	10	2,2
Iris-2c	$\gamma_{\odot} \searrow \ominus$	0%	2,5%	0%	**1,667**%	10	7,1
Iris-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	2,5%	0%	**1,667**%	10	7,1
Wine	$\gamma_{\odot} \searrow \ominus$	2,941%	9,028%	**0**%	**8,333**%	100	2,2
Wine	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	12,5%	0%	13,19%	10	2,2
Wine-2c	$\gamma_{\odot} \searrow \ominus$	5,882%	8,333%	**0**%	**7,639**%	10	2,2
Wine-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	10,42%	0%	11,81%	10	2,2
Glass	$\gamma_{\odot} \searrow \ominus$	26,38%	8,571%	**23,08**%	8,571%	10	0,71
Glass	$\gamma_{\bar{\Gamma}} \searrow \ominus$	30,77%	30,86%	30,77%	31,43%	1	0,71
Glass-2c	$\gamma_{\odot} \searrow \ominus$	10,26%	5,143%	10,26%	5,714%	0,01	0,71
Glass-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	17,95%	20%	**12,82**%	**19,43**%	100	2,2
Segment	$\gamma_{\odot} \searrow \ominus$	11,43%	8%	**2,857**%	**7,429**%	1000	7,1
Segment	$\gamma_{\bar{\Gamma}} \searrow \ominus$	11,43%	23,43%	**8,571**%	**22,29**%	1000	7,1
Segment-2c	$\gamma_{\odot} \searrow \ominus$	11,43%	7,429%	**8,571**%	8%	10	0,71
Segment-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	14,29%	16%	**5,714**%	19,43%	1	0,71
bupa	$\gamma_{\odot} \searrow \ominus$	34,33%	6,475%	**31,34**%	**5,396**%	100	0,71
bupa	$\gamma_{\bar{\Gamma}} \searrow \ominus$	26,87%	25,18%	**23,88**%	**24,46**%	1000	2,2
ionosphere	$\gamma_{\odot} \searrow \ominus$	15,94%	4,982%	8,696%	**6,05**%	10	0,71
ionosphere	$\gamma_{\bar{\Gamma}} \searrow \ominus$	7,246%	14,59%	7,246%	17,08%	10	0,71
breast-cancer	$\gamma_{\odot} \searrow \ominus$	1,481%	1,46%	1,481%	2,007%	10	2,2
breast-cancer	$\gamma_{\bar{\Gamma}} \searrow \ominus$	1,481%	6,934%	1,481%	7,299%	1	0,71
OptDigits-2c	$\gamma_{\odot} \searrow \ominus$	0%	1,958%	0%	**1,794**%	10	7,1
OptDigits-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0%	1,958%	0%	**1,794**%	10	2,2
Pima	$\gamma_{\odot} \searrow \ominus$	23,68%	4,058%	**21,71**%	**3,084**%	100	2,2
Pima	$\gamma_{\bar{\Gamma}} \searrow \ominus$	23,03%	23,21%	**19,74**%	**22,73**%	100	7,1
vehicle	$\gamma_{\odot} \searrow \ominus$	25,15%	6,922%	**22,75**%	**5,891**%	1000	2,2
vehicle	$\gamma_{\bar{\Gamma}} \searrow \ominus$	19,16%	23,12%	**16,77**%	**20,62**%	100	2,2
vehicle-2c	$\gamma_{\odot} \searrow \ominus$	8,982%	5,007%	**6,587**%	6,627%	10	0,71
vehicle-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	1,796%	12,22%	**0,5988**%	**8,837**%	100	2,2
letter-2c	$\gamma_{\odot} \searrow \ominus$	0,9677%	1,847%	**0,6452**%	2,41%	10	0,71
letter-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0,9677%	2,008%	**0,3226**%	2,651%	10	0,71
pendigits-2c	$\gamma_{\odot} \searrow \ominus$	2,613%	0,1283%	2,613%	0,1283%	0,01	71
pendigits-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	0,8253%	0,7056%	**0,2751**%	0,8339%	1	71

TAB. 5.6 – *Sélection des hyperparamètres des SVM avec une procédure de validation croisée en 10 parties (CV10) sur les bases d'apprentissage simplifiées par notre méthode.*

réaliser cette pré-sélection. Nous considérons que tout point trop éloigné de la frontière de décision n'est pas utile à la décision et peut être ignoré, ce qui correspond à les considérer comme des exemples non pertinents de type 2.

Définissons cette technique de pré-filtrage à partir de la fonction de décision produite par un SVM ou d'un ensemble de fonctions de décision lorsqu'un schéma multi-classe est utilisé. Soit $f_{i,j}$ la fonction de décision produite par un SVM pour discriminer la classe ω_i de la classe ω_j. Nous définissons l'ensemble des exemples bien classés qui sont suffisamment éloignés de la marge géométrique (ou de l'ensemble des marges géométriques) comme :

$$Z_+(\beta) = \left\{(\mathbf{x},y) \,|\, (\mathbf{x},y) \in Z_m^A, \forall i,j \in \mathcal{Y} : y \cdot f_{i,j}(\mathbf{x}) > 1+\beta\right\} \tag{5.18}$$

avec β un coefficient qui permet de fixer quelle doit être l'importance de l'éloignement de la marge géométrique d'un exemple pour pouvoir le considérer comme suffisamment éloigné de la frontière de décision. Nous définissons de la même façon l'ensemble des exemples mal classés et éloignés de la marge géométrique (ou de l'ensemble des marges géométriques) comme:

$$Z_-(\beta) = \left\{(\mathbf{x},y) \,|\, (\mathbf{x},y) \in Z_m^A, \forall i,j \in \mathcal{Y} : y \cdot f_{i,j}(\mathbf{x}) < -1-\beta\right\} \tag{5.19}$$

L'ensemble Z_- représente l'ensemble des points aberrants de Z_m^A qui n'auraient pas été retenus par notre méthode comme des exemples pertinents avec une forte probabilité. A partir de (5.20) et (5.20), nous définissons l'ensemble Z_{f}^A (f pour filtrage) qui correspond aux exemples gardés par notre technique de pré-filtrage comme:

$$Z_{\mathrm{f}}^A(\beta) = Z_m^A \setminus (Z_+(\beta) \cup Z_-(\beta)) \tag{5.20}$$

Le choix de la valeur de β correspond à un compromis entre importance de la simplification et risque de supprimer des exemples significatifs.

5.3.1.8 Expérimentations avec pré-filtrage

Nous avons appliqué notre technique de pré-filtrage à l'ensemble des bases de données présentes dans le tableau A.5. Les valeurs des hyper-paramètres pour l'entraînement des SVM sont celles qui sont répertoriées dans le tableau A.5 pour la méthode directe. La valeur choisie pour β est $0{,}2$. Elle correspond au choix d'avoir une pré-sélection qui conserve l'ensemble des vecteurs de support, plus un nombre réduit d'exemples parmi les plus proches des frontières de décision, ceci afin d'obtenir une réduction significative du nombre d'exemples Z_{f}^A comparativement à ceux de la base initiale d'apprentissage. Le tableau 5.7 montre que la réduction du nombre d'exemples par la technique de pré-filtrage peut être significative, en particulier pour un grand nombre de bases de tailles importantes. Cette réduction est plus facile lorsque le problème ne comporte que deux classes, car la détermination du rejet d'un exemple n'est réalisée qu'à partir d'une seule frontière de décision. La colonne *incohérent* illustre que la suppression d'exemples non significatifs de type 3 est peu fréquente. La raison principale est que les SVM produisent peu d'exemples de ce type (*i.e.* appartenant à Z_-) car le principe d'optimisation avec marge molle a pour objectif de réduire au maximum le nombre d'exemples de ce type et cela même si l'exemple est par nature aberrant. Notre méthode avec CAP est donc plus à même de rejeter les exemples de cette catégorie. Le tableau 5.7 permet la comparaison

Base	sélection	incohérent	e_V	e_{V_f}	n_{SV}	n_{SV_f}
QIjpeg2M	96,6 %	0	21,6 %	20,7 %	70	71
QIjpeg2M-2c	16,4 %	0	0,901 %	1,8 %	15	7
Iris	86,7 %	0	0 %	0 %	26	26
Iris-2c	23,3 %	0	0 %	0 %	5	5
Wine	43,1 %	0	0 %	0 %	40	40
Wine-2c	28,5 %	0	0%	2,94 %	27	22
Glass	87,4%	0	20,5 %	20,5 %	127	127
Glass-2c	48,6 %	0	12,8 %	12,8 %	70	69
Segment	96 %	0	2,86 %	2,86 %	101	101
Segment-2c	39,4 %	0	8,57 %	2,86 %	42	34
Bupa	69,4 %	0	31,3 %	31,3 %	171	170
Ionosphere	42,3 %	0	4,35 %	4,35 %	94	94
Breast-cancer	17,9 %	0	2,22 %	2,22 %	61	62
OptDigits-2c	17,1 %	0	0 %	0 %	37	37
Pima	64,3 %	0	23 %	23 %	330	330
Vehicle	68 %	0	17,4 %	17,4 %	400	400
Vehicle-2c	25,2 %	0	1,8 %	1,8 %	135	135
Letter-2c	9,96 %	0	0,968 %	0,968 %	52	52
Pendigits-2c	8,34 %	0	1,24 %	1,24 %	24	19
OptDigits	50,1 %	0	1,58 %	1,58 %	917	916
Serous	74 %	0	34,7 %	34,7 %	1972	1972
Serous-2c	7,06 %	2	2,17 %	2,3 %	149	112
PageBlocks	17,1 %	0	5,77 %	5,87 %	339	338
PageBlocks-2c	10,4 %	0	6,23 %	6,23 %	312	311
Satimage	45,3 %	0	10,1 %	10,9 %	1212	943
Satimage-2c	10,7 %	5	2,25 %	2,75 %	321	161
Pendigits	36,5 %	0	1,66 %	1,69 %	738	702
Letter	64,9 %	0	5,8 %	5,82 %	6734	6737
Shuttle-2c	15,4 %	43	2,03 %	4,35 %	4325	2041
Shuttle	96,1 %	0	0,359 %	0,359 %	2702	2690

TAB. 5.7 – *Pré-filtrage par SVM: la colonne* sélection *donne le pourcentage d'exemples conservés par le pré-filtrage, la colonne* incohérent *donne le nombre d'exemples qui ne sont pas gardés, car ils sont du mauvais côté de la marge géométrique du SVM. Les colonnes e_V et n_{SV} correspondent au pourcentage d'erreurs et au nombre de vecteurs de support produit par la fonction de décision avec la base initiale et les colonnes e_{V_f} et n_{SV_f} ont la même signification que précédemment mais avec la base pré-simplifiée.*

des fonctions de décision produites par un SVM lorsque la base d'entraînement qui est ou n'est pas pré-filtrée (les valeurs des hyper-paramètres sont les mêmes dans les deux cas). Dans le cas où l'ensemble Z_- est vide, le taux d'erreur et le nombre de vecteurs de support devraient être identique. Les approximations numériques, avec un critère d'arrêt pour SMO de $\epsilon = 10^{-2}$, peuvent cependant produire des solutions légèrement différentes[14]. Globalement, cette différence, lorsqu'elle existe, a des conséquences qui sont quasiment négligeables sur les performances en généralisation.

Nous avons ensuite appliqué notre méthode de sélection d'exemples pertinents sur plusieurs bases d'apprentissage après la phase de pré-filtrage. Certaines bases (**Segment** à **pendigits** dans le tableau 5.8), précédemment exploitées sans pré-filtrage, ont été réutilisées pour observer les effets de la pré-simplification sur les capacités de généralisation. Les autres bases (**OptDigits** à **Satimages-2c** dans le tableau 5.8) sont de tailles trop importantes pour être utilisées sans pré-filtrage. Le tableau 5.8 donne les résultats obtenus par la combinaison de ces deux techniques. Ces résultats montrent que le pré-filtrage, tel qu'il est appliqué, dégrade la qualité des solutions obtenues pour les bases précédemment

14. Même avec un critère d'arrêt par défaut de SMO (*i.e.* $\epsilon = 10^{-3}$), il existe encore des différences entre les deux fonctions de décision, bien que globalement elles tendent à diminuer.

		1-ppv		SVM	
Base	critère	e_V	Z %	e_V	VS %
Segment	$\gamma_{\odot} \searrow \ominus$	14,29%	14,29%	5,714%	13,71%
Segment	$\gamma_{\bar{\Gamma}} \searrow \ominus$	5,714%	16,57%	11,43%	16%
Segment-2c	$\gamma_{\odot} \searrow \ominus$	14,29%	17,71%	20%	9,143%
Segment-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	25,71%	15,43%	31,43%	8,571%
Bupa	$\gamma_{\odot} \searrow \ominus$	40,3%	7,914%	37,31%	6,475%
Bupa	$\gamma_{\bar{\Gamma}} \searrow \ominus$	52,24%	15,47%	37,31%	12,59%
breast-cancer	$\gamma_{\odot} \searrow \ominus$	1,481%	4,745%	1,481%	0,5474%
breast-cancer	$\gamma_{\bar{\Gamma}} \searrow \ominus$	18,52%	0,9124%	20%	0,7299%
vehicle	$\gamma_{\odot} \searrow \ominus$	43,71%	11,34%	42,51%	8,1%
vehicle	$\gamma_{\bar{\Gamma}} \searrow \ominus$	43,71%	17,97%	25,75%	15,76%
vehicle-2c	$\gamma_{\odot} \searrow \ominus$	25,15%	7,511%	22,16%	5,007%
vehicle-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	26,95%	6,48%	10,18%	6,038%
pendigits-2c	$\gamma_{\odot} \searrow \ominus$	3,439%	2,373%	4,814%	0,5131%
pendigits-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	2,338%	0,3849%	3,164%	0,2566%
OptDigits	$\gamma_{\odot} \searrow \ominus$	*13,31*%	10,48%	*9,881*%	7,604%
OptDigits	$\gamma_{\bar{\Gamma}} \searrow \ominus$	*13,7*%	8,649%	*6,588*%	7,082%
serous	$\gamma_{\odot} \searrow \ominus$	45,41%	**11,57**%	36,86%	**8,604**%
serous	$\gamma_{\bar{\Gamma}} \searrow \ominus$	*52,68*%	16,39%	37,63%	14,56%
PageBlocks-2c	$\gamma_{\odot} \searrow \ominus$	**5,958**%	**3,492**%	9,991%	1,461%
PageBlocks-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	8,433%	8,124%	9,991%	2,236%
satimage	$\gamma_{\odot} \searrow \ominus$	11,1%	**6,223**%	*80,15*%	5,547%
satimage	$\gamma_{\bar{\Gamma}} \searrow \ominus$	13,65%	15,17%	*80,15*%	15,04%
satimage-2c	$\gamma_{\odot} \searrow \ominus$	3,6%	11,82%	*46,1*%	11,32%
satimage-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	4,05%	8,185%	*53,9*%	3,292%

TAB. 5.8 – *Taux d'erreur et pourcentage d'exemples utilisés par la fonction de décision d'un ppv ou SVM avec pré-simplification et arrêt prématuré.*

utilisées (*cf.* tableau 5.4). Il est donc préférable de ne pas utiliser cette méthode de pré-filtrage lorsque la taille de la base n'est pas problématique. Pour les autres bases, lorsque l'on compare les résultats du tableau 5.8 à ceux des tableaux A.4 et A.5, on remarque une réduction importante du nombre d'exemples avec des taux d'erreur acceptables pour la règle des PPV, mais pour les SVM les résultats sont catastrophiques et généralement pires que ceux avec la règle PPV, alors que c'était l'effet contraire qui était auparavant observé. Il est possible de supposer que les hyper-paramètres des SVM utilisés avec les bases non filtrées ne sont plus appropriés avec les bases simplifiées (pré-filtrage et sélection des exemples les plus pertinents). Les remarques précédentes (*cf.* section 5.3.1.6) ont déjà mis en évidence cette possibilité.

Nous avons alors procédé à une phase de sélection d'hyper-paramètres en utilisant les deux mêmes procédures que celles utilisées à la section 5.3.1.6. Les tableaux 5.9 et 5.10 correspondent aux résultats produits par ces deux procédures de sélection des hyper-paramètres. A nouveau la sélection par validation croisée est plus performante que la sélection par minimisation du nombre de vecteurs de support. Lorsque les résultats du tableau 5.10 (validation croisée en 10 parties) sont comparés à ceux du tableau A.5, les avantages de la mise en cascade du pré-filtrage et de la sélection d'exemples pertinents sont globalement positifs, même si la méthode actuelle de pré-filtrage perturbe l'efficacité de la sélection d'exemples pertinents.

		Avant		Après minSV			
Base	critère	e_V	VS %	e_V	VS %	C	σ
Segment	$\gamma_{\odot} \searrow \ominus$	5,714%	13,71%	5,714%	13,71%	100	2,2
Segment	$\gamma_{\bar{\Gamma}} \searrow \ominus$	11,43%	16%	8,571%	16%	10	0,71
Segment-2c	$\gamma_{\odot} \searrow \ominus$	20%	9,143%	17,14%	8%	1000	7,1
Segment-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	31,43%	8,571%	31,43%	6,286%	1000	2,2
Bupa	$\gamma_{\odot} \searrow \ominus$	37,31%	6,475%	37,31%	4,317%	1000	2,2
Bupa	$\gamma_{\bar{\Gamma}} \searrow \ominus$	37,31%	12,59%	34,33%	8,993%	100	0,71
Breast-cancer	$\gamma_{\odot} \searrow \ominus$	1,481%	0,5474%	65,19%	0,365%	0,01	71
Breast-cancer	$\gamma_{\bar{\Gamma}} \searrow \ominus$	20%	0,7299%	65,19%	0,365%	0,01	71
Vehicle	$\gamma_{\odot} \searrow \ominus$	42,51%	8,1%	40,72%	7,364%	1000	7,1
Vehicle	$\gamma_{\bar{\Gamma}} \searrow \ominus$	25,75%	15,76%	25,75%	13,7%	1000	2,2
Vehicle-2c	$\gamma_{\odot} \searrow \ominus$	22,16%	5,007%	4,192%	3,829%	1000	2,2
Vehicle-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	10,18%	6,038%	5,389%	4,418%	1000	2,2
Pendigits-2c	$\gamma_{\odot} \searrow \ominus$	4,814%	0,5131%	49,93%	0,2566%	0,01	71
Pendigits-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	3,164%	0,2566%	50,07%	0,2566%	0,01	71
OptDigits	$\gamma_{\odot} \searrow \ominus$	9,881%	7,604%	11,73%	6,103%	1000	22
OptDigits	$\gamma_{\bar{\Gamma}} \searrow \ominus$	6,588%	7,082%	8,959%	6,168%	1000	22
Serous	$\gamma_{\odot} \searrow \ominus$	36,86%	8,604%	39,41%	6,934%	1000	7,1
Serous	$\gamma_{\bar{\Gamma}} \searrow \ominus$	37,63%	14,56%	41,84%	12,83%	1000	7,1
PageBlocks-2c	$\gamma_{\odot} \searrow \ominus$	9,991%	1,461%	5,591%	0,2738%	1000	2,2
PageBlocks-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	9,991%	2,236%	15,67%	0,6162%	1000	2,2
Satimage	$\gamma_{\odot} \searrow \ominus$	80,15%	5,547%	12,3%	4,487%	10	71
Satimage	$\gamma_{\bar{\Gamma}} \searrow \ominus$	80,15%	15,04%	18,35%	11,7%	1000	71
Satimage-2c	$\gamma_{\odot} \searrow \ominus$	46,1%	11,32%	3,5%	0,8343%	10	71
Satimage-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	53,9%	3,292%	3,5%	1,263%	10	71

TAB. 5.9 – *Importance de la sélection d'un modèle efficace sur une base simplifiée (minimum de vecteurs de support).*

		Avant		Après avec 10 CV			
Base	critère	e_V	VS %	e_V	VS %	C	σ
Segment	$\gamma_{\odot} \searrow \ominus$	5,714%	13,71%	5,714%	14,29%	10	0,71
Segment	$\gamma_{\bar{\Gamma}} \searrow \ominus$	11,43%	16%	8,571%	16%	10	0,71
Segment-2c	$\gamma_{\odot} \searrow \ominus$	20%	9,143%	14,29%	12,57%	1	0,71
Segment-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	31,43%	8,571%	22,86%	6,857%	1000	7,1
Bupa	$\gamma_{\odot} \searrow \ominus$	37,31%	6,475%	37,31%	7,914%	1	0,22
Bupa	$\gamma_{\bar{\Gamma}} \searrow \ominus$	37,31%	12,59%	29,85%	10,43%	1000	2,2
Breast-cancer	$\gamma_{\odot} \searrow \ominus$	1,481%	0,5474%	1,481%	0,5474%	10	0,71
Breast-cancer	$\gamma_{\bar{\Gamma}} \searrow \ominus$	20%	0,7299%	20%	0,7299%	10	0,71
Vehicle	$\gamma_{\odot} \searrow \ominus$	42,51%	8,1%	38,92%	8,1%	100	2,2
Vehicle	$\gamma_{\bar{\Gamma}} \searrow \ominus$	25,75%	15,76%	24,55%	14,43%	100	2,2
Vehicle-2c	$\gamma_{\odot} \searrow \ominus$	22,16%	5,007%	4,192%	3,829%	1000	2,2
Vehicle-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	10,18%	6,038%	4,79%	5,007%	100	2,2
Pendigits-2c	$\gamma_{\odot} \searrow \ominus$	4,814%	0,5131%	7,978%	1,09%	10	71
Pendigits-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	3,164%	0,2566%	2,889%	0,3207%	1	71
OptDigits	$\gamma_{\odot} \searrow \ominus$	9,881%	7,604%	10,54%	7,148%	10	2,2
OptDigits	$\gamma_{\bar{\Gamma}} \searrow \ominus$	6,588%	7,082%	7,115%	6,821%	10	2,2
Serous	$\gamma_{\odot} \searrow \ominus$	36,86%	8,604%	36,35%	9,108%	10	7,1
Serous	$\gamma_{\bar{\Gamma}} \searrow \ominus$	37,63%	14,56%	34,57%	15,35%	1	2,2
PageBlocks-2c	$\gamma_{\odot} \searrow \ominus$	9,991%	1,461%	5,225%	0,7531%	1	0,71
PageBlocks-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	9,991%	2,236%	5,5%	3,332%	1	0,22
Satimage	$\gamma_{\odot} \searrow \ominus$	80,15%	5,547%	12,3%	4,487%	10	71
Satimage	$\gamma_{\bar{\Gamma}} \searrow \ominus$	80,15%	15,04%	12,35%	12,2%	10	71
Satimage-2c	$\gamma_{\odot} \searrow \ominus$	46,1%	11,32%	3,5%	0,8343%	10	71
Satimage-2c	$\gamma_{\bar{\Gamma}} \searrow \ominus$	53,9%	3,292%	3,4%	1,601%	1	71

TAB. 5.10 – *Importance de la sélection d'un modèle efficace sur une base simplifié (validation croisé en 10 parties).*

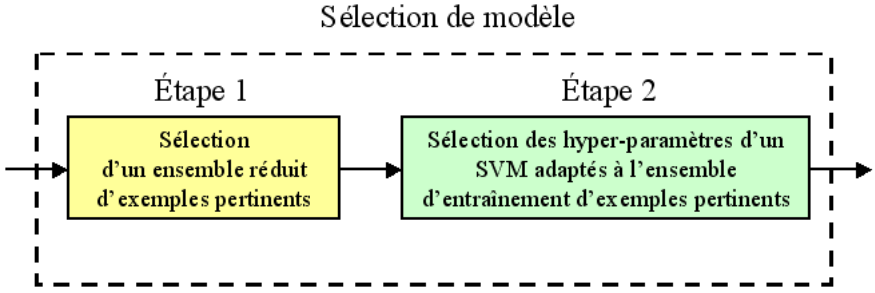


FIG. 5.5 – *Synopsis des deux étapes de la sélection de modèle en cascade.*

5.3.1.9 Schéma de simplification en cascade

A partir des résultats et remarques liés aux différentes expérimentations présentées précédemment, nous avons défini un schéma de simplification en cascade qui a pour but de réaliser la sélection d'un modèle performant pour la production de fonctions de décision de complexité réduite avec des SVM. La figure 5.5 illustre les deux étapes principales de la sélection de modèle en cascade relative à ce schéma de simplification. La première étape de ce schéma réalise la sélection d'un sous-ensemble d'exemples pertinents. La seconde étape réalise la recherche des hyper-paramètres optimaux lorsque les SVM sont entraînés à partir de l'ensemble d'exemples pertinents sélectionnés par la première étape.

La figure 5.6 donne les détails correspondant au schéma complet de simplification. La phase optionnelle de pré-filtrage de la section 5.3.1.7 n'est utilisée que si le nombre d'exemples de la base d'apprentissage est supérieur à 1500 exemples. A partir de la base initiale (ou de la base pré-filtrée), deux recherches d'un sous-ensemble d'exemples pertinents sont réalisées à partir de la méthode générique décrite en section 5.3.1.2. Chacune de ces deux recherches utilise un critère d'hypothèses différent (ainsi que le CAP associé) tels qu'ils sont décrits dans les sections 5.3.1.3 et 5.3.1.5 (*i.e.* $[\gamma_{\odot} \searrow \ominus]$ et $[\gamma_{\bar{\Gamma}} \searrow \ominus]$). La raison de l'exploitation de ces deux critères est que les résultats expérimentaux des sections 5.3.1.8 et 5.3.1.8 n'ont pas permis de mettre en évidence la supériorité d'un de ces critères par rapport à l'autre, ils ont tous les deux leurs avantages. Pour chaque base d'apprentissage simplifiée produite par ces deux procédures, une recherche des hyper-paramètres optimaux (C,σ) pour un SVM (ou ensemble de SVM si le problème est multi-classe) exploitant un noyau Gaussien est réalisée à partir d'une procédure de validation croisée en 10 parties. La base simplifiée qui permet d'obtenir l'erreur de validation croisée la plus faible est sélectionnée, ainsi que les hyper-paramètres correspondants. Lorsque l'écart est faible entre les estimations de l'erreur de validation croisée (inférieur à 1% en écart relatif) avec ces deux critères, c'est la base simplifiée de taille la plus faible qui est sélectionnée. Ceci afin de favoriser la réduction de la complexité de la fonction de décision produite, lorsque l'écart sur l'estimation de l'erreur de généralisation n'est pas significatif. La base simplifiée Z^p et les hyper-paramètres (C,σ) sélectionnés sont utilisés pour produire, à partir de l'algorithme des SVM, une fonction de décision simplifiée. Lorsque la base d'apprentissage comporte plus de deux classes, c'est un schéma de combinaison *un contre un* qui est utilisé pour produire la fonction de décision simplifiée et chaque SVM binaire intervenant dans ce schéma de combinaison utilise les mêmes hyper-paramètres (C,σ).

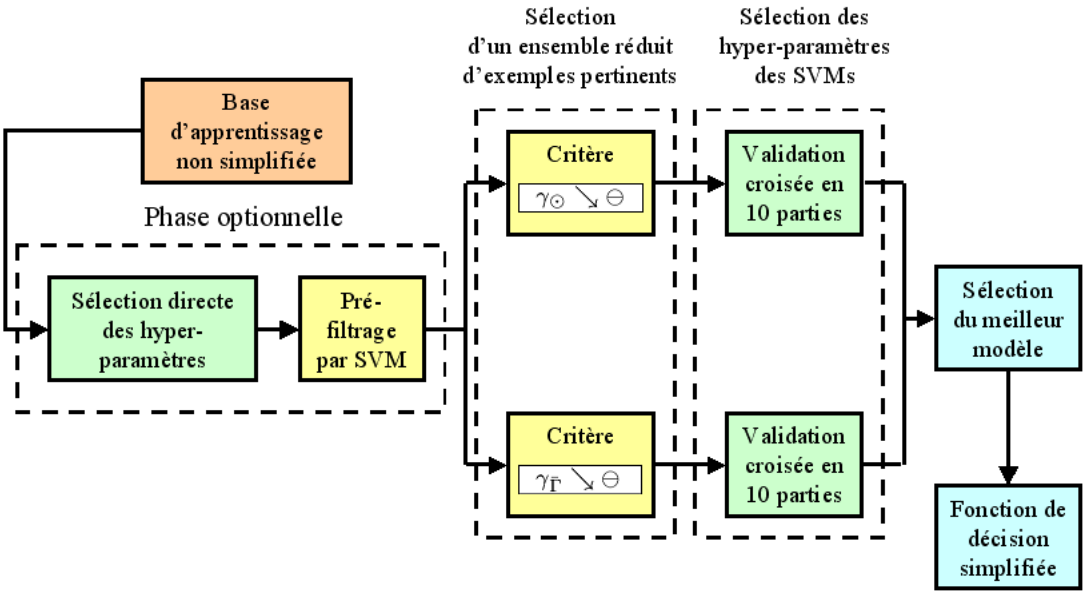


FIG. 5.6 – *Schéma en cascade détaillé de sélection de modèle pour produire des fonctions de décision de complexité réduite et performante avec des SVM.*

Globalement ce schéma réalise un apprentissage en cascade correspond à une sélection de modèle qui inclut les hyper-paramètres des SVM et la sélection d'un ensemble réduit de prototypes représentatifs de la base d'apprentissage. Pour ce schéma, les prototypes sont forcément des exemples de la base d'apprentissage (la seconde approche est différente, notamment à ce niveau là). Les risques de sur-apprentissage liés à l'exploitation d'un espace de modèles de dimensions importantes sont contrôlés par l'utilisation de plusieurs principes d'inférences. La sélection d'un sous-ensemble d'exemples utilise un principe SRM basé sur un schéma de compression guidé par l'exploitation de marges d'hypothèse. La sélection des hyper-paramètres utilise un principe de minimisation de l'erreur empirique à travers une procédure de validation croisée. Pour chaque couple d'hyper-paramètres utilisés, le principe SRM des SVM est également utilisé pour construire la fonction de décision à partir des données de la base d'entraînement simplifiée.

Expérimentations du schéma de simplification en cascade

Nous avons appliqué notre schéma de simplification en cascade à plusieurs des bases de données précédemment utilisées. Le tableau 5.11 permet de réaliser une comparaison entre un apprentissage classique avec des SVM et l'utilisation de notre schéma de simplification. Comme dans la section 5.3.1.6, la phase de pré-filtrage n'est pas utilisée pour les bases **QIjpeg2M** à **Pendigits-2c** du tableau 5.11 et elle l'est pour les autres bases présentes en fin du tableau 5.11. Globalement ces résultats montrent que notre schéma de simplification en cascade permet de réduire significativement le nombre d'exemples utilisés par la fonction de décision, ce qui peut avoir un impact très significatif lorsque des contraintes de temps réel doivent être pris en compte. Dans la grande majorité des cas, cette réduction de complexité s'accompagne d'une augmentation de la capacité de généralisation mesurée à partir du taux d'erreur sur la base de validation, ce qui illustre l'intérêt d'exploiter un

ensemble réduit d'exemples pertinents. Les deux critères $[\gamma_{\odot} \searrow \ominus]$ et $[\gamma_{\bar{\Gamma}} \searrow \ominus]$ ont des efficacités différentes suivant la nature de la base utilisée. Le critère $[\gamma_{\odot} \searrow \ominus]$ semble être plus performant sur les bases d'apprentissage de plus petites tailles, mais lorsqu'on regarde plus en détails les résultats des tableaux 5.6 et 5.10, l'écart entre ces deux critères n'est pas toujours significatif par rapport au taux d'erreur. Ceci semble indiquer que chacun de ces deux critères est globalement performant lorsque seules les capacités de généralisation sont considérées. Le tableau 5.6 illustre par contre que le critère $[\gamma_{\odot} \searrow \ominus]$ produit systématiquement un ensemble d'exemples de taille plus petite ou identique à celui produit avec le critère $[\gamma_{\bar{\Gamma}} \searrow \ominus]$. Suivant le principe SRM, il est logique de choisir l'ensemble qui comporte le moins d'exemples pour des capacités en généralisation identiques ou très proches. Ces remarques expliquent l'apparent avantage du critère $[\gamma_{\odot} \searrow \ominus]$ sur le critère $[\gamma_{\bar{\Gamma}} \searrow \ominus]$ avec les bases en début du tableau 5.11.

La comparaison des tableaux 5.6 et 5.11 montre que ce n'est pas toujours le meilleur critère qui est choisi vis-à-vis de ses capacités de généralisation estimées avec la base de validation (bases **Pima** et **PageBlock-2c**). Comme ni la procédure de validation croisée, ni l'utilisation d'un ensemble réduit d'exemples dans une base de validation, ne sont des estimateurs parfaits de l'erreur de généralisation, il est logique d'observer cet écart dans la sélection d'un des deux critères. Cependant, globalement l'erreur estimée avec la procédure de validation croisée, pour la sélection d'un critère, est fortement corrélée au taux d'erreur sur la base de validation pour ce critère.

Lorsque la phase de préfiltrage doit être réalisée, le schéma d'apprentissage semble moins performant dans ses capacités à réduire à la fois le taux d'erreur sur la base de validation et le nombre d'exemples utilisés, même si globalement il reste efficace. Pour la base *Serous* le nombre de vecteurs de support est divisé par 4 sans réduire les capacités de généralisation de la fonction de décision produite. Cependant, si pour la base *OptDigits* la réduction du nombre de vecteurs de support est également de 4, le taux d'erreur est lui multiplié par 4. La première raison de cette non-adéquation peut être que la méthode de pré-filtrage n'est pas efficace avec les critères d'hypothèses utilisés, s'ils ont besoin de suffisament d'exemples éloignés de la marge géométrique pour être efficaces. Une raison est liée à la difficulté à réaliser un apprentissage efficace avec peu d'exemples, lorsque le nombre d'attributs est conséquent par rapport à la taille de la base d'apprentissage. En effet, à partir du tableau 5.8, on en déduit que la base d'apprentissage après sélection des exemples pertinents contient 265 exemples pour un problème à 10 classes, soit seulement environ 53 exemples par problème binaire alors que chaque exemple est caractérisé par 64 attributs. Même si la méthode a sélectionné les exemples les plus pertinents, le problème de la malédiction de la dimensionnalité peut intervenir, surtout si certains de ces attributs ne sont pas représentatifs pour plusieurs des problèmes binaires induits par le schéma de combinaison binaire.

5.3.1.10 Discusion

A travers cette première approche où nous nous sommes intéressés à réduire la complexité des fonctions de décision produites pour des SVM, nous avons montré que la recherche d'exemples pertinents peut-être efficace pour réaliser cette simplification, même si la notion d'exemple pertinent est un concept difficile à formaliser. Nous avons entre autres montré, que dans le cadre de l'apprentissage supervisé, la notion de pertinence était

	classique				avec notre schéma de simplification				
Base	C	σ	e_V	VS %	C	σ	critère	e_V	VS %
QIjpeg2M	10	1,3	22%	60%	10	2,2	$[\gamma_{\odot} \searrow \ominus]$	**17,12**%	**8,621**%
QIjpeg2M-2c	10	1,6	0,9%	13%	10	0,71	$[\gamma_{\odot} \searrow \ominus]$	**0,0**%	**1,724**%
Iris	10	2	0%	22%	10	7,1	$[\gamma_{\odot} \searrow \ominus]$	0,0%	**4,167**%
Iris-2c	10	2,4	0%	3,3%	10	7,1	$[\gamma_{\odot} \searrow \ominus]$ & $[\gamma_{\bar{\Gamma}} \searrow \ominus]$	0,0%	**1,667**%
Wine	10	1,1	0%	28%	100	2,2	$[\gamma_{\odot} \searrow \ominus]$	0,0%	**8,333**%
Wine-2c	10	1,2	0%	19%	10	2,2	$[\gamma_{\bar{\Gamma}} \searrow \ominus]$	0,0%	**7,639**%
Glass	10	0,88	21%	72%	10	0,71	$[\gamma_{\odot} \searrow \ominus]$	*23,08*%	**8,571**%
Glass-2c	10	0,91	13%	40%	0,01	0,71	$[\gamma_{\odot} \searrow \ominus]$	**10,26**%	**5,714**%
Segment	10	1,1	2,9%	58%	1000	7,1	$[\gamma_{\odot} \searrow \ominus]$	**2,857**%	**7,429**%
Segment-2c	10	1,3	8,6%	24%	10	0,71	$[\gamma_{\bar{\Gamma}} \searrow \ominus]$	**5,714**%	**19,43**%
Bupa	10	0,82	31%	62%	10	0,71	$[\gamma_{\bar{\Gamma}} \searrow \ominus]$	**23,88**%	**24,46**%
Ionosphere	10	2,1	4,3%	33%	10	0,71	$[\gamma_{\bar{\Gamma}} \searrow \ominus]$	*7,246*%	**17,08**%
Breast-cancer	10	0,61	2,2%	11%	10	2,2	$[\gamma_{\odot} \searrow \ominus]$	**1,481**%	**2,007**%
OptDigits-2c	10	1,9	0%	6%	10	2,2	$[\gamma_{\odot} \searrow \ominus]$ & $[\gamma_{\bar{\Gamma}} \searrow \ominus]$	0%	**1,794**%
Pima	10	0,91	23%	54%	100	2,2	$[\gamma_{\odot} \searrow \ominus]$	**21,71**%	**3,084**%
Vehicle	10	1,4	17%	59%	100	2,2	$[\gamma_{\bar{\Gamma}} \searrow \ominus]$	**16,67**%	**20,62**%
Vehicle-2c	10	1,5	1,8%	20%	100	2,2	$[\gamma_{\bar{\Gamma}} \searrow \ominus]$	**0,5988**%	**8,837**%
Letter-2c	10	1,2	0,97%	4,3%	10	0,71	$[\gamma_{\bar{\Gamma}} \searrow \ominus]$	**0,3226**%	**2,651**%
Pendigits-2c	10	140	1,2%	1,5%	1	71	$[\gamma_{\bar{\Gamma}} \searrow \ominus]$	**0,2751**%	**0,8339**%
OptDigits	10	1,8	1,6%	30%	10	2,2	$[\gamma_{\bar{\Gamma}} \searrow \ominus]$	*7,115*%	**6,821**%
Serous	10	2	35%	62%	1	2,2	$[\gamma_{\bar{\Gamma}} \searrow \ominus]$	**34,57**%	**15,35**%
PageBlocks-2c	10	1	6,2%	7,1%	1	0,22	$[\gamma_{\bar{\Gamma}} \searrow \ominus]$	**5,5**%	**3,332**%
Satimage	10	100	10%	27%	10	71	$[\gamma_{\odot} \searrow \ominus]$	*12,3*%	**4,487**%
Satimage-2c	10	130	2,3%	7,2%	10	71	$[\gamma_{\odot} \searrow \ominus]$	*3,5*%	**0,8343**%

TAB. 5.11 – *Comparaison entre un apprentissage classique avec des SVM et notre schéma de simplification en cascade.*

en partie dépendante du principe d'inférence utilisé dans l'exploitation de la règle PPV et l'utilisation d'un principe SRM. Si les résultats obtenus grâce à notre schéma de simplification en cascade sont encourageants, plusieurs remarques et perspectives peuvent être évoquées :

Coût d'utilisation de la règle PPV

L'utilisation de la règle PPV est actuellement lourde en temps de calcul ce qui nous limite à des bases de taille réduite. Il devient alors évident qu'il sera nécessaire d'utiliser des méthodes algorithmiques plus performantes pour dépasser ces limitations. Dans [SHIN03], plusieurs références sont données sur des travaux permettant ces optimisations. L'utilisation de ces techniques permettrait d'utiliser notre schéma d'apprentissage avec des bases de tailles plus importantes (sans avoir recours à la notre technique de pré-filtrage qui n'est pas sans poser certains problèmes).

Remplacer la règle PPV par un SVM

La détermination de l'ensemble candidats Z_i^c à chaque itération utilise actuellement la règle PPV induite par Z_i^p. Une autre possibilité serait d'utiliser directement la fonction de décision d'un SVM produite à partir de Z_i^p. En effet des résultats expérimentaux montrent qu'elle est généralement plus performante que la règle PPV sur un même ensemble d'apprentissage. La principale raison (déjà évoquée) est que la frontière de décision d'un SVM est moins discontinue que celle relative à la règle PPV. Comme le but final est de produire des fonctions de décision performantes avec les SVM, il semble plus logique de les uti-

liser pour la sélection d'exemples pertinents. De plus, comme à chaque itération de notre méthode de sélection un seul exemple est ajouté, les techniques d'*alpha seeding* peuvent être facilement exploitées. Chaque nouvelle phase d'entraînement d'un SVM aurait alors un coût calculatoire faible. Cependant, l'utilisation de la fonction de décision produite par un SVM a également des désavantages :

- Le premier désavantage concerne l'importance du choix des hyper-paramètres des SVM pour la détermination de la cohérence. Nous avons vu précédemment que les SVM sont d'autant plus sensibles à ces réglages que le nombre d'exemples d'entraînement est faible. Le changement de la nature de la base d'entraînement (ajout d'exemples pertinents) peut également remettre en question ces choix. De plus, la sélection des hyper-paramètres par validation croisée a un coût important, en particulier si elle doit être réalisée à chaque ajout d'un exemple pertinent par notre méthode.
- Le second désavantage est que la fonction de décision produite par un SVM est binaire et nécessite l'utilisation d'un schéma de combinaison pour traiter des problèmes multi-classes, alors que la règle PPV peut être directement utilisée avec des problèmes multi-classes.

Cependant, l'utilisation des méthodes présentées dans [AYAT05] et dans [HASTIE04B] (*cf.* section 5.2.3), en les combinant avec des techniques d'*alpha seeding* (*cf.* chapitre 4), permettrait de réaliser la mise à jour des hyper-paramètres à moindre coût à chaque ajout d'un nouvel exemple pertinent par notre méthode et ainsi de réduire l'importance de ce premier désavantage. L'évolution de la marge géométrique d'un SVM et du nombre de vecteurs de support pourrait également servir à définir un nouveau CAP pour notre méthode.

Pour le désavantage lié à la nature multi-classe de la plupart des bases d'apprentissage, il est possible de l'éviter en réalisant la recherche d'exemples pertinents pour chaque problème binaire induit par le schéma multi-classe. Cette façon de procéder permet de prendre en compte les particularités de chaque problème de discrimination. En effet, la seconde approche pour produire des fonctions de décision simplifiées (*cf.* section 5.3.2) illustre que l'importance de la simplification est dépendante de la nature du problème de discrimination. De plus, plusieurs travaux [RIFKIN04, RAJAN04] insistent sur l'importance d'avoir une sélection des hyper-paramètres pour chaque SVM binaire inclut dans un schéma de combinaison[15]. Pour la plupart des schémas de décomposition multi-classe, chaque problème binaire dispose d'une base de taille plus faible que la base multi-classe initiale[16]. Grâce à cette possibilité, il devient possible d'exploiter des bases de taille plus importante sans avoir recours à une technique de pré-filtrage.

Globalement l'utilisation de la fonction de décision d'un SVM pour faciliter la sélection des exemples pertinents ouvre des perspectives intéressantes mêmes si des investigations supplémentaires doivent être réalisées.

Améliorations du critère $[\gamma_{\bar{\Gamma}} \searrow]$

15. La seconde partie du chapitre 6 illustre également cette importance.
16. La décomposition *un-contre-un* est celle qui favorise au maximum cet effet.

Les valeurs de marge d'hypothèses de la relation (5.14) sont actuellement utilisées par un simple calcul de moyenne. Si les résultats obtenus en exploitant la valeur de $\bar{\Gamma}$ sont encourageants, la prise en compte du nombre d'exemples qui ont z comme plus proche voisin et la répatition des valeurs d'hypothèses pour ces exemples doivent être utilisées plus finement. Un autre point à prendre en compte est que les valeurs de Γ sont calculées à partir de l'espace initial, il serait plus judicieux d'utiliser les valeurs des distances dans l'espace de redistribution à partir de l'utilisation de la relation 4.58. Ce passage sera de toute façon obligatoire dès que notre méthode devra être utilisée avec des fonctions noyaux différentes d'un noyau gaussien.

Combinaison des deux types de critères

L'utilisation de la distance entre deux exemples de classes différentes, avec le critère $[\gamma \nearrow]$, pour caractériser la proximité de ces deux exemples à la frontière de décision pose actuellement problème car elle peut choisir des couples d'exemples peu pertinents dès les premières itérations. Une possibilité d'amélioration pour éviter cet effet consisterait à obliger les deux exemples proches à avoir respectivement une valeur importante du critère $[\gamma_{\bar{\Gamma}} \searrow]$. La définition d'une marge de confiance hybride qui prenne en compte des mesures de marge géométrique relative à des couples d'exemples et de marge d'hypothèses relative à des singletons d'exemples permettrait de définir un compromis efficace entre ces deux notions essentielles de la pertinence des exemples que sont l'importance dans la définition de la frontière de décision et la représentativité d'un grand nombre d'autres exemples de sa classe. Les spécifications d'un tel critère de confiance restent à définir.

Améliorations des critères d'arrêt prématurés

Les critères d'arrêt proposés pour éviter d'ajouter des exemples peu pertinents dans la base simplifiée sont actuellement assez simples. Pour le critère basé sur le nombre de voisins les plus proches et de même classe, le choix d'une valeur arbitraire de 2 pour k doit pouvoir être amélioré. Il serait souhaitable de le faire dépendre de la nature de la base d'apprentissage. Pour le critère basé sur $\Gamma(z_1)$, le choix du critère d'arrêt va dépendre de l'utilisation plus fine de $\Gamma(z_1)$ qu'une simple valeur moyenne.
Un autre point à prendre en compte est qu'actuellement l'évolution de l'erreur empirique de la règle PPV pour déterminer la cohérence n'est pas prise en compte. Les figures C.1 à C.10 illustrent une forte corrélation entre le taux d'erreur sur la base d'apprentissage et de validation pour les premières itérations de notre méthode. Il serait donc utile de réfléchir à comment prendre en compte l'évolution de cette mesure d'erreur dans la définition d'un critère d'arrêt prématuré, tout en tenant compte des risques de sur-apprentissage.

Pour conclure sur cette première approche, au-delà des résultats prometteurs déjà évoqués, l'ensemble des perspectives qui viennent d'être citées suggère des possibilités accrues de notre méthode d'apprentissage.

5.3.2 Prototypage de la base d'apprentissage

Nous avons vu avec la première approche qu'il n'était pas nécessaire d'utiliser l'ensemble des données d'apprentissage, mais de choisir un ensemble réduit d'exemples représentatifs du problème d'apprentissage, pour produire une fonction de décision performante avec l'algorithme des SVM (ou la règle PPV). Bien entendu, plus il y a de données pour décrire un problème d'apprentissage et plus les possibilités pour produire une fonction de décision performante à partir de ces données sont grandes. La sélection d'un sous-ensemble réduit d'exemples (ou d'attributs) est lui-même un processus d'apprentissage et elle doit utiliser un principe d'inférence robuste pour garantir de bonnes capacités en généralisation. La première approche est basée sur la définition d'un schéma de compression correspondant à un principe de minimisation du risque structurel. Dans cette seconde approche nous nous intéressons aux techniques d'apprentissage non supervisé pour produire un ensemble réduit de prototypes à partir d'un ensemble plus conséquent de données d'apprentissage. Le domaine de l'apprentissage non supervisé est vaste et propose de nombreuses méthodes pour regrouper et résumer un ensemble de données dans des catégories de forte similarité interne [JAIN99, MEILĂ05, XU05]. Le choix d'une méthode de catégorisation par rapport à une autre dépend généralement de connaissances sur la nature du problème à traiter [XU05, TAN06]. Notre but, comme pour la première approche, est de produire un ensemble de prototypes suffisamment représentatifs de la base de données. Nous nous intéressons donc aux méthodes de catégorisation qui peuvent produire ces prototypes. Ces prototypes sont ensuite utilisés pour entraîner un SVM.

Dans cette seconde approche, les prototypes ne sont plus nécessairement des exemples de la base d'apprentissage, mais des données synthétisées à partir d'elle. Notre but est également de permettre à l'utilisateur de définir l'importance du compromis : réduction de la complexité et performances en généralisation. Dans le cadre d'applications qui ont des contraintes de temps réel, il peut être acceptable de diminuer légèrement les performances en généralisation si cela correspond à une baisse importante de la complexité des fonctions de décision produites. Pour cette seconde approche nous souhaitons également réaliser la sélection des attributs les plus pertinents avec toujours l'objectif de réduire la complexité des fonctions de décision produites et par la même occasion les problèmes liés à la malédiction de la dimensionnalité. L'utilisateur peut, comme pour les prototypes, définir un compromis entre réduction du nombre d'attributs utilisés et performances en généralisation. Ce compromis entre complexité et pouvoir de prédiction est réalisé à partir de la définition d'un nouveau critère de qualité relatif aux fonctions de décision produites (*cf.* section 5.3.2.4).

Nous avons également pour objectif que notre méthode soit suffisamment générique pour tirer profit de différentes techniques de classification non-supervisée. Les algorithmes de partitionnement ont pour premier but de regrouper ensemble des données en fonction de leurs similarités. Dans la plupart de ces algorithmes ce regroupement est réalisé à partir de la construction d'une structure de données qui résume l'information contenue dans une base par la production d'un ensemble de prototypes. Les algorithmes suivants font partie de cette catégorie: les centres mobiles [FORGY65, MACQUE67], la quantification vectorielle [GERSHO92], la classification floue [JAIN99], les réseaux à compétition et les cartes de Kohonen [KOHONE01], CURE (Clustering Using REpresentatives) [GUHA98], BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) [ZHANG96], des variantes à base de méta-heuristiques [JAIN99, RÖ04]. D'autres al-

gorithmes de catégorisation seraient facilement adaptables pour produire un ensemble réduit de prototypes par simple seuillage des caractéristiques attribuées aux exemples. Parmi ceux-ci, il y a : DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [ESTER96], DENCLUE (DENsity-based CLUstEring) [HINNEB98], Chamelon [KARYPI99], STING (STatistical INformation Grid) [WANG97], méthodes avec fonctions noyaux [TAX99, SCHÖL99, CHEN01, BEN-HU01]. Une autre possibilité pour choisir un sous-ensemble réduit de prototypes est de simplement réaliser un échantillonnage par tirage aléatoire dans chaque regroupement produit par une méthode de catégorisation qui ne construit pas directement un ensemble de prototypes. Quelle que soit la technique utilisée, le but est de produire un ensemble de prototypes représentatifs de la diversité des données liée au problème d'apprentissage, tout en réduisant la redondance inutile parmi ces données. Suivant la méthode de catégorisation utilisée, les données aberrantes peuvent également être plus ou moins ignorées, suivant la difficulté pour les prototypes de les représenter. Le nombre de prototypes produits et leurs qualités dépendent du choix d'un ensemble de paramètres propres à chaque algorithme de catégorisation. L'évaluation de la qualité d'une partition peut être réalisée suivant différents type de critères . La plupart de ces critères prennent en compte la complexité de la solution produite (nombre de prototypes ou de regroupements créés) et l'adéquation aux données. Cette adéquation aux données peut être difficile à mesurer si aucune base de test n'existe avec l'information sur la classe des exemples. Cela est encore plus difficile si le nombre de classes à identifier n'est pas connu. Ces différentes remarques expliquent que le nombre de critères de qualité que l'on trouve dans la littérature est très important [JAIN88, JAIN99, XU05, HALKID01, TAN06] suivant la nature du problème de catégorisation abordé. Comme évoqué précédemment, nous avons défini notre propre critère de qualité plutôt que d'utiliser un de ceux déjà existants. La raison est que notre méthode, même si elle exploite les techniques de catégorisation, est de nature différente à celle d'une catégorisation classique.

Le but de cette seconde approche étant de produire une fonction de décision performante avec les SVM à partir d'un ensemble réduit de prototypes, le choix des hyperparamètres des SVM, en plus de ceux liés à la production d'un ensemble réduit de prototypes efficaces, est important. Pour limiter la complexité de l'espace des modèles lié à cette optimisation, il est important de choisir une méthode de catégorisation qui puisse produire des ensembles de prototypes résumant plus ou moins fidèlement les données d'apprentissage à travers l'exploitation d'un nombre réduit de paramètres. Afin de simplifier cette première étude, nous avons choisi de nous limiter à l'exploitation de méthodes de catégorisation qui utilisent un seul paramètre. Cependant notre objectif à plus long terme est de pouvoir appliquer notre schéma d'apprentissage à différentes méthodes de catégorisation qui peuvent comporter un plus grand nombre de paramètres de réglage. A partir d'une analyse bibliographique de différentes techniques de catégorisation, notre choix s'est porté sur une version de l'algorithme LBG (Linde, Buzo et Gray [LINDE80]) issu de la quantification vectorielle. Cet algorithme produit 2^k prototypes par classe à partir de k itérations. Ses avantages sont :

- La production de prototypes est contrôlée par un seul paramètre k.
- La possibilité produire une simplification extrême qui correspond à n'avoir qu'un seul prototype par classe, ainsi que l'autre extrême qui correspond à ne réaliser aucune simplification.
- Il ne suppose pas de connaissance sur la nature des données à traiter.

– Les prototypes sont majoritairement dans les régions qui ont une forte densité d'exemples.

– Cette technique est utilisée dans différents domaines liés à la compression.

Le dernier point sur les avantages de cette méthode nous permet de faire un parallèle avec la première approche, car si ce schéma de compression n'est plus guidé par un principe d'inférence SRM, il en garde la philosophie. Ce schéma de compression est incontestablement plus approximatif, mais doit permettre de traiter des volumes de données plus importants. Le fait que les prototypes produits par cette méthode sont majoritairement dans les régions qui ont une forte densité d'exemples de la même classe, due à la minimisation de la distorsion, nous permet d'établir un parallèle entre le critère d'hypothèse $[\gamma_{\odot} \searrow]$ (*cf.* section 5.3.1.3) pour la sélection d'un exemple prototype et cette méthode de catégorisation. En effet, la probabilité qu'un prototype ait un nombre important d'exemples proches qui soient de sa classe augmente avec la densité d'exemples de sa classe dans son voisinage. C'est une approximation du critère d'hypothèse $[\gamma_{\odot} \searrow]$ qui est réalisée avec l'algorithme LBG, mais globalement cet algorithme doit produire des prototypes à la fois représentatifs et pertinents.

Les sections suivantes relatives à la présentation de cette seconde approche sont organisées de la manière suivante:
Dans une première partie, nous proposons une méthode de simplification d'une base d'apprentissage dont le niveau de simplification est contrôlé par le paramètre k de l'algorithme LBG. Nous réalisons une expérimentation de cette méthode. Les résultats de cette expérimentation illustrent qu'il existe une corrélation importante entre les capacités de généralisation avec l'ensemble réduit de prototypes produit par cette méthode et celles avec la base d'apprentissage sans simplification. Nous proposons également une technique simple qui permet de réaliser une sélection rapide des hyper-paramètres d'un SVM lorsque le nombre d'exemples est important. Nous réalisons ensuite une autre expérimentation dont l'objectif est d'illustrer que les hyper-paramètres efficaces pour un niveau de simplification donné ne sont pas systématiquement les plus efficaces avec un autre niveau de simplification. L'ensemble de ces résultats et remarques servent comme préliminaires à cette seconde approche.

Dans une seconde partie, nous revenons à notre objectif principal qui est de produire des fonctions de décision de complexités réduites avec des capacités en généralisation élevées. Nous définissons la nature de l'espace de recherche lié à la sélection d'un modèle idéal qui inclut la détermination des hyper-paramètres des SVM, du nombre de prototypes à produire et de la sélection des attributs. Nous définissons également un critère de qualité pour les fonctions de décision produites par notre métode de simplification à partir de SVM. Nous proposons une méthode méta-heuristique à base de recherche avec tabous pour l'optimisation de l'ensemble des paramètres du modèle. Une stratégie d'intensification et de diversification propre à cette optimisation est proposée. Plusieurs résultats expérimentaux sont proposés pour illustrer l'efficacité de la méthode de simplification proposée. Une discussion relative aux avantages, inconvénients et perspectives de cette seconde approche est réalisée[17].

17. La section 7.1.2 du chapitre 7 présente une application de cette méthode pour la production d'un classificateur de pixel performant et rapide à partir de l'exploitation d'un espace couleur hybride.

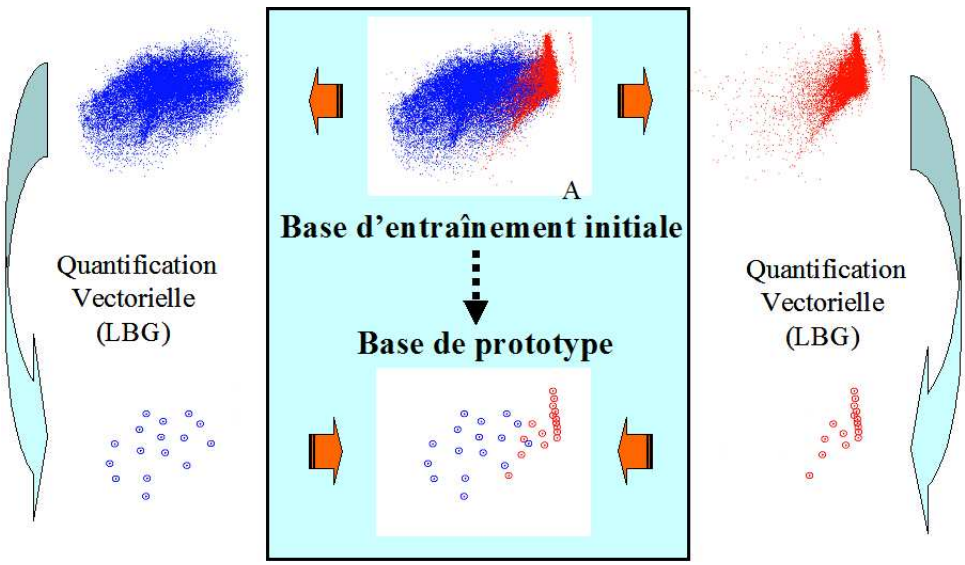


FIG. 5.7 – *Synopsis de notre méthode de simplification par quantification vectorielle à partir de l'algorithme LBG. Il y a 16 prototypes par classes ($k = 4$) pour cet exemple. Les données proviennent d'un problème de discrimination de pixels* (cf. *section 7.1.2): fond (rouge) ou cellule (bleu) où deux caractéristiques ont été sélectionnées (les composantes bleu et vert d'un espace rvb) pour permettre une illustration en 2d.*

5.3.2.1 Méthode de simplification

Le but de notre méthode de simplification est de produire, pour un niveau donné de simplification, un ensemble réduit de prototypes qui soit fortement représentatif de la répartition des données de chaque classe. Chaque prototype produit par notre méthode de simplification doit résumer le mieux possible un ensemble d'exemples d'une classe donnée de la base d'apprentissage. Pour produire ce résultat, l'algorithme LBG est appliqué, pour une valeur de k fixée représentant le niveau de simplification, à tous les exemples d'une classe donnée. Une valeur de k faible correspondra à un niveau de simplification important, et réciproquement. Cette procédure est repétée pour chaque classe de la base d'apprentissage. Comme l'objectif est de produire des fonctions de décision à partir des SVM, lorsque la base exploitée n'est pas binaire, le problème est transformé en une combinaison de problèmes binaires. Pour cette étude, nous utilisons le schéma de combinaison *un contre tous*. La figure 5.7 illustre le synopsis de notre méthode de simplification et l'algorithme 10 donne les détails de cette méthode. L'algorithme 9 donne les détails de l'algorithme LBG utilisé par notre méthode. Le partitionnement est réalisé en utilisant la règle PPV et la perturbation d'un centroïde pour produire 2 germes est proportionnelle à la variance des données rattachée au centroïde concerné.

Notre méthode de simplification produit le même nombre de prototypes par classe (sauf pour les niveaux de simplification les moins importants). Elle a donc pour effet d'équilibrer la base d'entraînement simplifiée produite, même si la base initiale ne l'était pas, en réalisant une réduction plus importante pour les classes majoritaires que pour les classes minoritaires. Pour pouvoir comparer les taux d'erreur à partir d'apprentissages réalisés sur des bases d'entraînement qui n'ont pas la même proportion d'exemples par classe

suivant le niveau de simplification employé, un taux d'erreur balancé (BER : *Balanced Error Rate*) est utilisé. Il est défini par :

$$e_{BER} = \frac{1}{|\mathcal{Y}|} \sum_{\omega \in \mathcal{Y}} e_\omega \tag{5.21}$$

avec e_ω le taux d'exemples mal classés de la classe ω et $\mathcal{Y}$ l'ensemble des classes. L'exemple de la base **Shuttle** est typique d'un problème mal balancé. Pour cette base, on remarque l'importance d'identifier correctement les exemples de classes fortement minoritaires, ainsi que l'importance d'utiliser le taux BER. En effet, un classificateur qui prédirait systématiquement la classe 1 à un taux d'erreur classique de 21,6 % sur cette base (car 78,4 % des exemples sont de la classe 1), alors que son taux d'erreur BER est de 83,3%. Le taux BER caractérise mieux la non-adéquation d'un classificateur lorsque la base n'est pas équilibrée en nombre d'exemples par classe.

Algorithme 9 LBG(X,k)

$\tilde{X}_0 \leftarrow \{\text{CENTROÏDE}(X)\}$
$P_0 \leftarrow \{X\}$
$i \leftarrow 0$ \\ Détermination des nouveaux germes à partir des centroïdes
pour $i \in \{1, \cdots, k\}$ **faire**
  $\tilde{X}'_i \leftarrow \emptyset$
  pour $j \in \{0, \cdots, |\tilde{X}_{i-1}|\}$ **faire**
    $\tilde{\mathbf{x}} \leftarrow \tilde{X}_{i-1}(j)$
    $\boldsymbol{\delta} \leftarrow \text{PERTURBATION}(\tilde{\mathbf{x}}, P_{i-1}(j))$
    $\tilde{X}'_i \leftarrow \tilde{X}'_i \cup \{\tilde{\mathbf{x}} - \boldsymbol{\delta}, \tilde{\mathbf{x}} + \boldsymbol{\delta}\}$
  fin pour
  $P_i \leftarrow \text{PARTITIONNEMENT}(\tilde{X}'_i)$
  $\tilde{X}_i \leftarrow \emptyset$ \\ Détermination des nouveaux centroïdes à partir de la partition
  pour $j \in \{0, \cdots, |\tilde{X}'_i|\}$ **faire**
    $\tilde{X}_i \leftarrow \tilde{X}_i \cup \{\text{CENTROÏDE}(P_i(j))\}$
  fin pour
fin pour
retourner $\tilde{X}_k$

Revenons à la figure 5.7, elle illustre parfaitement qu'un nombre réduit de prototypes (16 en l'occurrence) peut être suffisamment représentatif d'un nombre important d'exemples. Dans ce cas, si les hyper-paramètres du SVM entraîné sont correctement choisis, la fonction de décision produite devrait être également performante sur la base d'apprentissage initiale. La figure 5.8, illustre cette possibilité. Les figures 5.9 correspondent à un apprentissage avec simplification ($k = 7$) et sans simplification pour différentes valeurs des hyper-paramètres des SVM avec la base **Shuttle**. Ces résultats montrent que l'apprentissage avec des prototypes fournit des informations utiles sur les couples d'hyper-paramètres qui sont performants. L'apprentissage avec la version simplifiée de la base est plus sensible au choix des valeurs des hyper-paramètres. Nous avions déjà noté que plus le nombre d'exemples représentatifs d'un problème d'apprentissage est réduit et plus le

Algorithme 10 SIMPLIFICATION(Z,k)

$\tilde{Z} \leftarrow \emptyset$
pour $y \in \{-1, +1\}$ **faire**
  $X \leftarrow \{\mathbf{x}|(\mathbf{x},y) \in Z\}$
  si $2^k < |X|$ **alors**
    $\tilde{X} \leftarrow \text{LBG}(X,k)$
  sinon
    $\tilde{X} \leftarrow X$
  fin si
  $\tilde{Z} \leftarrow \tilde{Z} \cup \{(\tilde{\mathbf{x}},y)|\tilde{\mathbf{x}} \in \tilde{X}\}$
fin pour

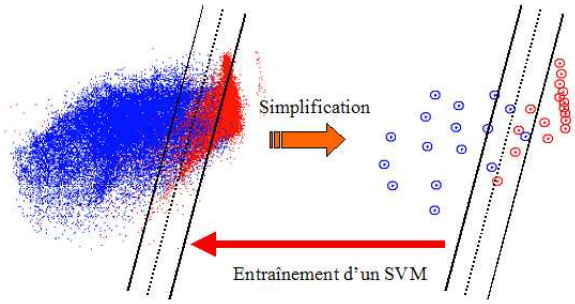
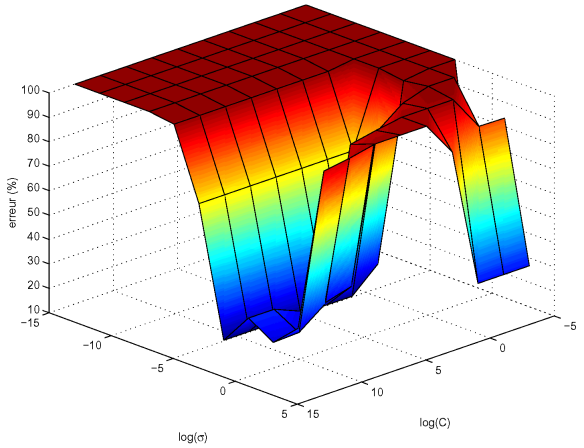
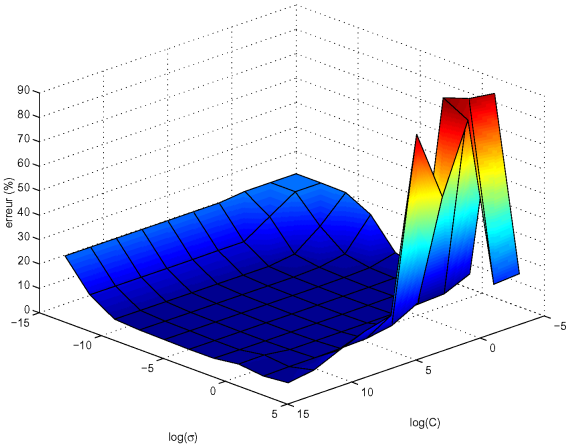


FIG. 5.8 – *Illustration du processus de construction d'une fonction de décision par un SVM entraîné sur une base d'apprentissage simplifiée ($k = 4$).*

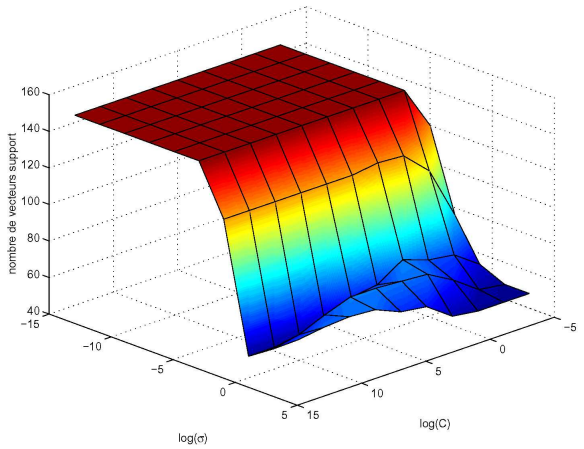
choix des hyper-paramètres doit être réalisé précisément (*cf.* section 5.3.1.6). La figure 5.9(a) illustre que dans ce cas c'est le choix de la largeur du noyau gaussien qui est le plus sensible à cette réduction. La raison est que lorsqu'il y a suffisamment d'exemples dans la base d'apprentissage, un réglage moins précis de σ est compensé par l'utilisation d'un plus grand nombre de vecteurs de support par la fonction de décision (*cf.* figure 5.9(d)). Les capacités en généralisation restent performantes (*cf.* figure 5.9(b)), même si elles sont légèrement amoindries. Ces résultats indiquent que la simplification de la base de données par notre méthode permet une identification plus marquée des hyper-paramètres qui permettent de produire des fonctions de décision qui ont des bonnes capacités de généralisation et un nombre réduit de vecteurs de support pour le niveau de simplification utilisé, mais également pour des niveaux de simplification moins importants. Ce résultat est très important si l'on cherche à optimiser ces deux critères. La comparaison des figures 5.9(c) et 5.9(d) illustre également qu'il est possible de produire une fonction de décision performante avec un nombre très faible de prototypes et donc de vecteurs de support. Un autre avantage est que les temps d'entraînement sont beaucoup plus courts avec la base simplifiée que dans le cas de la base non simplifiée, comme l'illustre les figures 5.9(e) et 5.9(f).



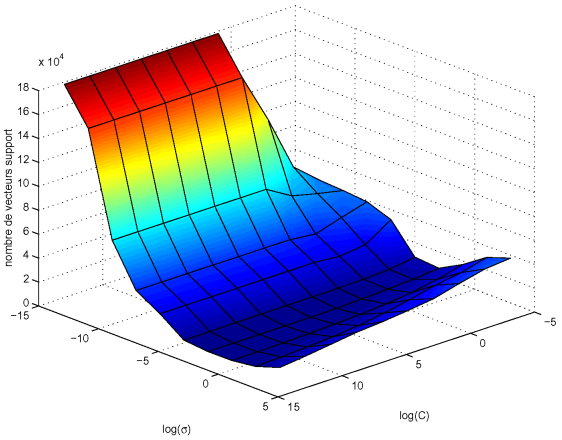
(a) Taux BER avec simplificateur



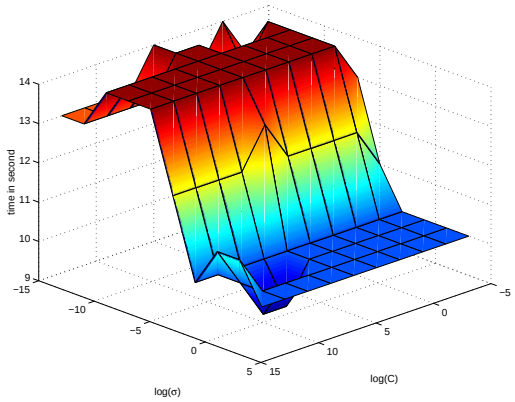
(b) Taux BER sans simplification



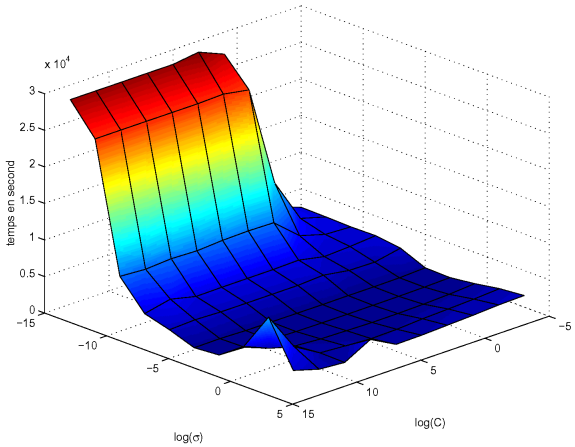
(c) Nombre de vecteurs de support avec simplification



(d) Nombre de vecteurs de support sans simplification



(e) Temps d'apprentissage avec simplificateur



(f) Temps d'apprentissage sans simplification

FIG. 5.9 – *Comparaison entre un apprentissage avec un ensemble réduit de prototypes et un apprentissage sans simplification.*

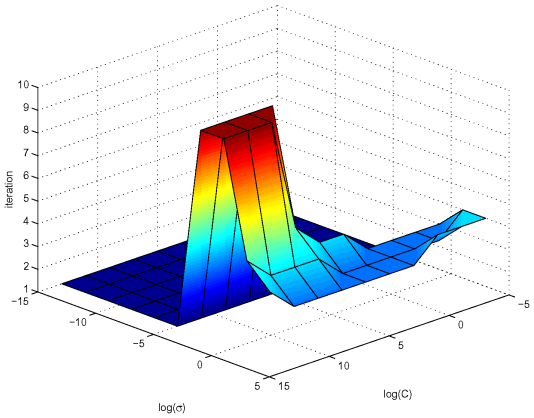


FIG. 5.10 – *Indique pour chaque couple (C,σ) l'itération à laquelle il est rejeté par l'algorithme 11.*

5.3.2.2 Sélection rapide d'hyper-paramètres par quantification vectorielle

A partir de ces remarques nous avons défini une méthode simple qui permet d'accélerer la sélection des hyper-paramètres d'un SVM qui est basée sur la procédure de *grid search* classique. Le principe est simple, il consiste à réaliser un entraînement avec l'ensemble des couples d'hyper-paramètres à tester à partir d'une version fortement simplifiée de la base d'entraînement et de supprimer l'ensemble des couples qui produisent des fonctions de décision qui ont de faibles performances en généralisation. Les performances en généralisation sont évaluées à partir du taux BER sur une base de test qui correspond à $1/3$ de la base d'apprentissage (le reste sert à la constitution de la base initiale d'entraînement). Ce processus est répété avec une base d'entraînement moins simplifiée, jusqu'à ne plus avoir que quelques couples à tester. Pour finir, le meilleur modèle est choisi à partir d'un entraînement à partir de la base non simplifiée. L'algorithme 11 correspond à la version détaillée de la méthode de sélection rapide. Pour cet algorithme les notations suivantes ont été utilisées:

- $k_{\min}$, $k_{\max}$: Les niveaux maximum et minimum de simplification,
- D : La fonction de décision d'un schéma *un-contre-tous* avec des SVM,
- R : La liste qui sert à l'enregistrement pour chaque couple θ d'hyper-paramètres ($\theta \equiv (C,\sigma)$) du taux d'erreur e_{BER} et du nombre de vecteurs de support n_{SV} de la fonction de décision correspondante,
- Θ: L'ensemble des couples d'hyper-paramètres à tester,
- φ: Critère de rejet d'un couple θ d'hyper-paramètres,
- MEILLEURSOLUTION: Retourne le couple d'hyper-paramètres, parmi ceux restant dans Θ, qui a le meilleur taux d'erreur balancé à partir des valeurs dans R.

La figure 5.10 est une illustration de l'application de cet algorithme avec la base **Shuttle**. Ce graphique représente à quelle itération de l'algorithme 11 certains couples sont supprimés de la liste R. Cette figure illustre que la suppression d'un grand nombre de couples est réalisée à partir des versions fortement simplifiées de la base d'entraînement. Comme le temps d'entraînement d'un SVM est en première approximation proportionnel au carré du nombre d'exemples, la durée nécessaire pour tester et rejeter un couple de para-

Algorithme 11 SelectionRapide(Z^A,Θ,$k_{\min}$,$k_{\max}$)

$(Z^E,Z^T) \leftarrow$ ECHANTILLONAGE(Z^A)\\ séparation proportionnelle $2/3$ et $1/3$
pour $k \in \{k_{\min},\cdots,k_{\max}\}$ **faire**
  $\tilde{Z}^E \leftarrow$ SIMPLIFICATION(Z^E,k)
  $R \leftarrow \emptyset$
  pour $\theta \in \Theta$ **faire**
    $D \leftarrow$ ENTRAÎNEMENTSVMS($\tilde{Z}^E$,θ)
    $e_{\mathrm{BER}} \leftarrow$ TAUXERREURBALANCÉ(D,Z^T)
    $n_{\mathrm{SV}} \leftarrow$ NOMBREDEVECTEURSSUPPORT(D)
    $R \leftarrow R \cup \{(\theta,e_{\mathrm{BER}},n_{\mathrm{SV}})\}$
  fin pour
  $e^*_{\mathrm{BER}} \leftarrow$ MINIMUMERREUR(R)
  $n^*_{\mathrm{SV}} \leftarrow$ MINIMUMSV(R)
  pour $(\theta,e_{\mathrm{BER}},n_{\mathrm{SV}}) \in R$ **faire**
    si $\varphi(e_{\mathrm{BER}}) > e^*_{\mathrm{BER}} \vee \varphi(n_{\mathrm{SV}}) > n^*_{\mathrm{SV}}$ **alors**
      $\Theta \leftarrow \Theta \backslash \{\theta\}$
    fin si
  fin pour
fin pour
retourner MEILLEURSOLUTION(Θ)

mètres non appropriés est fortement réduite. On remarque également que les plus mauvais couples de paramètres sont supprimés dès les premières itérations. La figure 5.9(f) illustre que les temps d'entraînement sont encore plus importants avec ces couples lorsqu'aucune simplification n'est réalisée. Pour ces couples d'hyper-paramètres, l'effet de réduction des temps d'entraînement est accru avec notre méthode de sélection rapide.

Le tableau 5.12 donne les durées d'entraînement[18] nécessaire à la sélection des hyper-paramètres avec une procédure de *grid search* classique et avec notre méthode de sélection rapide. Les valeurs des couples des hyper-paramètres testées vérifient les relations suivantes : $\log_2(C) \in \{-5,\cdots,15\}$ et $\log_2(\sigma) \in \{-15,\cdots,5\}$. $k_{\min} = 2$ et $k_{\max} = \arg\max(\lceil \log_2(m_\omega) \rceil)$ pour les 3 bases testées. Le critère de rejet choisi est : $\varphi(x) = x/2$. Les résultats du tableau 5.12 montrent une réduction significative de la durée de la procédure de sélection des hyper-paramètres des SVM grâce à notre méthode. Les capacités de généralisation des modèles sélectionnés, évaluées à partir de la base de validation, ont des performances similaires (très légèrement dégradées pour la base satimage). La difficulté avec cette méthode est le choix de φ. Trop sélectif, la procédure de sélection sera rapide, mais risque de dégrader les performances en généralisation. Pas assez sélectif, la durée de sélection peut devenir supérieure à la procédure classique. Au-delà d'une méthode simple et rapide de sélection de modèle, ces résultats illustrent à nouveau que l'entraînement d'un

18. La librairie SVMTorch [COLLOB01] a été utilisée avec un PIII 1.2Ghz avec 512MO. SVMTorch n'est plus actuellement la librairie SVM la plus rapide, les temps d'entraînement seraient maintenant plus courts avec la librairie libsvm [CHANG01] avec un même ordinateur, mais le ratio entre les deux durées reste globalement le même.

	notre méthode		***grid search* classique**	
	durée	**taux BER**	**durée**	**taux BER**
satimage	8,5	10,7%	29,9	10,1%
letter	19	6,34%	827	6,34%
shuttle	3,8	0,09%	412	0,09%

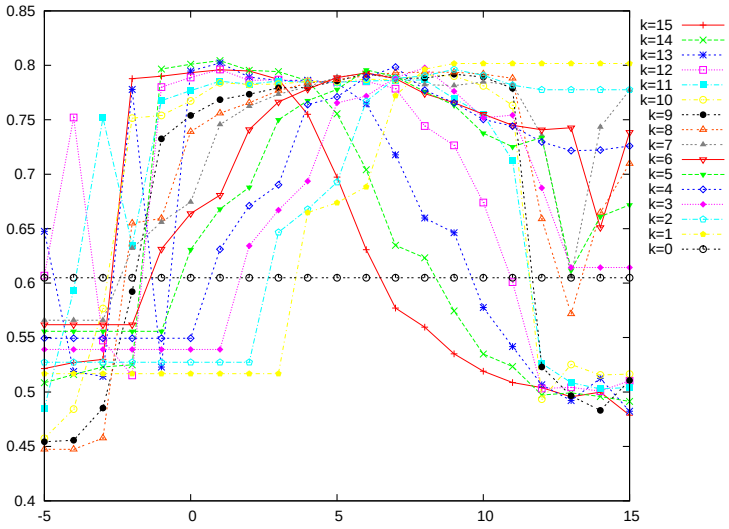
TAB. 5.12 – *taux d'erreur BER et temps d'entraînement (durée)en heures.*

SVM avec une version correctement simplifiée de la base de données initiale donne des informations pertinentes sur le comportement d'un entraînement avec une version non simplifiée de cette base.

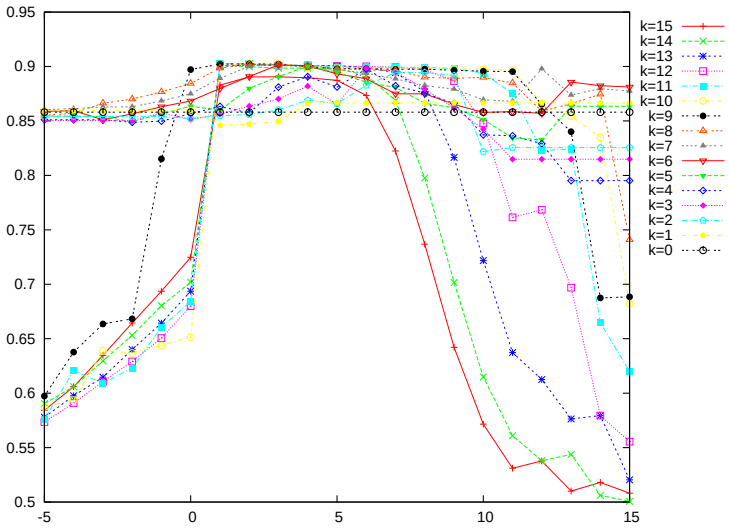
5.3.2.3 Liens entre hyper-paramètres et simplification

Les résultats précédents ont montré que les capacités en généralisation d'un SVM avec une version simplifiée de la base d'entraînement d'un SVM sont corrélées avec ceux d'un apprentissage avec une version moins simplifiée. S'il y a corrélation, elle n'est pas totale. Pour illustrer cette remarque, concentrons-nous sur l'importance de l'hyper-paramètre de régularisation C d'un SVM. Si les données sont faiblement bruitées, la constante C doit avoir une valeur élevée, car il n'y a aucune raison d'autoriser un grand nombre d'exemples à avoir leurs variables ressorts différentes de zéro. Si les données sont fortement bruitées, il est important de choisir une valeur de C relativement faible pour ne pas être trop sensible au bruit lié à ces données. Les graphiques de la figure 5.11 illustrent l'importance du choix de la constante C suivant le niveau de simplification. Le problème de classification de pixels est initialement fortement bruité (la figure 5.8 illustre ce fait, même si seulement deux attributs de l'espace RVB sont représentés). Il est habituel d'avoir un nombre important d'erreurs sur la base d'entraînement initiale et la régularisation à travers le paramètre C ne doit pas être trop importante, sinon les capacités en généralisation sont dégradées. La figure 5.11 illustre cet effet lorsque les valeurs de k sont élevées (*i.e.* avec une faible simplification). Plus le niveau de simplification est important et plus chaque prototype est individuellement représentatif de la classe qu'il représente. Globalement cela correspond à avoir un ensemble de données plus faiblement bruitées (les prototypes de la figure 5.8 illustrent également ce fait). De plus, mal classer un de ces prototypes correspond également à mal classer la majorité des exemples qu'il représente. L'adéquation avec la base initiale est par conséquent fortement réduite. Le paramètre C doit avoir une valeur plus importante pour réduire le nombre de prototypes pouvant être mal classés. A nouveau, la figure 5.11 illustre cet effet avec les plus faibles valeurs de k. L'ensemble de ces remarques montrent les limites de la méthode proposée en section 5.3.2.2 lorsque les données sont fortement bruitées et que le niveau de simplification initial est important (les valeurs de $k_{\min}$, $k_{\max}$ et la fonction φ doivent être judicieusement choisies pour l'algorithme 11).

Les résultats de la figure 5.12 montrent également que si les hyper-paramètres sont correctement choisis pour un niveau de simplification donné, les capacités de généralisation peuvent être très proches pour différents niveaux de simplification. L'entraînement avec une version fortement simplifiée de la base initiale peut donc produire des fonctions de décision performantes. Ces fonctions de décision auront en plus l'avantage d'être de complexité réduite et elles nécessiteront un temps d'entraînement beaucoup plus faible

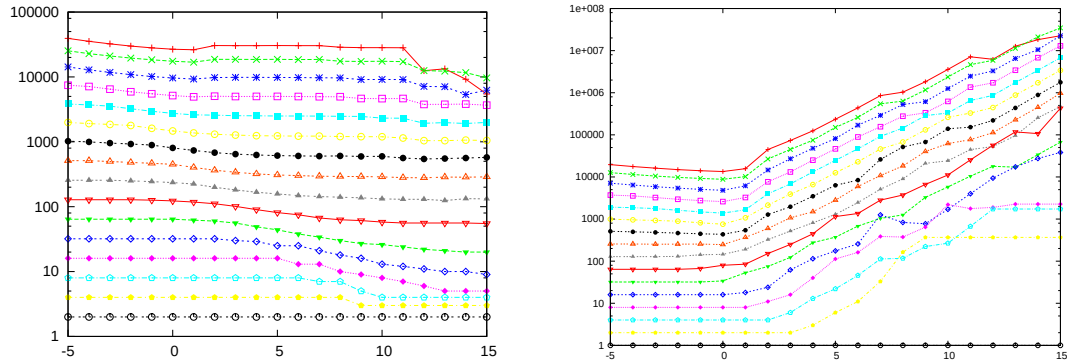


(a) classe $\{2\}$ contre les classes $\{1,3\}$



(b) classe $\{3\}$ contre les classes $\{1,2\}$

FIG. 5.11 – *Taux de reconnaissance ($1 - e_{\mathrm{BER}}$ en ordonnée) en fonction de la constante de régularisation C (l'abscisse des deux graphiques correspond à $log_{\sqrt{2}}(C)$) et du niveau de simplification k pour deux des trois fonctions de décision du problème* **Classpixel** *(*cf. *section 7.1.2) avec un noyau gaussien ($\sigma = 1/4$).*



(a) (b)

FIG. 5.12 – *Nombre de vecteurs de support (a) et d'itérations de SMO (b) en fonction de la constante de régularisation C (l'abscisse des deux graphiques correspond à $log_{\sqrt{2}}(C)$) et du niveau de simplification k pour la discrimination de la classe 2 contre les classes 1 et 3 du problème* **Classpixel** *(*cf. *section 7.1.2) avec un noyau gaussien ($\sigma = 1/4$).*

comme l'illustrent les graphiques de la figure 5.12. Le problème est de choisir automatiquement le bon niveau de simplification suivant le problème traité, ainsi que les bonnes valeurs pour les hyper-paramètres. Par exemple pour la figure 5.11(a) c'est le choix $k = 1$ avec $C > 32$ ($log_{\sqrt{2}}(C) > 10$) qui est approprié et pour la figure 5.11(b) c'est le choix $k = 3$ avec $C \approx 8$ ($log_{\sqrt{2}}(C) \approx 6$) qui est approprié.

Pour observer l'influence du niveau de simplification sur les capacités de généralisation, nous avons utilisé notre méthode de simplification avec deux autres bases de référence. Pour chaque niveau k de simplification utilisé, une procédure de sélection de modèle de type *grid search* est réalisée à partir de l'utilisation d'un taux d'erreur BER estimé sur la base de test. Le taux de reconnaissance ($1 - e_{\mathrm{BER}}$) est ensuite estimé à partir de la base de validation. La figure 5.13(a) montre que la base **Adults** produit de bons résultats avec un niveau de simplification important, pour la base **Web** le gain en généralisation reste significatif jusqu'à $k = 12$. Une baisse des performances pour les versions les moins simplifiées ($k = 14$ correspond pour ces deux bases à une exploitation directe de la base initiale) est remarquée pour ces deux bases (effet plus marquée pour la base **Web**). Ces résultats illustrent à nouveau qu'un ensemble de prototypes bien choisis permet d'améliorer les capacités de généralisation (*i.e.* réduction du bruit entachant les données d'apprentissage). L'augmentation importante de la complexité des fonctions de décision produites (*cf.* figure 5.13(b)) et des durées d'entraînement (*cf.* figure 5.13(c)) est à nouveau observée.

L'ensemble des résultats (de cette section et de la précédente) illustrent que le choix d'un compromis efficace entre performances en généralisation et complexité est difficile à réaliser. De plus, la sélection des hyper-paramètres des SVM peut être plus ou moins dépendante du niveau de simplification, suivant la nature des données utilisées.

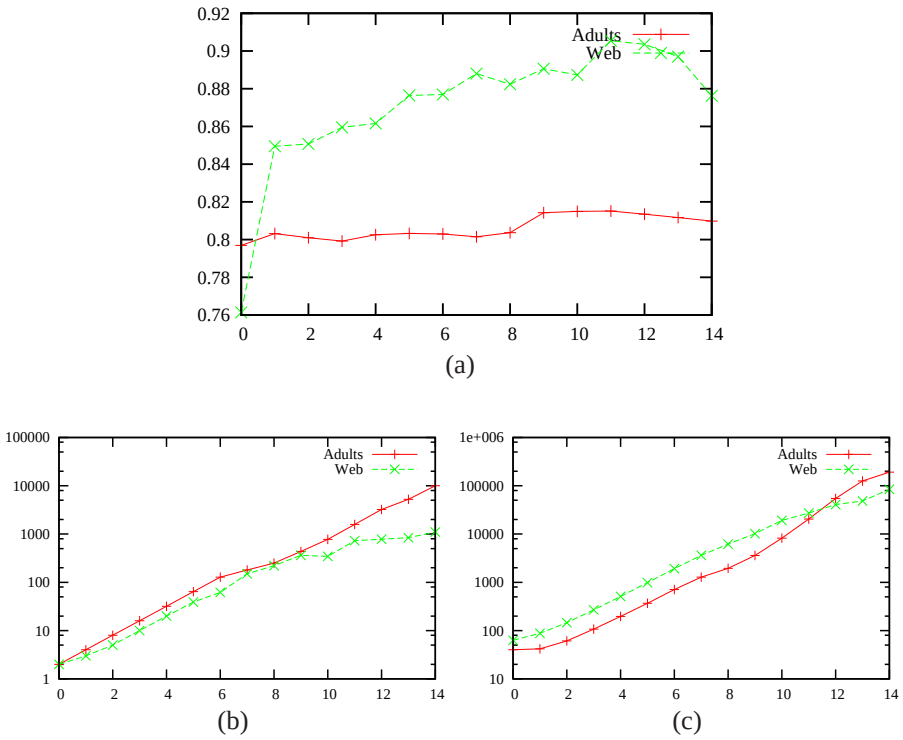


FIG. 5.13 – *Taux de reconnaissance sur la base de validation (a) et nombre de vecteurs de support de la fonction de décision (b) pour le modèle sélectionné par une procédure de* grid search *pour chaque niveau de simplification* k*. Le graphique (c) donne la durée totale en secondes de la procédure de* grid search *pour chaque niveau* k *de simplification donné pour ces deux bases (*$log_{\sqrt{2}}(C) \in \{-10, \cdots, 20\}$ *et* $log_{\sqrt{2}}(\frac{1}{2\sigma^2}) \in \{-10, \cdots, 10\}$*).* **adult** *et* **web** *sont les deux bases utilisées.*

5.3.2.4 Espace de recherche et critère de qualité

A partir des remarques et résultats précédents, la sélection d'un modèle efficace[19] $\boldsymbol{\theta}$ pour produire une fonction de décision performante et de complexité réduite nécessite la sélection de la simplification à appliquer à la base initiale d'entraînement et des valeurs des hyper-paramètres associés à l'entraînement d'un SVM. La simplification appliquée dépend exclusivement de la valeur de k pour la méthode de simplification utilisée. Cette valeur pour un problème d'apprentissage binaire donné est comprise entre 0 et $\log_{\sqrt{2}}(\max(m_{+1},m_{-1}))$ avec m_{+1} et m_{-1} le nombre d'exemples pour chacune des deux classes. Pour les hyper-paramètres, si on se limite à un noyau gaussien, ce sont les valeurs de C et de σ qui doivent être déterminées. Pour garder une approche similaire à celle de la recherche avec grille, les valeurs possibles de C et σ sont quantifiées ($log_{\sqrt{2}}(C) \in \{-10,\cdots,20\}$ et $log_{\sqrt{2}}(\frac{1}{2\sigma^2}) \in \{-10,\cdots,10\}$ pour les expérimentations réalisées). Dans le cas où la sélection des attibuts doit également être réalisée, nous ajoutons au modèle la détermination d'un vecteur $\boldsymbol{\beta}$ d'élements binaires. Un attribut i est sélectionné si $\boldsymbol{\beta}_i = 1$ (dans le cas contraire $\boldsymbol{\beta}_i = 0$). Si aucune sélection d'attributs ne doit être réalisée, le vecteur $\boldsymbol{\beta}$ a tous ses éléments fixés à 1.

Un modèle $\boldsymbol{\theta}$ est donc un vecteur de $n_{\boldsymbol{\theta}}$ variables entières, notées θ_i, qui ont chacune un domaine de définition qui correspond à un intervalle. Pour un modèle $\boldsymbol{\theta}$ donné, nous notons $C_{\boldsymbol{\theta}}$, $\sigma_{\boldsymbol{\theta}}$, $k_{\boldsymbol{\theta}}$ et $\boldsymbol{\beta}_{\boldsymbol{\theta}}$ les valeurs des paramètres qui sont liées à $\boldsymbol{\theta}$. Pour prendre en compte la sélection d'attributs réalisé dans l'apprentissage d'un SVM, nous utilisons la fonction noyau $K_{\boldsymbol{\beta}}$ suivante:

$$K_{\boldsymbol{\beta}}(\mathbf{u},\mathbf{v}) = \exp\left(\frac{-\sum_{i=1}^{n}\beta_i\left(\mathbf{u}_i-\mathbf{v}_i\right)^2}{2\sigma^2}\right) \tag{5.22}$$

L'algorithme 12 décrit les étapes d'entraînement d'un SVM pour un modèle $\boldsymbol{\theta}$, ainsi que les données utilisées pour déterminer la qualité q d'une fonction de décision D produite par un SVM. Pour un problème binaire donné la phase ECHANTILLONAGE n'est réalisée que pour le premier modèle $\boldsymbol{\theta}$ exploité. Pour chaque problème d'apprentissage binaire, la simplification de la base d'entraînement Z^E pour un niveau k donné est enregistrée (ainsi que toutes celles de niveaux inférieurs à k), ceci afin d'éviter d'avoir à la recalculer. Toute nouvelle simplification à réaliser, dont le résultat pour un niveau k spécifié n'a pas été préalablement enregistré, est produite à partir de la première simplification d'un niveau k' ($k' < k$) déjà enregistré. Globalement cette procédure de sauvegarde permet que la réalisation d'un nombre important d'entraînements pour différents modèles soit plus rapide, en évitant de quantifier de multiples fois et de la même façon les mêmes données.

Reste à définir la Qualité d'une Fonction de Décision (QFD) à partir d'un compromis entre le taux d'erreur BER et la complexité de la fonction de décision produite. Cette complexité dépend du nombre de vecteurs de support nécessaires à sa définition et du nombre d'attributs utilisés (lorsqu'une sélection d'attributs est réalisée) par cette fonction. Notre définition empirique de ce critère de qualité est la suivante:

$$q_{\mathrm{FD}} = (1 - e_{\mathrm{BER}}) - c_{\mathrm{VS}}\log_2(1 + n_{\mathrm{VS}}) - c_\beta \log_2(\text{coût}(\boldsymbol{\beta})) \tag{5.23}$$

19. Nous utilisons les mêmes notations que celles introduites dans la section 3.1.

Algorithme 12 SVM-QFD(Z^A, $\boldsymbol{\theta}$)

$(C_{\boldsymbol{\theta}},\sigma_{\boldsymbol{\theta}},k_{\boldsymbol{\theta}},\boldsymbol{\beta}_{\boldsymbol{\theta}}) \leftarrow$ EXTRAIREPARAMÈTRES($\boldsymbol{\theta}$)
$(Z^E,Z^T) \leftarrow$ ECHANTILLONAGE(Z^A)\\ séparation proportionnelle $2/3$ et $1/3$
$\tilde{Z}^E \leftarrow$ SIMPLIFICATION($Z^E,k_{\boldsymbol{\theta}}$)
$D \leftarrow$ ENTRAÎNEMENTSVM($\tilde{Z}^E,C_{\boldsymbol{\theta}},\sigma_{\boldsymbol{\theta}},\boldsymbol{\beta}_{\boldsymbol{\theta}}$)
$e_{\mathrm{BER}} \leftarrow$ TAUXERREURBALANCÉ(D,Z^T)
$n_{\mathrm{VS}} \leftarrow$ NOMBREDEVECTEURSSUPPORT(D)
$q \leftarrow$ QFD($e_{\mathrm{BER}},n_{\mathrm{SV}},\boldsymbol{\beta}_{\boldsymbol{\theta}}$)
retourner q

avec coût($\boldsymbol{\beta}$) $= \sum \boldsymbol{\beta}_i\boldsymbol{\kappa}_i$. Les constantes $\boldsymbol{\kappa}_i$ représentent le coût relatif d'extraction des attributs i. Lorsque ces coûts ne sont pas connus (ce qui est généralement le cas avec des bases de *benchmark*), $\boldsymbol{\kappa}_i = 1$ pour tous les attributs. Les valeurs des constantes c_{VS} et c_β servent à la pénalisation de la complexité et sont fixées par l'utilisateur. Ils représentent respectivement l'importance de la pénalisation due à l'augmentation du nombre de vecteurs de support et à l'augmentation du nombre d'attributs utilisés ou des coûts associés. L'idée directrice, pour cette définition, est qu'un doublement du nombre de vecteurs de support doit au moins correspondre à une diminution de la valeur de c_{VS} pour le taux d'erreur BER pour que le processus décisionnel correspondant soit considéré comme étant de qualité supérieure. A partir de la constante c_β l'utilisateur fixe le même type de compromis entre l'augmentation du coût d'extraction des attributs et la diminution du taux d'erreur. Le choix des valeurs de c_{VS} et c_β est laissé à l'utilisateur suivant l'importance qu'il accorde à la réduction du temps de décision par rapport à la réduction du taux d'erreur. Lorsque les valeurs de c_{VS} et c_β sont faibles (< 0,01), c'est le principe de parcimonie qui est privilégié. Dans le cas où c_{VS} et c_β sont nulles, la solution de meilleure qualité n'est pas forcément celle qui correspond à une non simplification de la base initiale comme le montre la figure 5.13(a)[20]: la sélection automatique du niveau optimal de simplification (ou de l'ensemble optimal d'attributs) est alors nécessaire.

5.3.2.5 Recherche avec tabous pour la sélection de modèles

Considérations générales

L'espace de recherche Θ a une cardinalité importante lorsque la sélection d'attributs doit être réalisée:

$$|\Theta| = \prod_{i=1}^{n_{\boldsymbol{\theta}}} \left(\theta_i^{\max} - \theta_i^{\min} + 1\right) \tag{5.24}$$

Lorsque la sélection d'attributs n'est pas à réaliser, la taille de l'espace de recherche est moins importante, mais l'exploration de toutes les solutions envisageables reste problématique lorsque les bases d'entraînement ont plusieurs dizaines de milliers d'exemples. Il est alors important de définir une méthode de sélection qui puisse explorer l'espace Θ de manière plus efficace et dans des temps raisonnables qu'une évaluation systématique de tous les modèles de Θ et ceci même si la sélection d'attributs n'est pas à réaliser.

20. La même remarque peut être faite avec la sélection d'attributs. Voir par exemple [CHEN05A].

Les remarques précédentes nous ont orientés vers une approche à base de méta-heuristique pour la sélection d'un modèle optimisant au mieux le critère de qualité (5.23). La recherche avec tabous est une des méthodes méta-heuristiques qui nous a semblé la plus pertinente pour réaliser cette sélection de modèle. Elle a d'ailleurs déjà été utilisée pour la sélection d'attributs [KORYCI04] et pour la sélection des hyper-paramètres des SVM [CAWLEY01] avec de bons résultats dans les deux cas. L'originalité est ici, d'utiliser la recherche avec tabous pour optimiser l'ensemble de ces paramètres, ainsi que la sélection du niveau de simplification. L'objectif est de permettre une sélection rapide de bons modèles à partir d'une représentation simplifiée et de n'utiliser les représentations moins simplifiées que si elles diminuent suffisamment le taux d'erreur. Pour obtenir un tel résultat, la recherche avec tabous doit commencer avec une représentation fortement simplifiée de la base d'entraînement (une valeur de k faible).

Un autre point important, précédemment signalé en section 5.3.2.3, est que lorsque le niveau de simplification est changé, les valeurs des hyper-paramètres doivent être adaptées. Comme la recherche avec tabous exploite les solutions $\boldsymbol{\theta}$ dans un voisinage restreint, elle peut réaliser ces adaptations en faisant suivre, si nécessaire, un changement (mouvement) du niveau de simplification par des changements adéquats relatifs aux hyper-paramètres.

Mouvements relatifs au changement de modèle

A partir de la définition d'une solution (modèle) $\boldsymbol{\theta}$ un mouvement de la recherche avec tabous consiste à ajouter ou supprimer une quantité[21] de $\pm\delta$ à une des variables θ_i (*cf.* section 3.1.1.2). Ces mouvements correspondent suivant la variable modifiée à augmenter ou diminuer les valeurs des hyper-paramètres ($C \leftarrow \sqrt{2}^{\left(log_{\sqrt{2}}(C)\pm\delta\right)}$ ou $\sigma \leftarrow \sqrt{2}^{\left(log_{\sqrt{2}}(\sigma)\pm\delta\right)}$) ou le niveau de simplification ($k \leftarrow k \pm \delta$). Lorsque la sélection d'attributs doit être réalisée, un mouvement correspond à ajouter ou supprimer un attribut ($\boldsymbol{\beta}_i \leftarrow \boldsymbol{\beta}_i + 1 = 1$ ou $\boldsymbol{\beta}_i \leftarrow \boldsymbol{\beta}_i - 1 = 0$).

Solutions taboues

Pour éviter la formation de cycles et l'existence d'oscillations rapides entre des états fortement similaires (*i.e.* modifier régulièrement un ensemble restreint de variables en utilisant systématiquement un ensemble réduit de valeurs dans leur domaine de définition) la notion de solutions taboues a été définie de la façon suivante:

$$\bar{\Theta}^{it}_{tabu} = \{\theta \in \Omega \mid \exists\, i,t' : t' \in [1,\ldots,t],\, \theta_i \neq \theta_i^{it-1} \wedge \theta_i = \theta_i^{it-t'}\} \tag{5.25}$$

avec t un paramètre ajustable correspondant à la notion de mémoire à court terme. Nous avons choisi que la durée de la mémoire à court terme dépende de la dimension de l'espace de recherche de la façon suivante:

$$t = \sum\nolimits_{i=1}^{n_{\boldsymbol{\theta}}} \theta_i^{\max} - \theta_i^{\min} \tag{5.26}$$

Cette définition de la notion de solutions taboues correspond à refuser de modifier une variable, si la nouvelle valeur qu'elle doit prendre correspond à une valeur qu'elle avait

21. $\delta = 1$ pour les mouvements de base.

eu précédemment dans les t dernières itérations. Cette définition correspond bien aux objectifs de réduire fortement le risque de formation de cycles, ainsi que les phénomènes d'oscillations tels que décrits précédemment. Pour pouvoir vérifier facilement qu'une solution est taboue, il est suffisant de mémoriser pour chaque valeur du domaine de définition des variables, l'itération à laquelle un mouvement l'a utilisée [22]. Lorsqu'un mouvement est réalisé, pour tester si la solution produite n'est pas taboue, il suffit alors de vérifier pour la variable modifiée que l'écart entre l'itération actuelle et la dernière itération où elle a eu cette valeur est supérieure à t. Cette définition des solutions taboues peut par contre interdire de sélectionner une solution meilleure que celles déjà retenues. Pour réduire cet effet négatif, nous utilisons le critère classique d'aspiration (*cf.* section 3.1.1.4).

Stratégies d'intensification et de diversification

Nous avons également défini une stratégie d'intensification et une stratégie de diversification pour améliorer les performances de notre recherche de modèles avec tabous. La première raison est le fait de devoir tester un grand nombre de solutions à chaque itération de la recherche $\boldsymbol{\theta}$ avec tabous. Cet effet est encore plus marqué lorsqu'une sélection d'attributs doit être réalisée et que le nombre d'attributs utilisés est important. Dans le cas où le modèle actuel a une qualité faible par rapport aux meilleures solutions déjà rencontrées, l'ajout ou la suppression d'un attribut a un effet marginal sur la qualité de la solution produite. Il est alors préférable de se concentrer sur l'importance de modifier les valeurs de C, σ ou k. La seconde raison évoquée dans la section 3.1.2 correspond au fait qu'il est préférable de modifier fortement une solution pour s'éloigner rapidement des régions de l'espace de recherche qui ne semblent plus permettre d'améliorer la qualité d'une solution. La stratégie d'intensification teste seulement les mouvements qui agissent sur les valeurs de C, σ et k avec $\delta = 1$, si la solution actuelle $\boldsymbol{\theta}^{it}$ a une QFD qui n'est pas suffisamment proche de celle de la meilleure solution $\boldsymbol{\theta}^{*}$ découverte lors des précédentes itérations (*cf.* procédure MOUVEMENTSPOSSIBLERESTREINT dans l'algorithme 13). Si elle est suffisament proche de $\boldsymbol{\theta}^{*}$, elle teste également les mouvements qui agissent sur la sélection des attributs (*cf.* procédure MOUVEMENTSPOSSIBLECOMPLET dans l'algorithme 13). Si la stratégie d'intensification ne parvient pas à augmenter la valeur $\boldsymbol{\theta}^{*}$ au bout de plusieurs itérations de la recherche avec tabous, il est considéré que cette stratégie a échoué et il y a basculement vers la stratégie de diversification.

L'algorithme 13 donne les détails de la stratégie d'intensification. Les notations utilisés dans cet algorithme sont les suivantes:

- η est un coefficient compris entre 0 et 1 qui permet de définir le seuil à partir duquel une solution est considérée comme suffisament prometteuse (par rapport au meilleur modèle découvert) pour que les mouvements concernant la sélection d'attributs soient autorisés.
- Θ_{suivant} correspond à l'ensemble des solutions (taboues ou non) qui sont dans le voisinage direct de la solution actuelle $\boldsymbol{\theta}^{it}$ en tenant compte de la politique des mouvements autorisés (*i.e.* complet ou restreint).
- La procédure MEILLEURESOLUTIONNONTABOUES retourne la meilleure solution qui ne soit pas taboue (sauf effet d'aspiration) à partir de l'estimation de la QFD de

22. Le compteur pour chaque valeur possible de chaque variable est initialisé à $-\infty$.

Algorithme 13 INTENSIFICATION($\boldsymbol{\theta}^{it}$)

si $q_{\mathrm{FD}}(\boldsymbol{\theta}^{it}) > \eta \cdot q_{\mathrm{FD}}(\boldsymbol{\theta}^{*})$ **alors**
  $\Theta_{\text{suivant}} \leftarrow$ MOUVEMENTSPOSSIBLECOMPLET($\boldsymbol{\theta}^{it}$)
sinon
  $\Theta_{\text{suivant}} \leftarrow$ MOUVEMENTSPOSSIBLERESTREINT($\boldsymbol{\theta}^{it}$)
fin si
$\boldsymbol{\theta}^{it+1} \leftarrow$ MEILLEURESOLUTIONNONTABOUES(Θ_{suivant})
si $q_{\mathrm{FD}}(\boldsymbol{\theta}^{it+1}) > q_{\mathrm{FD}}(\boldsymbol{\theta}^{\text{intensification}})$ **alors**
  $\boldsymbol{\theta}^{\text{intensification}} \leftarrow \boldsymbol{\theta}^{it+1}$
  $n_{\text{SansAmélioration}} \leftarrow 0$
  si $q_{\mathrm{FD}}(\boldsymbol{\theta}^{it+1}) > q_{\mathrm{FD}}(\boldsymbol{\theta}^{*})$ **alors**
    $\boldsymbol{\theta}^{*} \leftarrow \boldsymbol{\theta}^{it+1}$
    $n_{\text{Echec}} \leftarrow 0$
  fin si
sinon
  $n_{\text{SansAmélioration}} \leftarrow n_{\text{SansAmélioration}} + 1$
  si $n_{\text{SansAmélioration}} > n^{\max}_{\text{SansAmélioration}}$ **alors**
    $n_{\text{Echec}} \leftarrow n_{\text{Echec}} + 1$
    si $n_{\text{Echec}} > n^{\max}_{\text{Echec}}$ **alors**
      `ContinuerRercherche` $\leftarrow$ non
    sinon
      `Strategie` $\leftarrow$ DIVERSIFICATION
    fin si
  fin si
fin si

l'ensemble des solutions de Θ_{suivant}. $\boldsymbol{\theta}^{\text{intensification}}$ correspond à la meilleure solution durant les itérations d'une stratégie d'intensification ($\boldsymbol{\theta}^{\text{intensification}}$ est initialisée avec la valeur de la solution actuelle au début d'une phase d'intensification).

- $n_{\text{SansAmélioration}}$ compte le nombre sucessif d'itérations avec une stratégie d'intensification qui n'ont pas permis d'améliorer la solution $\boldsymbol{\theta}^{\text{intensification}}$ et $n^{\max}_{\text{SansAmélioration}}$ correspond au nombre maximum de ces itérations sans amélioration qui provoque automatiquement le basculement vers une stratégie de diversification.
- n_{Echec} compte le nombre de recours à la stratégie d'intensification qui n'ont pas permis d'améliorer la QFD de la meilleure solution visitée $\boldsymbol{\theta}^*$. Il est important de noter que la découverte d'une solution meilleure que l'actuelle $\boldsymbol{\theta}^*$ ré-initialise le compteur d'échec à zéro. Si le nombre d'échec est trop important (le nombre maximum est fixé par $n^{\max}_{\text{Echec}}$), alors la recherche avec tabous s'arrête[23] et retourne la valeur de $\boldsymbol{\theta}^*$.

La stratégie de diversification a pour but de changer de façon significative la nature du modèle $\boldsymbol{\theta}$ actuel et d'atteindre par là même de nouvelles regions de l'espace de recherche, afin de découvrir des modèles de meilleurs qualités. Ce but est atteint en utilisant des mouvements qui ont une plus grande amplitude ($\delta > 1$) et en restreignant chaque itération de la recherche avec tabous à des mouvements qui ne peuvent agir que sur une seule variable. Il y a donc au maximum deux mouvements envisagés: $\boldsymbol{\theta}_i + \delta$ et $\boldsymbol{\theta}_i - \delta$ (*cf.* procedure DEUXMOUVEMENTPOSSIBLE dans l'algoritme 14). De plus, les variables qui sont liées à la sélection d'attributs ne sont pas modifiées par la stratégie de diversification. Le choix de la variable à modifier est réalisé par un tirage aléatoire uniforme parmi clles agissant sur C, σ et k relativement à l'espace des modèles utilisés. L'algorithme 14 donne les détails de la procédure de diversification. L'algorithme 14 partage un certain nombre de notations communes avec l'algorithme 13. Lorsque la stratégie adoptée devient la diversification, la variable $n_{\text{Diversification}}$ est initialisée à zéro. Pour obliger à explorer des régions de l'espace de recherche de plus en plus éloignées en fonction du nombre consécutif d'échecs, le nombre d'itérations maximum de la phase de diversification est égal à: $n^{\max}_{\text{Diversification}} = n_{\text{Echec}} \cdot n^{\max}_{\text{Intensification}}$. Si une solution meilleure que toute celles explorées est découverte, il y a basculement automatique vers la stratégie d'intensification (effet d'aspiration), ceci afin d'explorer au mieux les régions de l'espace de recherche aux alentours de cette solution. Dans le cas contraire, la dernière itération de la phase de diversification fixe comme modèle initial de prochaine phase d'intensification le meilleur modèle découvert lors de la phase de diversification (indépendamment des solutions déjà explorées par les précédentes phases de ces deux stratégies).

Initialisation de la méthode

La méthode de recherche avec tabous pour la sélection d'un modèle optimal utilise comme solution initiale[24] $\boldsymbol{\theta}$ celle qui correspond à $C_{\boldsymbol{\theta}} = 1$, $\sigma_{\boldsymbol{\theta}} = 1$ et

$$k_{\boldsymbol{\theta}} = \lfloor \log_2 (\max (m_{+1}, m_{-1})) / 3 \rfloor \tag{5.27}$$

avec m_{+1} et m_{-1} le nombre d'exemples dans chacune des deux classes d'un problème binaire. La stratégie de départ est l'intensification. La valeur initiale (5.27) de k est choisie

23. Cela traduit le fait qu'une succession d'alternance entre une stratégie d'intensification et de diversification n'a pas permis de trouver un meilleur modèle que le dernier modèle $\boldsymbol{\theta}^*$ exploré.

24. Une autre solution pourrait être de choisir la valeur la plus proche de celle déterminée par la procédure directe de la section A.2

Algorithme 14 DIVERSIFICATION($\boldsymbol{\theta}^{it}$)

$\delta \leftarrow n_{\text{Echec}} + 1$
$i \leftarrow$ Tirage aléatoire d'une variable éligible
$\Theta_{\text{suivant}} \leftarrow$ DEUXMOUVEMENTPOSSIBLE($\boldsymbol{\theta}^{it}$,i,δ)
$\boldsymbol{\theta}^{it+1} \leftarrow$ MEILLEURESOLUTIONNONTABOUES(Θ_{suivant})
si $q_{\text{FD}}(\boldsymbol{\theta}^{it+1}) > q_{\text{FD}}(\boldsymbol{\theta}^{\text{diversification}}) \vee n_{\text{Diversification}} = 0$ **alors**
  $\boldsymbol{\theta}^{\text{diversification}} \leftarrow \boldsymbol{\theta}^{it+1}$
  si $q_{\text{FD}}(\boldsymbol{\theta}^{it+1}) > q_{\text{FD}}(\boldsymbol{\theta}^{*})$ **alors**
    $\boldsymbol{\theta}^{*} \leftarrow \boldsymbol{\theta}^{it+1}$
    $n_{\text{Echec}} \leftarrow 0$
    `Strategie` $\leftarrow$ INTENSIFICATION
  fin si
fin si
$n_{\text{Diversification}} \leftarrow n_{\text{Diversification}} + 1$
si $n_{\text{Diversification}} > n^{\max}_{\text{Diversification}}$ **alors**
  $\boldsymbol{\theta}^{\text{diversification}} \leftarrow \boldsymbol{\theta}^{it+1}$
  `Strategie` $\leftarrow$ INTENSIFICATION
fin si

suffisamment petite pour commencer avec une version suffisamment simplifiée de la base d'apprentissage afin de permettre que les premières itérations de notre méthode soient rapides pour produire les premières solutions avec une QFD élevée (*i.e.* les $\boldsymbol{\theta}^{*}$ dans l'algorithme 13). Ces solutions permettront de réaliser une sélection rapide des attributs et des valeurs des hyper-paramètres pour identifier une première région d'intérêt dans l'espace de recherche. Cette première caractérisation d'un modèle de bonne qualité sera utile pour la suite du processus de recherche avec tabous en limitant le nombre d'itérations utilisant les mouvements relatifs aux attributs. Une valeur initiale de k faible favorise également l'utilisation d'une solution parcimonieuse lorsque plusieurs solutions de QFD identiques existent.

5.3.2.6 Résultats expérimentaux

Plusieurs expérimentations sont réalisées pour mesurer l'efficacité de la méthode de sélection de modèle proposée pour produire des fonctions de décision efficaces en généralisation et de complexité réduite. Elles ont également pour objectif de mesurer l'influence de plusieurs paramètres de la méthode. Dans les tableaux relatifs à ces expérimentations les notations suivantes sont utilisées:

– T_{app} désigne la durée en secondes nécessaires à notre méthode pour la sélection d'un modèle $\boldsymbol{\theta}^{*}$ qui est considéré comme la solution optimale du problème. Cette durée inclut les temps nécessaires à la production des différentes versions simplifiées de la base d'entraînement avec l'algorithme 10.

$n^{\max}_{\text{SansAmélioration}}$	T_{app}	k	$log_{\sqrt{2}}(C)$	$log_{\sqrt{2}}(\sigma)$	n_{SV}	$q_{\text{BER}_{\text{v}}}$	q_{FD}
2	603	2	0	2	4	0,781	0,777
3	1025	2	1	4	4	0,793	0,789
5	2375	3	0	8	6	0,800	0,801
7	4067	3	2	8	6	0,801	0,802
10	7202	4	3	5	11	0,799	0,803

TAB. 5.13 – *Influence du paramètre $n^{\max}_{\text{SansAmélioration}}$ sur la qualité de fonction de décision produite et sur la durée T nécessaire à la sélection du meilleur modèle avec la base d'apprentissage* **Adult** *($c_{\text{VS}} = 10^{-3}$).*

- q_{FD} désigne la qualité de la fonction de décision produite avec le modèle $\boldsymbol{\theta}^*$ estimée à partir du critère (5.23). La valeur e_{BER} dans ce critère est déterminée à partir de de base de test Z^T (*cf.* algorithme 12),
- $q_{\text{BER}_{\text{v}}}$ est égale à $1-e_{\text{BER}}$ et l'erreur BER est évaluée à partir de la base de validation Z^V,
- n_{VS} correspond au nombre de vecteurs de support utilisés par la fonction de décision,
- n_{AS} correspond au nombre d'attributs sélectionnés relativement au modèle $\boldsymbol{\theta}^*$.

Le premier de ces paramètres est $n^{\max}_{\text{Echec}}$. Il représente le nombre d'échecs autorisés, ainsi que le nombre d'alternances entre la stratégie d'intensification et de diversification avant arrêt de la recherche avec tabous. S'il est évident que cette valeur ne doit pas être trop faible car un arrêt trop rapide limiterait les possibilités d'optimisation de la méthode, une valeur trop grande est inutile car l'amplitude des mouvements deviendrait trop grande et cela aurait également pour effet d'augmenter inutilement la durée de la recherche avec tabous. Par rapport aux domaines de définition des variables relatives à $\log_{\sqrt{2}}(C)$, $\log_{\sqrt{2}}(\sigma)$ et k, nous avons choisi $n^{\max}_{\text{Echec}} = 5$.

Le second de ces paramètres est $n^{\max}_{\text{SansAmélioration}}$. Il règle l'importance de la phase d'intensification par rapport à la phase de diversification. Les tests réalisés montrent qu'il ne doit pas être trop faible, car dans ce cas les solutions produites sont de qualités médiocres. Une augmentation conséquente de la valeur de $n^{\max}_{\text{SansAmélioration}}$ est inutile, car elle limite l'intérêt de la diversification et a pour conséquence une augmentation de la durée de la recherche sans augmentation significative de la QFD liée au modèle sélectionné. Le tableau 5.13 illustre cet effet et des résultats similaires ont été obtenus avec d'autres bases d'apprentissage. Une valeur de 5 pour $n^{\max}_{\text{SansAmélioration}}$ semble donc être un bon compromis.

Un autre paramètre important, lorsque la sélection d'attributs est réalisée avec notre méthode, est la valeur de η. Nous avons déjà évoqué que le fait de tester à chaque itération de la recherche avec tabous toutes les solutions liées à l'ajout ou la suppression d'un attribut est fortement pénalisant en temps de calcul, en particulier lorsque le nombre d'attributs est important. Dans la grande majorité des cas, l'ajout ou la suppression d'un attribut produit une modification faible de la qualité de la fonction de décision. Il semble alors logique de prendre pour η une valeur proche de 1 pour réaliser les optimisations fines liées aux attributs seulement lorsque les autres paramètres sont choisis avec efficacité. Les expérimentations réalisées avec différentes bases confirment ces remarques et une valeur de $\eta = 0{,}99$ semble un bon choix. Le tableau 5.14 illustre cet effet avec la base **Adult**. Il montre en particulier que la durée d'apprentissage augmente rapidement, si une valeur trop faible de η est choisie.

η	T_{app}	k	$log_{\sqrt{2}}(C)$	$log_{\sqrt{2}}(\sigma)$	n_{SV}	n_{AS}	$q_{\text{BER}_{\text{v}}}$	q_{FD}
0,995	6654	6	0	0	50	9	0,793	0,789
0,99	9508	1	-3	6	4	22	0,817	0,803
0,98	160179	4	0	8	18	20	0,811	0,804
0,95	195043	0	10	5	2	41	0,819	0,803
0	310047	12	1	5	3286	44	0,818	0,805

TAB. 5.14 – *Influence du paramètre η sur la qualité de fonction de décision produite et sur la durée T nécessaire à la sélection du meilleur modèle avec la base d'apprentissage* **Adult** *($c_{\text{VS}} = c_\beta = 10^{-3}$).*

Pour l'ensemble des expérimentations suivantes, les valeurs des paramètres $n^{\max}_{\text{Echec}}$, $n^{\max}_{\text{SansAmélioration}}$ et η sont celles qui ont été précédemment données. Le tableau 5.15, donne les résultats obtenus avec deux problèmes binaires qui ont déjà été utilisés en section 5.3.2.3. Notre méthode a été utilisée avec différents coefficients de pénalisation de la complexité. Lorsque l'on compare les résultats obtenus avec notre méthode (sans sélection d'attributs) avec ceux d'une recherche exhaustive (pour le même niveau de simplification) déductibles la figure 5.13, on observe que:

- les capacités de généralisation des deux fonctions de décision produites sont très proches,
- le nombre de vecteurs de support de ces deux fonctions de décision est pratiquement identique,
- la durée pour la sélection d'un modèle avec la méthode taboue est du même ordre de grandeur qu'une recherche exhaustive pour un niveau de simplification proche de celui choisi par notre méthode,
- la durée pour cette sélection d'un modèle est très inférieure à une procédure *grid search* classique sans simplification si la sélection d'attributs n'est pas réalisée,
- la durée pour la sélection d'un modèle incluant la sélection d'attributs reste exploitable.

Globalement, ces résultats illustrent que chaque solution optimale produite correspond à un compromis complexité/généralisation qui est fortement dépendant de la préférence exprimée par la valeur de c_{VS}. De plus, la diversité des compromis réalisés suivant que la base exploitée est **Adult** ou **Web** illustre que la nature des données influence grandement l'expression de ce compromis.

Adult: en observant la figure 5.13(a) seul un compromis favorisant fortement la réduction du taux d'erreur par rapport à celle de la complexité doit permettre d'améliorer significativement la valeur de q_{BER}, dans le cas contraire le modèle choisi doit correspondre à un niveau de simplification important. Les expérimentations réalisées avec notre méthode illustrent cet effet prévisible, car seule la valeur la plus faible de c_{VS} correspond à la sélection d'un modèle utilisant un niveau de simplification faible, les trois autres valeurs de c_{VS} plus grandes correspondent à la sélection de modèles qui utilisent tous un niveau de simplification important.

Web: en observant la figure 5.13(a) on remarque que plusieurs compromis existent suivant l'importance qui est accordée à la réduction de la complexité par rapport à la réduction du taux d'erreur. Les résultats dans le tableau 5.15 illustrent que trois types de fonctions de décision sont produites et elles correspondent respectivement à une forte, moyenne ou faible pénalisation de la complexité. L'effet sur les capaci-

		avec sélection d'attributs ($c_\beta = c_{\mathrm{VS}}$)						sans sélection ($c_\beta = 0$)				
Z	c_{VS}	T_{app}	k	n_{VS}	n_{AS}	$q_{\mathrm{BER_v}}$	q_{FD}	T_{app}	k	n_{VS}	$q_{\mathrm{BER_v}}$	q_{FD}
A	$1 \cdot 10^{-2}$	5634	0	2	44	0,815	0,762	1400	0	2	0,794	0,789
A	$5 \cdot 10^{-3}$	26122	1	2	28	0,814	0,784	5700	2	4	0,797	0,799
A	$2 \cdot 10^{-3}$	16095	4	12	44	0,819	0,793	2685	3	6	0,799	0,800
A	$1 \cdot 10^{-4}$	127096	10	764	55	0,818	0,817	7079	13	5274	0,817	0,811
W	$1 \cdot 10^{-1}$	4762	1	3	44	0,822	0,598	1736	1	3	0,841	0,695
W	$1 \cdot 10^{-2}$	25693	2	5	149	0,873	0,846	4378	5	39	0,875	0,835
W	$1 \cdot 10^{-3}$	197229	9	506	227	0,897	0,881	18127	11	730	0,904	0,898

TAB. 5.15 – *Influence du compromis entre capacité de généralisation et complexité de la fonction de décision produite. La colonne 'Z' contient un A ou un W suivant que la base* **Adult** *ou* **Web** *est utilisée par notre méthode.*

tés en généralisation de ces fonctions de décision est non négligeable, car entre la plus pénalisée et la moins pénalisée, il y a un écart de plus de 6%.

Lorsque la sélection d'attributs est réalisée, on observe incontestablement une augmentation de la durée de sélection d'un modèle, mais elle reste raisonnable par rapport à l'augmentation importante du nombre de mouvements à prendre en compte.

Ces résultats soulignent également l'efficacité de la stratégie d'intensification à travers l'utilisation du paramètre η. Pour la base **Adult**, les capacités en généralisation sont augmentées et la complexité réduite pour les fonctions de décision produites avec sélection d'attributs par rapport à celles sans sélection. Cette observation illustre l'effet positif que peut avoir la sélection d'attributs. Pour la base **Web**, cet effet n'est pas observé et correspond plus à un compromis entre réduction de la complexité et augmentation des capacités de généralisation.

Les résultats dans les tableaux 5.16, 5.17 et 5.18 correspondent à l'application de notre méthode pour produire des fonctions de décision simplifiées dans le cadre d'un schéma de combinaison *un-contre-tous* avec respectivement les problèmes multi-classes **Shuttle**, **ClassPixels** et **OpticDigits**. Pour chacune de ces bases notre méthode est appliquée avec et sans pénalisation de la complexité. Ces résultats illustrent que même sans pénalisation de la complexité, des versions plus ou moins simplifiées de la base initiale sont utilisées pour produire les fonctions de décisions binaires. Une pénalisation moyenne de la complexité permet de produire des fonctions de décision de complexités plus réduites avec une durée de sélection plus faible sans pour autant réduire significativement les capacités de généralisation des fonctions de décision binaire produites. La dernière ligne de ces trois tableaux concerne la fonction de décision multi-classe construite à partir de la combinaison des n_ω fonctions de décision binaire. A la lecture des dernières lignes de ces tableaux, on remarque que:

– le taux $\tilde{q}_{\mathrm{BER_v}}$ de la fonction multi-classe est faiblement dégradée par la combinaison de ces multiples simplifications,

– le nombre d'attributs utilisés $\cup\beta_i$ [25] par le schéma multi-classe peut être important, car chaque sélection est indépendante et ne tient pas compte des attributs sélectionnés pour les autres fonctions de décision binaire.

25. $\cup\beta_i = \sum I(\exists\omega = j : \beta_{i,j} = 1)$ avec $\beta_{i,j}$ qui désigne si le $i^{\text{ème}}$ attribut est utilisé par la $j^{\text{ème}}$ fonction de décision binaire intervenant du schéma de décomposition.

	$c_{\rm VS} = c_\beta = 10^{-2}$					$c_{\rm VS} = c_\beta = 0$				
FD	$T_{\rm app}$	k	$n_{\rm VS}$	$n_{\rm AS}$	$q_{\rm BER_v}$	$T_{\rm app}$	k	$n_{\rm VS}$	$n_{\rm AS}$	$q_{\rm BER_v}$
1	207	4	7	2	99,85%	38106	15	127	3	99,83%
2	67	0	2	1	99,93%	14062	10	20	3	99,95%
3	45	0	2	1	99,94%	7948	11	38	3	99,95%
4	152	5	9	2	99,91%	31027	14	63	4	99,94%
5	44	3	2	1	99,98%	36637	7	13	2	99,96%
6	113	2	5	1	99,97%	394	6	24	6	99,97%
	$\sum T_{\rm app}$	$1/n_\omega \sum k$	$\sum n_{\rm VS}$	$\cup\beta_i$	$\tilde{q}_{\rm BER_v}$	$\sum T_{\rm app}$	$1/n_\omega \sum k$	$\sum n_{\rm VS}$	$\cup\beta_i$	$\tilde{q}_{\rm BER_v}$
	628	2,33	27	6	99,93%	128174	10,50	285	9	99,95%

TAB. 5.16 – *Sélection de modèle avec un schéma* un-contre-tous *(la colonne* **FD** *donne le numero de la classe singleton pour chacun des problèmes binaires) avec la base* **Shuttle**.

	$c_{\rm VS} = c_\beta = 10^{-2}$					$c_{\rm VS} = c_\beta = 0$				
FD	T_{app}	k	$n_{\rm VS}$	$n_{\rm AS}$	$q_{\rm BER_v}$	T_{app}	k	$n_{\rm VS}$	$n_{\rm AS}$	$q_{\rm BER_v}$
1	3971	3	4	5	96,03%	10666	5	19	14	96,04%
2	9497	4	20	7	87,56%	10905	4	21	18	87,60%
3	5089	3	9	6	91,76%	9715	4	19	18	91,89%
	$\sum T_{\rm app}$	$1/n_\omega \sum k$	$\sum n_{\rm VS}$	$\cup\beta_i$	$\tilde{q}_{\rm BER_v}$	$\sum T_{\rm app}$	$1/n_\omega \sum k$	$\sum n_{\rm VS}$	$\cup\beta_i$	$\tilde{q}_{\rm BER_v}$
	18557	3,33	33	12	84,87%	31282	4,33	59	25	85,01%

TAB. 5.17 – *Sélection de modèle avec un schéma* un-contre-tous *pour la base* **ClassPixels**.

	$c_{\rm VS} = c_\beta = 10^{-2}$					$c_{\rm VS} = c_\beta = 0$				
FD	T_{app}	k	$n_{\rm VS}$	$n_{\rm AS}$	$q_{\rm BER_v}$	T_{app}	k	$n_{\rm VS}$	$n_{\rm AS}$	$q_{\rm BER_v}$
0	212	4	5	16	99,93%	395	4	16	64	100,0%
1	694	4	23	29	99,71%	2406	6	95	21	100,0%
2	288	4	10	18	99,93%	328	4	29	31	100,0%
3	221	2	7	14	98,70%	335	4	19	25	99,28%
4	271	4	10	19	100,0%	531	4	27	62	99,35%
5	242	4	8	14	99,93%	416	5	29	17	100,0%
6	248	2	7	22	100,0%	4169	4	32	64	100,0%
7	254	3	9	23	99,78%	330	4	12	27	99,78%
8	767	6	32	22	97,85%	4597	8	193	59	97,85%
9	563	4	32	29	98,25%	575	4	32	29	98,32%
	$\sum T_{\rm app}$	$1/n_\omega \sum k$	$\sum n_{\rm VS}$	$\cup\beta_i$	$\tilde{q}_{\rm BER_v}$	$\sum T_{\rm app}$	$1/n_\omega \sum k$	$\sum n_{\rm VS}$	$\cup\beta_i$	$\tilde{q}_{\rm BER_v}$
	3569	3,7	143	59	97,35%		14082	484	64	97,42%

TAB. 5.18 – *Sélection de modèle avec un schéma* un-contre-tous *pour la base* **OpticDigits**.

Globalement ces résultats illustrent qu'il est possible de produire un schéma *un-contre-tous* efficace et de complexité réduite en simplifiant séparément chaque fonction de décision binaire utilisée dans ce schéma de combinaison.

5.3.2.7 Discusion

Cette seconde approche montre que les techniques de catégorisation peuvent être utiles à la définition de méthodes de simplifications des fonctions de décision produites pas les SVM. La sélection d'un modèle complexe qui correspond à la recherche des hyper-paramètres d'un SVM, du niveau de simplification et des attributs utiles peut être réalisée dans des temps raisonnables grâce à l'utilisation judicieuse de la recherche avec tabous et de la définition de stratégies d'intensification et de diversification adaptées à cette méthode. Plusieurs remarques peuvent être faites pour des évolutions futures de notre méthode.

Nombre de prototypes différent par classe

Dans la version actuelle de notre méthode, le nombre de prototypes est le même pour les deux classes. Si ce choix a l'avantage d'équilibrer les classes, il est possible que la variance entre les exemples relatifs à une classe soit plus ou moins grande que celle des exemples relatifs à l'autre classe. La fusion de classes du problème initial pour produire des sous-problèmes binaires peut accentuer cet effet (notamment avec la décomposition *un-contre-tous*). Pour prendre en compte cette possibilité, il est préférable que le nombre de prototypes pour chaque classe ne soit pas déduit à partir d'une seule variable k dans la seconde approche.

Exploiter d'autres algorithmes de catégorisation

Nous avons signalé précédemment que l'algorithme LBG est un des algorithmes de catégorisation qui peut être utilisé pour réaliser la production de prototypes. Cependant, d'autres algorithmes de catégorisation pourraient être exploités. L'algorithme BIRCH [ZHANG96] est l'un d'eux, car il définit une représentation hiérarchique compacte et peut être plus facilement appliqué à des bases de données plus volumineuses. Il a déjà été utilisé avec les SVM [YU03B], mais sans la sélection globale d'un modèle intégrant également les hyper-paramètres (il n'y a pas également de sélection d'attributs). Un autre algorithme de catégorisation qui pourrait être utilisé à la place de LBG est Chamelon [KARYPI99]. Il a l'avantage d'utiliser des structures de graphes qui permettent d'identifier des regroupements de formes non hyper-sphériques.

Exploiter des techniques d'*alpha seeding*

Notre méthode a des temps exploitables pour la sélection d'un modèle $\boldsymbol{\theta}$, mais il est possible de les améliorer en utilisant des techniques d'*alpha seeding* et des critères d'arrêt différents pour l'algorithme SMO. En effet, un mouvement de la recherche avec tabous change lègèrement la solution (en particulier avec la stratégie d'intensification), il est alors possible d'utiliser le résultat de la solution actuelle pour définir de meilleures conditions d'initialisation pour les phases d'entraînement relatives à l'ensemble des mouvements à tester. La définition de ces techniques suivant la nature du mouvement à prendre en compte reste à définir. Il est alors préférable dans ce cas que les prototypes soient systématiquement des exemples de la base d'apprentissage. Cela est facilement réalisable avec l'algorithme LBG en remplaçant chaque centroïde par l'exemple le plus proche de ce centroïde.

Amélioration du critère de qualité

Le critère de qualité d'une fonction de décision utilisée comme compromis précision/complexité est actuellement fortement empirique dans sa définition. La définition d'un critère moins empirique à partir de théorie de MDL (Minimal Description Length) [RISSAN99] doit aussi être envisagée.

Hybridation des deux approches

Les deux approches pourraient être combinées lorsque les bases d'apprentissage sont trop volumineuses (rendant par la même inutile, l'utilisation de la méthode de pré-filtrage, qui était de toute façon peu convaincante). Les résultats avec la base **OpticDigits** (*cf.* tableau 5.18) illustrent l'efficacité de la méthode lorsque l'on compare les résultats avec ceux des tableaux A.5 et 5.11 [26]. La difficulté réside dans le choix d'un compromis complexité/généralisation adéquat pour produire une base simplifiée suffisamment réduite pour qu'elle soit exploitable par la première approche sans que la réduction ne soit trop grande pour profiter des bénéfices de la première approche. De plus, la seconde approche aura l'avantage de fournir des indications précieuses sur les bonnes valeurs des hyper-paramètres pour la première approche.
A plus long terme, une autre solution est de définir une hybridation de ces deux approches. La recherche de prototypes pertinents sur des versions simplifiées de la base d'apprentissage serait directement pilotée par la recherche avec tabous qui réaliserait en même temps la sélection des hyper-paramètres des SVM. L'utilisation de techniques d'*alpha seeding* serait alors encore plus intéressante afin de produire une méthode dont les temps de calcul soient les plus faibles possible.

5.3.3 Conclusion

Après avoir présenté les objectifs de ce chapitre, nous avons insisté sur l'importance de la sélection d'un modèle efficace pour produire une fonction de décision de complexité réduite. Nous avons présenté différentes méthodes proposées par différents auteurs qui ont en commun la sélection de modèles avec des SVM et la simplification des fonctions de décision (SVM ou règle PPV). Nous avons ensuite proposé deux approches pour produire des fonctions de décision performantes et de complexité réduite.

La première approche est basée sur un principe de minimisation du risque structurel et elle définit un schéma de compression guidé par des mesures de confiance. Cette approche à pour but de ne conserver que les exemples les plus pertinents de la base d'apprentissage et de choisir les hyper-paramètres appropriés aux SVM. Nous insistons sur le fait que la notion d'exemples pertinents est difficile à formaliser et qu'elle est dépendante de l'algorithme d'apprentissage utilisé. Nous montrons qu'un parallèle est réalisable entre la notion d'exemples pertinents et la notion d'attributs pertinents. Nous proposons alors un algorithme générique pour réaliser la sélection itérative des exemples pertinents dans le cadre d'un principe de minimisation du risque structurel. Nous proposons 4 critères pour mesurer la pertinence globale des exemples et guider notre algorithme dans la sélection itérative des exemples pertinents afin de produire une règle PPV minimal (en nombre d'exemples) et cohérente avec l'ensemble d'apprentissage. Plusieurs résultats expérimentaux illustrent que deux de ces critères sont plus performants si la sélection d'exemples pertinents s'arrête avant d'atteindre la cohérence avec la base d'apprentissage. Nous proposons deux critères d'arrêt relatifs à ces deux critères de pertinence afin d'améliorer les capacités en généralisation et l'importance de la simplification réalisée. A partir des dif-

26. La comparaison de ces résultats doit être prise avec précaution, car le schéma de combinaison multi-classe n'est pas le même dans les différents cas. Cependant, les expérimentations du chapitre 6 illustre que la différence entre un schéma *un-contre-un* et *un-contre-tous* est généralement faible.

férentes formulations proposées, un schéma de simplification en cascade est défini 5.5. Les résultats obtenus avec ce schéma montrent qu'il est possible de produire des fonctions de décision de complexités réduites tout en améliorant les capacités de généralisation des SVM. Plusieurs perspectives sont proposées pour améliorer et étendre cette première approche.

La seconde approche réalise la réduction du nombre d'exemples utilisés par la fonction de décision d'un SVM en exploitant une technique de catégorisation. L'objectif est de produire des prototypes représentatifs de la base initiale. Un algorithme LBG issu de la quantification vectorielle est utilisé pour produire des versions plus ou moins simplifiées de la base d'apprentissage. Le schéma de simplification réalise la sélection d'un modèle qui inclut le choix du niveau de simplification, la sélection des attributs utiles et les hyper-paramètres des SVM. Un nouveau critère de qualité pour les fonctions de décision est défini et il correspond à un compromis précision/complexité. La sélection d'un modèle optimisant ce nouveau critère de qualité est réalisée à partir de la définition d'une recherche avec tabous adaptée à ce problème d'optimisation. L'utilisation de cette métaheuristique permet une approche globale (*i.e.* non cascadée) de la sélection de modèle. Les résultats expérimentaux montrent que notre méthode produit, dans des temps exploitables, des fonctions de décision de complexité réduite qui conservent de bonnes capacités de généralisation. Plusieurs perspectives sont proposées pour améliorer ou étendre cette seconde approche.

Globalement ces deux approches illustrent l'intérêt de produire des fonctions de décision simplifiées avec les SVM. La première approche montre qu'il est possible de produire des fonctions de décision plus performantes en généralisation en écartant les exemples les moins pertinents de l'ensemble d'apprentissage d'un SVM et que cela a pour conséquence de produire des fonctions de décision simplifiées. Deux bases de données utilisés dans deux applications (*cf.* chapitre 7) ont des capacités de prédiction améliorées grâce à l'utilisation de cette technique. On peut noter que, pour ces deux applications, la réduction des temps de décision n'était pas l'objectif prioritaire. La seconde approche montre qu'il est possible de réaliser un compromis efficace entre réduction des temps de décision et capacités prédictives de la fonction de décision à partir de la recherche de prototypes représentatifs de la base d'apprentissage. Un algorithme de catégorisation issu de la quantification vectorielle est utilisé pour produire les prototypes, mais le principe général de notre méthode est suffisamment générique pour pouvoir exploiter d'autres algorithmes de catégorisation. Pour cette seconde approche la possibilité réduire les temps de décision est plus importante. Cette possibilité est exploitée pour produire un classificateur de pixels rapide pour la segmentation d'images couleur médicales (*cf.* chapitre 7).

L'intérêt porté à la production de fonctions de décision de complexité réduite avec des SVM est croissant. De nouvelles techniques ont été proposées récemment [KEERTH06, KATAGI06]. La comparaison entre ces nouvelles méthodes et nos approches (et leurs évolulons) devra être réalisée par la suite.

Une autre possibilité applicative non évoquée précédemment, liée à notre première approche, est le fait de pouvoir montrer à l'oracle, lors de la constitution d'une base, quels sont les exemples les plus et les moins pertinents, ainsi que ceux qui sont redondants ou trop éloignés de la frontière de décision. Ces informations peuvent être utiles à l'oracle pour optimiser la constitution d'une base d'apprentissage lorsque la production de nouveaux exemples a un coût important.

CHAPITRE 6

SCHÉMAS MULTI-CLASSES DE COMBINAISON AVEC DES SVM

Sommaire

Pour aborder un problème de classification supervisée avec une base multi-classe deux approches existent [HSU02, RIFKIN04, GUERME05]. La première [GUERME05] correspond à l'optimisation d'un problème unique à partir de l'ensemble des données. La seconde [HSU02, RIFKIN04] correspond à définir un ensemble de sous-problèmes binaires, puis à réaliser l'optimisation de chacun de ces sous-problèmes. Cette seconde approche produit plusieurs fonctions de décision binaires et il est nécessaire de définir la façon de les combiner pour produire au final une décision multi-classe. La première approche semble la plus simple et elle est facile à réaliser avec des méthodes d'apprentissage comme les réseaux de neurones, les arbres de décision, la règle *k*-PPV, etc. car les fonctions de décision produites permettent de sélectionner directement une classe parmi un ensemble de n_c classes ($n_c \geq 2$). Pour les SVM, la fonction de décision produite est par nature binaire, ce qui rend la première approche problématique. Des extensions multi-classes ont été proposées [CRAMME01, DARCY05], mais elles sont généralement difficiles à exploiter [RIFKIN04, ABE05]. La seconde approche a plusieurs avantages, en particulier pour les SVM [ALLWEI00, ABE05], mais également pour les algorithmes d'apprentissage qui permettent d'appréhender directement des problèmes multi-classes [PRICE94, FÜRNK01, SAVICK03, OU04, LEZORA05]. La résolution globale d'un problème multi-classe est intrinsèquement plus difficile que chaque problème binaire pris séparément, car un grand nombre de frontières de décision doivent être apprises simultanément, alors que seule la frontière entre deux classes doit l'être avec un problème binaire. De plus, une sélection de modèles efficaces, correspondant à la recherche des attributs ou exemples pertinents et choix des valeurs des paramètres propres à l'algorithme d'apprentissage utilisé, peut être réalisée pour chaque classificateur binaire. Cela permet de spécialiser chaque fonction de décision à chaque problème binaire. Dans le cas d'une approche globale multi-classe, la sélection de modèle est forcément globale et soit elle n'a aucune correspondance possible avec l'approche par combinaison, soit elle nécessite la sélection d'un modèle dans un espace de recherche de dimension beaucoup plus grande. Dans l'autre cas, la résolution de chaque sous-problème binaire est plus rapide et facile à réaliser car chaque sous-problème est intrinsèquement plus facile que le problème initial. La réduction des temps d'entraînement est plus marquée si le nombre de données utilisées pour chaque sous-problème binaire est plus réduit que ceux du problème multi-classe.

Dans ce chapitre nous nous intéressons à la production de fonctions de décision performantes dans le cadre de la combinaison de classificateurs binaires. Dans une première partie de ce chapitre nous donnons un état de l'art relatif aux différentes approches proposées pour réaliser des schémas de combinaison de classificateurs binaires. La seconde partie propose deux approches pour améliorer les capacités de généralisation de schémas de combinaison à partir de SVM. La première approche consiste à définir un schéma de combinaison hybride basé sur l'utilisation conjointe d'une décomposition *un-contre-un* et *un-contre-tous*. La seconde approche optimise globalement, à partir d'un algorithme évolutionnaire, l'ensemble des hyper-paramètres des SVM intervenant dans un schéma de combinaison. Une discussion en fin de chapitre propose différentes perspectives liées à la combinaison de classificateurs binaires.

6.1 Schémas de combinaison de SVM binaires

La définition d'un schéma de combinaison correspond (1) à définir l'ensemble des sous-problèmes binaires à résoudre, (2) à produire l'ensemble des fonctions de décision binaires correspondant à ces sous-problèmes binaires puis, (3) à définir un principe de décodage qui permet de sélectionner la classe la plus vraisemblable à partir des prédictions réalisées par l'ensemble des fonctions de décision binaire. Introduisons quelques notations avant d'aborder les étapes (1) et (3) d'un schéma de combinaison (l'étape (2) est pour cet état de l'art considéré comme triviale). Un sous-problème de discrimination binaire issu d'un problème multi-classe est défini à partir d'un vecteur $\boldsymbol{\psi}$. Ce vecteur représente pour chaque classe ω_y de la base initiale sa correspondance avec des exemples considérés comme appartenant à la classe positive ($\psi_y = +1$), à la classe négative ($\psi_y = -1$) ou non présent ($\psi_y = 0$) pour le problème binaire induit. La définition d'une base d'apprentissage binaire $Z(\boldsymbol{\psi})$ à partir d'une base multi-classe Z et de la transformation binaire induite par $\boldsymbol{\psi}$ est la suivante:

$$Z(\boldsymbol{\psi}) = \{(\mathbf{x}, +1) \,|\, (\mathbf{x},y) \in Z, \psi_y = +1\} \cup \{(\mathbf{x}, -1) \,|\, (\mathbf{x},y) \in Z, \psi_y = -1\} \qquad (6.1)$$

$Z(\boldsymbol{\psi})^E$, $Z(\boldsymbol{\psi})^T$ et $Z(\boldsymbol{\psi})^V$ correspondent respectivement aux bases d'entraînement, de test et de validation relatives aux problèmes de discrimination induits par le vecteur $\boldsymbol{\psi}$. $D(\boldsymbol{\psi})$ désigne une fonction de décision produite à partir d'une base d'entraînement $Z(\boldsymbol{\psi})^E$. La valeur d'une de ces fonctions de décision binaires pour un exemple $\mathbf{x}$ est notée $D(\boldsymbol{\psi},\mathbf{x})$.

6.1.1 Méthodes de décomposition

La décomposition d'un problème multi-classe en k problèmes binaires est la première étape d'un schéma de combinaison binaire. Elle correspond à définir k vecteurs $\boldsymbol{\psi}$. A partir de ces vecteurs, une matrice Ψ de codage du problème multi-classe est définie.

$$\Psi = \begin{array}{c|ccc} & \omega_1 & \cdots & \omega_{n_c} \\ \hline \boldsymbol{\psi}_1 & \psi_{1,1} & \cdots & \psi_{1,n_c} \\ \vdots & \vdots & \ddots & \\ \boldsymbol{\psi}_k & \psi_{k,1} & & \psi_{k,n_c} \end{array} \qquad (6.2)$$

Chaque ligne de Ψ correspond à un des vecteurs $\boldsymbol{\psi}_i$ ($1 \leq i \leq n_c$) et chaque colonne de Ψ indique le problème binaire dans lequel les exemples d'une classe donnée sont utilisés. La structure de la matrice Ψ défini la nature du schéma de combinaison utilisé. Dans la majorité des schémas de combinaison utilisés, cette structure est prédéfinie [DIETTE95, VAPNIK98, KREBEL99]. Pour d'autres méthodes, la structure de la matrice (les différentes décompositions binaires) est produite par un schéma d'apprentissage [PÉREZ-02, ABE05]. Si la seconde approche a la possibilité de s'adapter à la nature d'un problème multi-classe, elle est plus difficile à réaliser et demande des temps d'apprentissage plus importants. Cette seconde approche reste un domaine de recherche ouvert. La première approche est beaucoup plus utilisée en pratique pour des raisons de simplicité car elle permet d'obtenir globalement de bons résultats. Parmi les méthodes de décomposition utilisant une matrice prédéfinie Ψ, on peut citer les méthodes: *un-contre-tous* [VAPNIK98], *un-contre-un* [KREBEL99], DDAG (*Decision Directed Acyclic Graph*) [PLATT99C]. La méthode *moitié contre moitié* [LEI05] ou celles basées sur l'utilisation de matrices ECOC (*Error-Correcting Output Codes*) [DIETTE95, ALLWEI00] peuvent utiliser une matrice arbitraire ou définie par un processus d'apprentissage [PÉREZ-02, VURAL04]. Moreira et al [MOREIR98] proposent par exemple une méthode de décomposition, que nous noterons IPCCC (*Improved Pairwise Coupling with Correcting Classifiers*), qui utilise conjointement deux matrices de décomposition.

6.1.1.1 Un contre tous

La méthode *un-contre-tous* (*one against all*) est la plus simple et elle produit autant de problèmes binaires que de classes. Chaque problème binaire correspond à la discrimination des exemples d'une classe avec ceux des autres classes. La matrice (6.3) illustre la décomposition *un-contre-tous* pour un problème avec 4 classes.

$$\Psi = \begin{bmatrix} +1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 \\ -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & +1 \end{bmatrix} \tag{6.3}$$

Cette méthode de décomposition produit peu de problèmes binaires. Cependant, ceux-ci ont l'inconvénient d'utiliser systématiquement l'ensemble des exemples du problème multi-classe et par conséquent les temps d'apprentissage de chaque sous-problème binaire sont importants [LEI05]. La discrimination d'une classe par rapport à l'ensemble des autres est généralement un problème d'apprentissage difficile. Cette méthode produit également des problèmes binaires qui sont mals balancés, ce qui complique le problème de discrimination. Cependant, Rifkin et al [RIFKIN04] montrent, à partir de nombreuses expérimentations, que cette méthode produit généralement de bons résultats malgré sa simplicité. L'une des raisons possibles est liée au fait que c'est la décomposition qui produit le moins de problèmes binaires et que par conséquent le processus de décodage en est simplifié (*cf.* section 6.1.2).

6.1.1.2 Un contre un

La méthode *un-contre-un* (*One Against One*, *Round Robin*, ou *Pairwise Coupling*) est sûrement la méthode la plus utilisée. L'idée est que pour chaque sous-problème binaire

seule deux classes du problème initial sont retenues. Le nombre de problèmes binaires induits par cette décomposition est de $n_c(n_c-1)/2$. La matrice (6.4) illustre cette décomposition pour un problème avec 4 classes.

$$\Psi = \begin{bmatrix} +1 & -1 & 0 & 0 \\ +1 & 0 & -1 & 0 \\ +1 & 0 & 0 & -1 \\ 0 & +1 & -1 & 0 \\ 0 & +1 & 0 & -1 \\ 0 & 0 & +1 & -1 \end{bmatrix} \tag{6.4}$$

C'est la même règle de décomposition qui est utilisée avec la méthode DDAG, seul le principe de combinaison des fonctions de décision binaires change.

Avec cette décomposition, chaque problème binaire est de taille plus faible que le problème initial, ce qui réduit les temps d'entraînement liés à chaque problème de discrimination. Ce gain est atténué par l'augmentation du nombre de problèmes binaires à traiter. La discrimination est facilitée, car la frontière de décision ne concerne que deux classes à chaque fois. Une difficulté avec cette méthode est de définir une fonction de combinaison efficace à partir des fonctions de décision binaires produites. En effet, chaque classificateur est spécialisé dans la discrimination des exemples relativement à deux classes et lorsque qu'un exemple n'appartenant à aucune de ces deux classes est présenté à ce classificateur, la prédiction produite pour cet exemple n'est pas informative. Globalement, il n'y a que (n_c-1) classificateurs binaires pertinents pour un exemple donné et $(n_c-1)(n_c-2)/2$ classificateurs binaires non pertinents pour le même exemple. Plus le problème a de classes et plus le bruit introduit par l'ensemble des classificateurs non pertinents rend le décodage difficile. La section 6.1.2 propose plusieurs solutions pour répondre à ce problème. La méthode suivante permet également de pallier ce défaut.

6.1.1.3 IPCCC

La méthode de décomposition IPCCC (*Improved Pairwise Coupling with Correcting Classifiers*) utilise comme première matrice de décomposition celle d'une décomposition *un-contre-un* (6.4). Elle utilise également une deuxième matrice où chaque ligne correspond à la discrimination de deux classes par rapport aux autres. Dans la deuxième matrice, cela consiste à construire un classificateur binaire séparant les classes ω_i et ω_j des autres classes pour chaque problème binaire séparant les classes ω_i et ω_j dans (6.4). Il y a donc pour cette seconde décomposition $n_c(n_c-1)/2$ problèmes binaires produits[1]. Soit un total de $n_c(n_c-1)$ problèmes binaires. La matrice (6.5) illustre la seconde décomposition pour un problème avec 4 classes.

$$\Psi = \begin{bmatrix} +1 & +1 & -1 & -1 \\ +1 & -1 & +1 & -1 \\ +1 & -1 & -1 & +1 \\ -1 & +1 & +1 & -1 \\ -1 & +1 & -1 & +1 \\ -1 & -1 & +1 & +1 \end{bmatrix} \tag{6.5}$$

1. Dans la matrice (6.5), les lignes 3 à 6 sont redondantes avec les lignes 1 à 3. Ce n'est qu'un cas particulier pour $n_c = 4$, pour les autres valeurs de n_c, chaque problème binaire est distinct

Le but est d'utiliser les classificateurs liés à la seconde matrice pour déterminer la confiance dans les prédictions de ceux liés à la première matrice. La section 6.1.2 explique comment combiner ces deux ensembles de fonctions de décision binaires pour produire une décision multi-classe performante.

6.1.1.4 Moitié contre moitié

Pour cette méthode de décomposition, une ligne de la matrice Ψ correspond à une première dichotomie sur l'ensemble des classes et les autres lignes à des dichotomies sur les sous ensembles produits par la première dichotomie (et les suivantes). Le nombre de problèmes binaires est $2^{\lceil \log_2(n_c) \rceil} - 1$. Il est environ égal au nombre de classes n_c. La matrice (6.6) donne un exemple de décomposition pour un problème à 4 classes.

$$\Psi = \begin{bmatrix} +1 & +1 & -1 & -1 \\ +1 & -1 & 0 & 0 \\ 0 & 0 & +1 & -1 \end{bmatrix} \tag{6.6}$$

La nature de la décomposition est dépendante de la permutation des labels par rapport à l'ordre des colonnes de la matrice Ψ . Le choix d'une permutation parmi les $n_c!$ possibles peut avoir des influences sur les capacités en généralisation du schéma de combinaison produit. Dans [LEI05], il est démontré que les temps d'entrainement pour l'ensemble de ces problèmes binaires sont du même ordre que ceux de la méthode *un-contre-un*, lorsque la matrice Ψ est fixée abitrairement. Si la matrice Ψ est produite par un processus d'apprentissage, cette remarque n'est plus vraie.

6.1.1.5 ECOC (*Error Correcting Output Codes*)

Les matrices de décomposition précédentes sont des cas particuliers des méthodes $ECOC$, mais leurs structures font qu'elles ne sont pas les plus appropriées pour réaliser le décodage $ECOC$. Les matrices Ψ de type $ECOC$ comportent k lignes et le nombre de lignes désigne la longueur du code utilisé [DIETTE95]. Le principe est de produire un grand nombre de lignes (donc de décompositions binaires) et de choisir la classe la plus cohérente par rapport à l'ensemble de ces prédictions (*cf.* section 6.1.2.1). Le nombre maximum de lignes, sans classificateurs binaires redondants, dépend de la possibilité d'avoir ou non des valeurs nulles dans la matrice Ψ [DIETTE95, ALLWEI00]. Si le type de matrice Ψ choisi permet des valeurs nulles, le nombre maximum de lignes (non redondantes) est borné par $(3^{n_c} - 1)/2$. Dans le cas contraire, le nombre maximum de lignes est de

$(2^{n_c-1}-1)$. Les matrices (6.7) et (6.8) correspondent respectivement à un code exhaustif avec[2] et sans valeurs nulles dans Ψ.

$$\Psi = \begin{bmatrix} +1 & -1 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & +1 & -1 \\ +1 & -1 & -1 & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & -1 & -1 & +1 \\ +1 & -1 & -1 & -1 \\ \cdots & \cdots & \cdots & \cdots \\ -1 & +1 & +1 & -1 \end{bmatrix} \tag{6.7}$$

$$\Psi = \begin{bmatrix} +1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 \\ -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & +1 \\ -1 & -1 & +1 & +1 \\ -1 & +1 & -1 & +1 \\ -1 & +1 & +1 & -1 \end{bmatrix} \tag{6.8}$$

Dès que le nombre de classes augmente, il n'est plus possible de produire un code exhaustif et des techniques ont été proposées pour produire des matrices de tailles plus réduites qui ont de bonnes propriétés de décodage [DIETTE95, ALLWEI00]. Un des problèmes avec ces méthodes est qu'elles produisent généralement un nombre très important de fonctions de décision binaires. Ceci a pour conséquence d'augmenter la durée globale d'entraînement. Les méthodes de décomposition précédentes peuvent être traitées comme des cas particuliers de la méthode ECOC en ce qui concerne le décodage.

6.1.2 Principes de décodage ou de combinaison

La principale difficulté dans l'utilisation de schémas de combinaison de classificateurs binaires est le problème des inconsistances qui peuvent exister entre l'ensemble des prédictions produites par ces classificateurs binaires. La première raison de l'existence de ces inconsistances est liée à la méthode de décomposition utilisée. La figure 6.1 illustre ce premier type d'inconsistances avec des séparateurs linéaires pour les décompositions *un-contre-un* et *un-contre-tous*. On remarque que dans l'espace des données certaines régions ont plus d'une classe candidate, même si individuellement aucun classificateur binaire ne produit d'erreur relativement aux données connues. La deuxième raison est liée au fait que les classificateurs binaires ne sont généralement pas parfaits et produisent des erreurs de prédiction. Par exemple, pour une décomposition un-contre-tous, il est possible que tous les classificateurs binaires prédisent une classe négative, le problème est alors de définir comment choisir celui qui a réalisé une erreur. Plusieurs principes de décodage ont alors été définis pour permettre, à partir de l'ensemble des prédictions des classificateurs binaires, de choisir une classe unique. Certaines de ces méthodes permettent également d'avoir une estimation de confiance (généralement par une estimation de probabilité) de l'appartenance d'un exemple donné à chacune des classes du problème initial.

2. Seulement 6 des 25 lignes sont représentées.

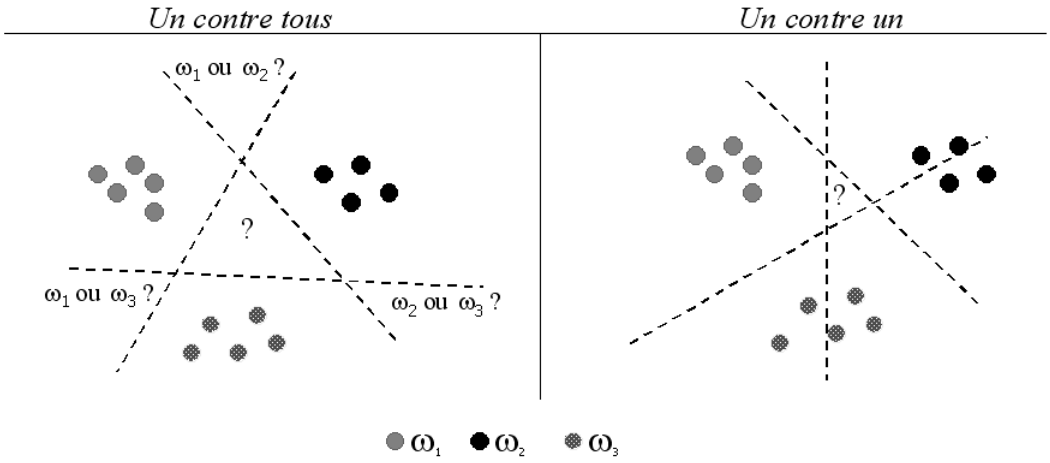


FIG. 6.1 – *Problèmes d'inconsistance (représenté par des ?) dans la sélection d'une classe unique à partir des frontières de décision binaires pour les décompositions* un-contre-un *et* un-contre-tous.

Il est possible de distinguer deux familles de principes de décodage (ou combinaison de classificateurs binaires). La première famille correspond à un décodage global. Il détermine tout d'abord la sortie de toutes les fonctions de décisions produites par la méthode de décomposition. A partir de l'ensemble de ces prédictions, une valeur de confiance (ou probabilité) est déterminée pour chaque classe du problème initial et la plus probable est choisie. La seconde famille correspond à un décodage en cascade et une procédure de sélection de classes est définie pour permettre de traiter le problème de l'inconsistance entre les différentes prédictions des fonctions de décision binaires. A chaque niveau de ce processus de décodage un sous ensemble restreint de classes candidates est retenu (directement ou par élimination d'autres classes). A la dernière étape de ce processus, une seule classe est finalement retenue.

Le décodage en cascade exploite généralement moins de fonctions de décision et il est par conséquent plus rapide en prédiction. Par contre, une erreur de prédiction d'un des classificateurs binaires peut suffire pour réaliser globalement une mauvaise prédiction avec ce type de décodage. Le décodage global peut par contre réussir à corriger certaines des erreurs de prédiction des classificateurs binaires.

6.1.2.1 Décodage global

Il existe un grand nombre de principes de décodage global, ainsi que différentes variantes, nous ne nous intéressons ici qu'aux plus courantes dans le cadre des SVM.

Vote majoritaire

Le vote majoritaire est certainement la méthode la plus utilisée. Elle est simple et applicable à l'ensemble des méthodes de décomposition. Le principe est de considérer qu'une fonction de décision binaire vote pour une classe (ou en ensemble de classes) suivant qu'elle prédit une classe positive ou négative pour le problème binaire induit. C'est le vec-

teur $\boldsymbol{\psi}$, associé à la fonction de décision binaire considérée, qui détermine les classes qui reçoivent un vote positif. Au final, la classe retenue est celle qui reçoit le maximum de votes. Le nombre de votes définit une mesure de confiance pour chaque classe. Lorsque la décomposition (matrice Ψ) ne réalise pas le même nombre de votes par classe (nombre de $+1$ et -1 dans une colonne de Ψ), c'est la proportion de votes par rapport au maximum pour une classe donnée qui est pris en compte. Cela peut se formaliser de la façon suivante:

$$p(\omega_i|\mathbf{x}) = \frac{|\text{Votes}(\omega_i,\mathbf{x},\Psi)|}{|\text{Votants}(\omega_i,\mathbf{x},\Psi)|} \tag{6.9}$$

avec

$$\text{Votes}(\omega_i,\mathbf{x},\Psi) = \{j|1 \leq j \leq k, \psi_{j,i} \neq 0, \psi_{j,i} \cdot D(\boldsymbol{\psi}_j,\mathbf{x}) > 0\} \tag{6.10}$$

et

$$\text{Votants}(\omega_i,\mathbf{x},\Psi) = \{j|1 \leq j \leq k, \psi_{j,i} \neq 0\} \tag{6.11}$$

La proportion $p(\omega_i|\mathbf{x})$ peut être utilisée comme une mesure de probabilité $p(\omega = \omega_i|\mathbf{x})$ après re-normalisation des valeurs.

Cutzu [CUTZU03] propose un autre principe de vote (dans le cadre d'une décomposition *un-contre-un*) basé sur les classes éliminées au lieu des classes élues par les classificateurs binaires, mais globalement le principe général relatif à la notion de vote reste le même.

Principes probabilistes

Si la sortie du classificateur correspond à une mesure de probabilité (en la classe positive par exemple), il est alors possible pour les méthodes de décomposition *un-contre-tous*, *un-contre-un* et IPCCC de déterminer une mesure de probabilité relative à chaque classe. Dans le cas des SVM, il est possible d'utiliser la méthode de Platt [PLATT99B] pour produire cette estimation de probabilité. Pour cette section nous considérons que toutes les fonctions de décision binaires $D(\boldsymbol{\psi},\mathbf{x})$ produisent une estimation de probabilité que l'exemple $\mathbf{x}$ soit de classe positive pour le problème binaire induit par $\boldsymbol{\psi}$.

Décomposition *un-contre-tous*

Pour la décomposition *un-contre-tous*, le calcul est direct et ne nécessite qu'une normalisation:

$$p(\omega = \omega_i|\mathbf{x}) = \frac{D(\boldsymbol{\psi}_i,\mathbf{x})}{\sum_{j=1}^{n_c} D(\boldsymbol{\psi}_j,\mathbf{x})} \tag{6.12}$$

Décomposition IPCCC

Pour la décomposition IPCCC, les deux matrices (6.4) et (6.5) de décomposition définissent deux types de fonctions de décision binaires.
Celles du premier type fournissent une estimation de la probabilité qu'un exemple $\mathbf{x}$ soit de la classe ω_i connaissant la restriction à deux classes ω_i et ω_j ($\omega_i \neq \omega_j$):

$$p(\omega = \omega_i|\ \mathbf{x}, \omega \in \{\omega_i,\omega_j\}) = D\left(\boldsymbol{\psi}^{(1)}_{n_{i,j}},\mathbf{x}\right) \tag{6.13}$$

avec $\boldsymbol{\psi}^{(1)}$ un vecteur associé à la première décomposition et $n_{i,j}$ l'indice du vecteur $\boldsymbol{\psi}^{(1)}$ correspondant au sous problème binaire de discrimination de la classe ω_i avec la classe ω_j [3]. Celles du second type fournissent une estimation de la probabilité qu'un exemple soit de la classe ω_i ou de la classe ω_j ($\omega_i \neq \omega_j$):

$$p(\omega \in \{\omega_i,\omega_j\}|\ \mathbf{x}) = D\left(\boldsymbol{\psi}^{(2)}_{n_{i,j}},\mathbf{x}\right) \tag{6.14}$$

A partir des expressions (6.13) et (6.14) associées à un couple $(\boldsymbol{\psi}^{(1)}_{n_{i,j}},\boldsymbol{\psi}^{(2)}_{n_{i,j}})$, on remarque qu'il est possible d'estimer la probabilité qu'un exemple soit de la classe ω_i par le produit de (6.13) par (6.14). Il y a (n_c-1) façons de réaliser cette estimation. La méthode proposée [MOREIR98] correspond à prendre la moyenne de ces estimations:

$$p(\omega = \omega_i|\ \mathbf{x}) = \frac{1}{N}\sum_{\substack{j=1\\ j\neq i}}^{n_c} D\left(\boldsymbol{\psi}^{(1)}_{n_{i,j}},\mathbf{x}\right)\cdot D\left(\boldsymbol{\psi}^{(2)}_{n_{i,j}},\mathbf{x}\right) \tag{6.15}$$

avec $N = \frac{n_c(n_c-1)}{2}$ un coefficient de normalisation. Les termes (6.14) ont un rôle de classificateurs correcteurs des estimations (6.13) pour la détermination des $p(\omega = \omega_i|\ \mathbf{x})$.

Décomposition *un-contre-un*

Dans une méthode de décomposition *un-contre-un*, il n'y a pas de classificateurs correcteurs. La difficulté est alors de définir les valeurs des $p(\omega = \omega_i|\ \mathbf{x})$ à partir des estimations des $p(\omega = \omega_i|\ \mathbf{x}, \omega \in \{\omega_i,\omega_j\})$ réalisées à partir de l'expression (6.13). Idéalement, les estimations $p(\omega = \omega_i|\ \mathbf{x})$ doivent satisfaire la relation suivante :

$$\forall\omega_i : p(\omega = \omega_i|\mathbf{x},\omega \in \{\omega_i,\omega_j\}) = \frac{p(\omega = \omega_i|\mathbf{x})}{p(\omega = \omega_i|\mathbf{x}) + p(\omega = \omega_j|\mathbf{x})} \tag{6.16}$$

En pratique, la relation (6.16) ne peut être satisfaite car les données déterminées par (6.13) sont plus ou moins bruitées. Plusieurs principes ont alors été proposés [WU04] pour produire des solutions qui approximent (6.16) en relâchant les contraintes :

$$\forall\omega_i : p(\omega = \omega_i|\mathbf{x},\omega \in \{\omega_i,\omega_j\}) = \frac{p(\omega = \omega_i|\mathbf{x})}{p(\omega = \omega_i|\mathbf{x}) + p(\omega = \omega_j|\mathbf{x})} + \varepsilon_i \tag{6.17}$$

Le but est que les erreurs représentées par les ε_i soient faibles. Nous ne citons que les principes les plus connus qui approchent cet objectif (pour plus de détails sur d'autres variantes se référer à [WU04]) :

– **Méthode basée sur le vote majoritaire :**
Ce premier principe est basé sur l'utilisation du vote majoritaire pour réaliser l'approximation (6.17). L'estimation de la probabilité en chaque classe est la suivante:

$$p(\omega = \omega_i|\mathbf{x}) = \frac{1}{N}\sum_{\substack{j=1\\ j\neq i}}^{n_c} I\left(D\left(\boldsymbol{\psi}^{(1)}_{n_{i,j}},\mathbf{x}\right) > 0{,}5\right) \tag{6.18}$$

3. La relation suivante $D\left(\boldsymbol{\psi}^{(1)}_{n_{j,i}},\mathbf{x}\right) = 1 - D\left(\boldsymbol{\psi}^{(1)}_{n_{i,j}},\mathbf{x}\right)$ est utilisée si ω_i n'est pas la classe positive dans le problème binaire induit.

avec $N = \frac{n_c(n_c-1)}{2}$ un coefficient de normalisation et $I(v)$ une fonction qui vaut 1 si v est vrai et 0 sinon.

– **Méthode de Price et al :**
Price et al [PRICE94] démontrent que:

$$\sum_{\substack{j=1 \\ j\neq i}}^{n_c} p(\omega = \omega_i|\mathbf{x},\omega \in \{\omega_i,\omega_j\}) - (n_c - 2)p(\omega = \omega_i|\mathbf{x}) = 1 \tag{6.19}$$

A partir de (6.16) et (6.19) ils obtiennent alors la relation suivante:

$$p(\omega = \omega_i|\mathbf{x}) = \frac{1}{\sum\limits_{\substack{j=1 \\ j\neq i}}^{n_c} D\left(\boldsymbol{\psi}^{(1)}_{n_{i,j}},\mathbf{x}\right) - (n_c - 2)} \tag{6.20}$$

– **Méthode d'Hastie et al:**
Hastie et al [HASTIE97] proposent de choisir les valeurs $p(\omega = \omega_i|\mathbf{x})$ de façon à ce qu'elles minimisent une mesure d'entropie relative (la distance de *Kullback-Leibler* [KULLBA51]) entre les valeurs déduites de l'application de la relation (6.16) et celles prédites par les fonctions de décision binaires $D\left(\boldsymbol{\psi}^{(1)}_{n_{i,j}},\mathbf{x}\right)$. Notons respectivement ces deux types d'estimation $p_{i,j}$ et $D_{i,j}$. Notons également p_i les valeurs de $p(\omega = \omega_i|\mathbf{x})$ et $\mathbf{p}$ le vecteur contenant l'ensemble des valeurs p_i. L'objectif est de minimiser la fonctionnelle $l(\mathbf{p})$ suivante:

$$l(\mathbf{p}) = \sum_{i\neq j} n_{i,j} D_{i,j} \ln \frac{D_{i,j}}{p_{i,j}} \tag{6.21}$$

avec $p_{i,j} = p_i/(p_i + p_j)$ et $n_{i,j}$ le nombre d'exemples dans la base d'entraînement binaire correspondante. Hastie et al proposent alors un algorithme itératif simple pour réaliser la minimisation de (6.21).

– **Méthode Wu et al:**

Wu et al [WU04] montrent que la méthode de Hastie pose certains problèmes, car elle est d'autant plus imprécise que la relation suivante : $i \neq j \rightarrow p_i + p_j \approx 2/n_c$, n'est pas vérifiée pour plusieurs couples (i,j). Dans la majorité des cas, les valeurs de $\mathbf{p}$ sont cependant fortement éloignées de ces conditions. Wu et al proposent alors de résoudre le problème suivant:

$$\forall i : p_i = \sum_{\substack{j=1 \\ j\neq i}}^{n_c} \left(\frac{(p_i + p_j)}{n_c - 1}\right) D_{i,j} \tag{6.22}$$

sous la contrainte $\sum\limits_{i=1}^{n_c} p_i = 1$. On remarque le lien avec la relation (6.15) (à un facteur de normalisation près), sauf que sans classificateurs correcteurs la valeur de $(p_i + p_j)$ doit être déduite. Wu et al montrent que le problème (6.22) est équivalent à la minimisation d'un problème dual convexe et ils proposent un algorithme pour le résoudre[4].

4. D'autres principes sont proposés par Wu et al [WU04] pour déduire des valeurs efficaces de $\mathbf{p}$ à partir des valeurs des $D_{i,j}$, mais leurs intérêts restent anecdotiques par rapport au principe énoncé ici.

Codes correcteurs d'erreurs (ECOC)

Dans la section 6.1.2.1 nous avons vu que la décomposition ECOC produit une matrice Ψ de k lignes et que pour chaque ligne une fonction de décision est produite. Notons $D_C(\mathbf{x}) = (I(D(\boldsymbol{\psi}_1,\mathbf{x}) > 0), \cdots, I(D(\boldsymbol{\psi}_k,\mathbf{x}) > 0))$ le code produit par l'ensemble de ces fonctions de décision pour un exemple $\mathbf{x}$. Si le code $D_C(\mathbf{x})$ correspond à une colonne i de la matrice Ψ, le choix de la classe d'appartenance de l'exemple $\mathbf{x}$ est direct et correspond à ω_i. Par exemple pour la matrice (6.8), la classe d'un exemple $\mathbf{x}$ est ω_2 si $D_C(\mathbf{x}) = (-1,+1,-1,-1,-1,+1,+1)$. Dans le cas où $D_C(\mathbf{x})$ ne correspond à aucune colonne de Ψ, Deitterich et al [DIETTE95] proposent d'utiliser des techniques de code correcteur d'erreurs pour déterminer la classe d'un exemple. La capacité de correction des erreurs dépend de la constitution de la matrice Ψ. Soit $d_H(C_i(\Psi),C_j(\Psi))$, la distance de Hamming entre les codes constitués par les colonnes i et j (notées $C_i(\Psi)$ et $C_j(\Psi)$) de la matrice Ψ, et $d_H^{\min}$ la valeur minimum de cette distance pour l'ensemble des couples (i,j) avec $i \neq j$. La capacité de correction est au moins de $\lfloor \left(d_H^{\min} - 1\right)/2 \rfloor$ erreurs. Par exemples pour les matrices (6.3), (6.5) et (6.8), la distance $d_H^{\min}$ est respectivement de 2, 4 et 4, soit respectivement un nombre de corrections d'erreurs de 0, 1 et 1. Si le nombre d'erreurs qui peuvent être corrigées n'est pas directement lié au nombre k de lignes de la matrice Ψ, il en dépend fortement. On remarque en particulier que la décomposition *un-contre-tous* ne permet de corriger aucune erreur car $d_H^{\min} = 2$ quel que soit le nombre de classes. A partir de ces remarques, le principe de décodage basé sur les codes correcteurs correspond à choisir pour un exemple $\mathbf{x}$, la classe ω_i la plus proche par rapport à la distance de Hamming correspondante :

$$\omega(\mathbf{x}) = \arg\min_{\omega_i} d_H(D_C(\mathbf{x}),C_i(\Psi)) \tag{6.23}$$

Lorsque la matrice Ψ contient des 0, la distance de Hamming entre deux codes peut être étendue de la façon suivante:

$$d_{H_0}(C_1,C_2) = \sum_{j=1}^{k} \epsilon_{H_0}(C_{1,j},C_{2,j}) \tag{6.24}$$

avec $C_{i,j}$ la $j^{\text{ème}}$ valeur du code C_i et $\epsilon_{H_0}(u,v) = \begin{cases} 0 & \text{si } u = 0 \text{ ou } v = 0 \text{ ou } u = v \\ 1 & \text{si } u \neq v \end{cases}$

Pour cette formulation, les matrices (6.3) et (6.7) ont pour distance $d_{H_0}^{\min}$ respectivement 1 et 3.

Allwein et al [ALLWEI00] étendent la méthode ECOC en proposant une distance qui ne soit pas seulement discrète comme celle de Hamming, mais qui prend en compte la magnitude de l'erreur faite par chaque fonction de décision binaire par rapport à un code donné. Cette prise en compte est réalisée à partir de l'utilisation d'une fonction de perte L :

$$d_L(D,C) = \sum_{j=1}^{k} \epsilon_L(D(\boldsymbol{\psi}_j,\mathbf{x}),C_j) \tag{6.25}$$

avec C_j la $j^{\text{ème}}$ valeur du code C et $\epsilon_L(u,v) = \begin{cases} 0 & \text{si } v = 0 \\ L(u,v) & \text{sinon} \end{cases}$

Dans le cas des SVM, la fonction de perte L prend en compte l'éloignement à la marge

(cette fonction est utilisée pour les expérimentations de la section 6.2.1.2)[5]: $L(u,v) = \max(1-uv,0)$.

Allwein et al montrent également qu'avec cette formulation les schémas de décomposition *un-contre-un* et *un-contre-tous* ont des possibilités de correction d'erreurs accrues, alors qu'elles étaient nulles avec l'utilisation de la distance de Hamming.

Autres principes

Abe [ABE05] propose d'utiliser des règles floues (*cf.* Chapitre 3 de [ABE05]) pour lever les problèmes d'inconsistance dans la sélection d'une classe unique. Il montre que l'utilisation de règles floues est fortement lié au principe de décodage ECOC proposé par Allwein et al [ALLWEI00] et que, expérimentalement, les résultats sont très proches entre ces deux principes de décodage.

Quost et al [QUOST04] proposent de réaliser le décodage en utilisant la théorie de l'évidence. Ils proposent deux formulations adaptées respectivement aux décompositions *un-contre-tous* [QUOST06] et *un-contre-un* [QUOST05].

Une autre possibilité est de considérer les sorties de l'ensemble des classificateurs binaires comme des attributs. Une nouvelle base d'apprentissage est induite à partir des valeurs de ces attributs pour l'ensemble des exemples de la base initiale. L'idée est alors d'utiliser un algorithme d'apprentissage directement multi-classe avec cette nouvelle base pour produire la fonction de décodage [MAYORA99, SAVICK03, LEZORA05].

6.1.2.2 Décodage en cascade

Il existe un grand nombre de schémas de combinaison basés sur un décodage en cascade [PLATT99C, VURAL04, LEI05] et elles dépendent en partie de la méthode de décomposition utilisée. La majorité des procédures de décodage en cascade réalisent une élimination itérative des classes pour qu'au final il n'en reste plus qu'une. Le synopsis général d'une procédure d'élimination est le suivant:

1. Choisir une des fonctions de décision binaire.
2. Déterminer la classe positive ou négative prédite.
3. Eliminer la (les) classe(s) ω ne pouvant être retenue(s) à partir du vecteur $\boldsymbol{\psi}$.
4. Choisir une autre fonction de décision binaire susceptible de réduire davantage le nombre de classes éligibles,
5. Répéter les étapes 2 à 5 jusqu'à ce qu'il ne reste plus qu'une classe.

Cette procédure correspond à la définition implicite d'un arbre de décision. La figure 6.2 illustre celui lié à la production d'un DDAG [PLATT99C] et la figure 6.3 celui lié à la méthode *moitié contre moitié* [LEI05].

Une autre possibilité pour réaliser un décodage en cascade est d'utiliser un sous-ensemble des fonctions de décision binaires produites par la méthode de décomposition

5. Dans [ALLWEI00], d'autres fonctions de perte sont proposées suivant le classificateur binaire utilisé et la signification de la valeur de $D(\boldsymbol{\psi}_j,\mathbf{x})$.

pour sélectionner un ensemble réduit de classes de confiance. Ce processus est répété jusqu'à sélection d'une classe unique. A chaque nouvelle itération, seuls les classificateurs exploitant des classes du sous-ensemble de confiance peuvent être sélectionnés pour réaliser la définition du prochain sous-ensemble de confiance. La figure 6.4 illustre ce principe de décodage.

Dans [ABE05], Abe propose par exemple d'utiliser d'abord les classificateurs les plus performantes en généralisation pour sélectionner cette ensemble de classes de confiance, puis de procéder de manière récursive avec les classificateurs de moins en moins performants.

Dans [LEZORA04], Lezoray et al proposent de définir un graphe d'induction qui prend en compte les capacités de généralisation de chaque classificateur binaire pour déterminer la manière de le parcourir. Ils montrent en particulier que la classe sélectionnée est dé-

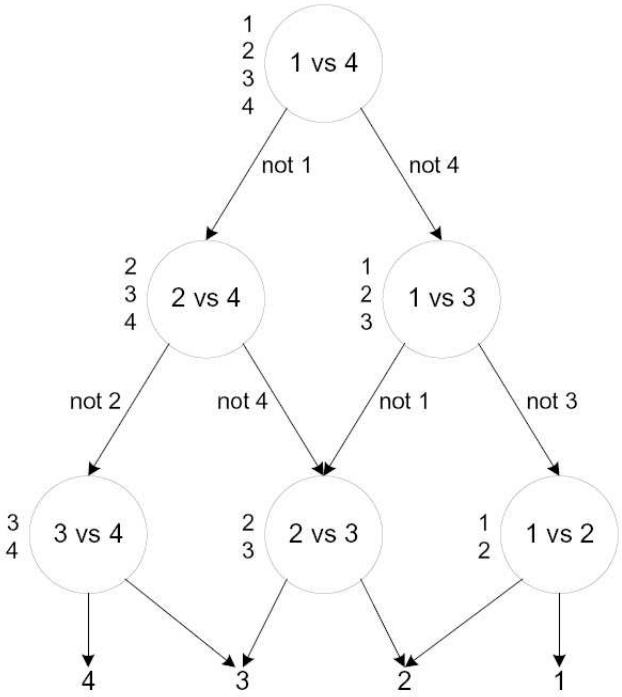


FIG. 6.2 – *Le DAG de décision relatif à un problème à 4 classes [*PLATT99C*]. A chaque étage du DAG, une classe est éliminée.*

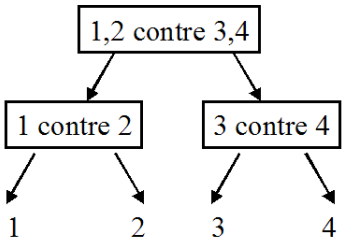


FIG. 6.3 – *Arbre de décision relatif au décodage* moitié contre moitié. *A chaque étape, la moitié des classes sont éliminées.*

pendante du nœud sélectionné comme point de départ dans le graphe. D'autres études montrent que les capacités de généralisation d'un arbre ou graphe d'induction dépendent de l'ordre dans lequel les fonctions de décision binaires produites par la méthode de décomposition sont utilisées. La construction de ce graphe ou son exploitation dynamique correspond à un processus d'apprentissage lié à la répartition des données entre les différents sous-ensembles de classes [ABE05, LEZORA05]. Certaines approches cherchent à optimiser la structure d'un DAG appellé alors ADAG (Adaptive DAG) [PHETKA03, TAHAHA03].

6.1.2.3 Décodage hybride

Des principes de décodage à partir de méthodes de décomposition binaire qui ne sont ni exclusivement globaux, ni exclusivement en cascade ont été également définis. Nous présentons en section 6.2.1 un de ceux-ci. Sans chercher à être exhaustif, nous nous limiterons à citer quelques schémas de combinaison basés sur un principe hybride de décodage [CHOU03, FRANK04]. Ils sont basés essentiellement sur des méthodes d'ensemble de classificateurs [FÜRNK02].

6.1.3 Importance de la sélection de modèle

Dans le cadre de l'utilisation de SVM dans un schéma de combinaison de classificateurs binaires se pose le problème du choix des valeurs des hyper-paramètres. Dans le cadre plus général d'utilisation de méthodes d'apprentissage dédiées à des problèmes binaires, elles nécessitent généralement la sélection d'un modèle (*cf.* chapitre 5 par exemple) qui a un impact important sur les capacités de généralisation de la fonction de décision produite. Se pose alors le problème du choix d'un ensemble des modèles relatifs à chaque

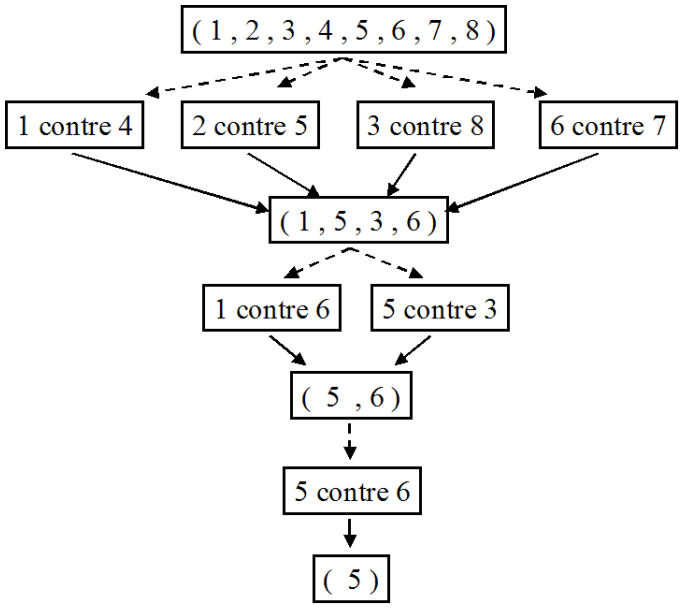


FIG. 6.4 – *Sélection itérative d'un sous-ensemble de classes de confiances.*

problème binaire produit par la méthode de décomposition. Il est possible de distinguer deux cas de figures relatifs à la sélection de modèle pour un schéma de combinaison binaire.

1. Le modèle sélectionné est commun (les hyper-paramètres des SVM par exemple) à l'ensemble des fonctions de décision intervenant dans un schéma de combinaison. La sélection d'un modèle efficace est alors réalisée à partir de l'estimation du pouvoir de généralisation du schéma global de combinaison sans tenir compte des performances individuelles de chaque fonction de décision binaire [WU04, VURAL04, ABE05].
2. La sélection d'un modèle optimal est réalisée pour chaque fonction de décision binaire. Il y a alors autant de modèles à sélectionner que de fonctions de décision produites par la méthode de décomposition. Ces fonctions de décision binaires sont utilisées telles quelles par le principe de décodage, ce qui correspond à faire l'hypothèse qu'optimiser individuellement chaque fonction de décision binaire produit également l'optimisation de la combinaison de ces fonctions de décision [PÉREZ-02, RIFKIN04].

Remarques : La première méthode de simplification des fonctions de décision binaires présentée en section 5.3.1 correspond au premier cas de sélection du modèle et la seconde méthode de simplification présentée en section 5.3.2 correspond au deuxième cas. Nous verrons en section 6.2.2 qu'il peut être judicieux d'utiliser un troisième principe de sélection qui utilise un modèle distinct pour chaque fonction de décision binaire à produire, mais sélectionne l'ensemble de ces modèles de façon à ce que le pouvoir de généralisation du schéma de combinaison global soit optimisé.

6.2 Nouvelles approches

6.2.1 Un schéma hybride

Les schémas de combinaison *un-contre-un* et *un-contre-tous* ont été souvent comparés entre eux, ainsi qu'avec d'autres schémas de combinaison, en particulier lorsque les classificateurs binaires sont des SVM [RIFKIN04, DUAN05, ABE05]. Suivant les études réalisées, le premier est supérieur au second [DUAN05], ou inversement [RIFKIN04], ou bien ils ont des capacités de généralisation similaires [ABE05]. Dans [RIFKIN04], il est précisé que la sélection des valeurs des hyper-paramètres pour chaque SVM joue un rôle essentiel et qu'elle influence la comparaison de deux schémas de combinaison. Lorsqu'ils sont comparés avec d'autres schémas de combinaisons [RIFKIN04], s'ils ne sont pas les meilleurs, ils en sont très proches. Il apparaît à la lecture de ces différents résultats qu'un schéma de combinaison sera plus ou moins performant suivant la nature du problème binaire à traiter. C'est pour cette raison que certains schémas de combinaison multi-classes cherchent à apprendre la structure du schéma de décomposition. A partir de l'ensemble des résultats de ces différents travaux, les schémas *un-contre-un* et *un-contre-tous* nous ont semblé globalement performants et de complexité raisonnable. Nous avons donc cherché comment les utiliser conjoitement pour produire un schéma de combinaison de fonc-

tions binaires plus performant en généralisation, le but étant de tirer profit des avantages individuels de ces deux schémas et de réduire leurs désavantages.

Nous avons alors cherché à déterminer les différents facteurs qui peuvent désavantager un de ces deux schémas par rapport à l'autre. Le nombre de classes est un facteur important pour représenter la difficulté du décodage. Cette difficulté de décodage augmente moins rapidement avec la décomposition *un-contre-tous* qu'avec la décomposition *un-contre-un*, car le nombre de fonctions binaires augmente linéairement dans le premier cas et quadratiquement dans le second cas. Nous avons également évoqué en section 6.1.2.1 le problème du bruit de décodage lié à la structure de la matrice de décomposition *un-contre-un*, l'augmentation rapide du nombre de classificateurs binaires pour ce type de schéma amplifie ce phénomène. L'ensemble de ces remarques illustre les désavantages liés à la décomposition *un-contre-un* et montrent qu'ils sont plus marqués avec des problèmes comportant un nombre important de classes.
Un autre facteur à prendre en compte est le fait que la ressemblance peut être plus ou moins importante entre certaines classes d'un problème d'apprentissage qu'entre d'autres classes du même problème. Prenons pour exemple la reconnaissance de lettres manuscrites et limitons nous aux 4 lettres a,o,m,n. Les couples de lettres (a,o) et (m,n) sont plus proches que les couples (a,m), (a,n), (o,n) et (o,m). Laissons temporairement de côté les désavantages précédemment évoqués de la décomposition *un-contre-un* pour nous concentrer sur un effet négatif lié à la décomposition *un-contre-tous* lorsqu'au moins deux classes ont une forte ressemblance (ou autrement dit, il existe des classes qui sont intrinsèquement plus difficiles à discriminer que d'autres classes dans un même problème). Prenons pour support la figure 6.5 pour illustrer cet effet négatif. Dans cette figure, les deux classes fortement semblables sont les classes 2 et 3. La classe 1 représente une classe (ou le regroupement d'ensemble d'autres classes) qui a une différence plus marquée avec les deux classes[6] 2 et 3. Les trois sous-problèmes binaires de la décomposition *un-contre-un* sont moins difficiles (2 contre 3 est un peu plus difficile) que ceux de la décomposition *un-contre-tous* car 2 des 3 sous-problèmes binaires sont plus difficiles (2 contre 1,3 et 3 contre 1,2). Ceci est principalement dû à la nécessité de réaliser la discrimination d'une classe, désignée comme positive, avec le regroupement des autres classes, désignées comme classe négative, qui contient des exemples très proches de ceux de la classe positive. De plus, la dispersion de la classe négative peut être importante due au regroupement d'exemples de nature très différente. Les deux fonctions de décision binaires concernées dans notre exemple sont moins performantes que celle réalisant la discrimination de la classe 1 contre les classes 2 et 3. Globalement, toujours pour cet exemple, le schéma *un-contre-tous* est plus performant pour la reconnaissance des exemples qui sont de la classe 1 que pour ceux qui sont de classe 2 ou 3 car il peut plus facilement les confondre. Il semble logique que cet effet négatif soit encore plus important lorsque le nombre de classes augmente. L'ensemble de ces remarques illustre les désavantages liées à la décomposition *un-contre-tous* avec certains types de problèmes multi-classes et elles montrent qu'ils sont plus marqués lorsque ces problèmes comportent un nombre important de classes.

A partir de ces remarques, on arrive à la conclusion que ces deux schémas de combinaison ont chacun des désavantages et que suivant la nature du problème multi-classe

6. Pour le problème de reconnaissance de lettres manuscrites, la correspondance suivante peut être réalisée: $2 \equiv$ a, $3 \equiv$ o, $1 \equiv$ les autres lettres.

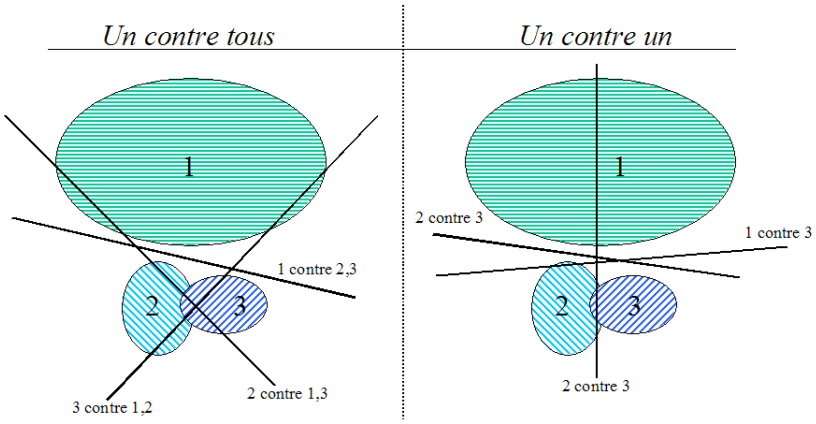


FIG. 6.5 – *Exemples de frontières de décision suivant la décomposition* un-contre-tous *ou* un-contre-un *lorsque deux classes ont une ressemblance marquée.*

considéré, en particulier lorsque le nombre de classes est important, il faudra choisir celui dont les désavantages sont les moins marqués.

Pour réduire les inconvénients relatifs aux deux schémas *un-contre-un* et *un-contre-tous*, nous avons défini un schéma hybride qui exploite les deux décompositions associées à ces deux schémas, mais qui change le principe de décodage. Avant de donner les détails de notre schéma de combinaison, nous donnons le principe du décodage hybride de notre méthode et illustrons à partir de l'exemple de la figure 6.5 son fonctionnement. Les remarques précédentes montrent que le schéma *un-contre-tous* est efficace pour sélectionner des classes candidates pertinentes, mais une discrimination fine est rendue difficile car chaque classificateur binaire n'est pas assez spécifique. Une des idées de notre principe de décodage est d'utiliser le schéma *un-contre-tous* pour sélectionner un ensemble de classes candidates parmi les plus probables au lieu de chercher à n'en sélectionner qu'une. Pour l'exemple de la figure 6.5, suivant l'appartenance des exemples à la classe 1, 2 ou 3 les classes candidates seront, dans la majorité des cas, respectivement (1), (2,3) ou (3,2). Si une seule classe candidate est sélectionnée, le décodage est terminé. Dans le cas contraire, la sélection finale est réalisée à partir d'un schéma *un-contre-un* simplifié. Pour l'exemple de la figure 6.5, si l'ensemble des classes candidates est (2,3), le schéma *un-contre-un* est réduit à sa plus simple expression et correspond à utiliser le classificateur binaire *2 contre 3*. Plus généralement, la deuxième étape de notre principe de décodage est d'utiliser des classificateurs binaires spécialisés dans la discrimination des classes fortement plausibles pour réaliser la sélection finale d'une classe unique à partir d'un schéma *un-contre-un* dans lequel chaque classificateur binaire est pertinent. On remarque que ce principe de décodage hybride en deux étapes réduit les inconvénients liés aux deux schémas *un-contre-tous* et *un-contre-un*.

6.2.1.1 Synopsis

Le synopsis de notre schéma hybride de combinaison de fonction de décision binaire est représenté par la figure 6.6. Il s'agit d'un décodage en cascade en deux étapes. La première étape sélectionne à partir d'un décodage global basé sur une décomposition *un-contre-tous* un ensemble de classes candidates. La sélection des classes candidates est

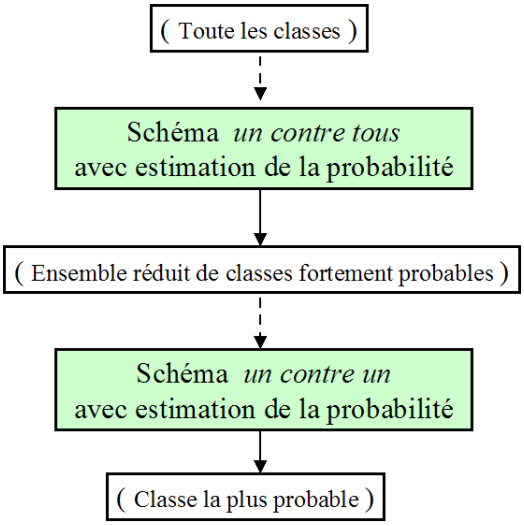


FIG. 6.6 – *Synopsis du schéma hybride de combinaison de classificateurs binaires.*

réalisée à partir de l'estimation de la probabilité de chaque classe par l'expression (6.12). Nous proposerons deux critères par la suite pour réaliser cette sélection. Notons ω_1, l'ensemble réduit de classes fortement probables sélectionné par cette première étape et Ψ_1 la matrice correspondant à la décomposition *un-contre-un* pour le sous-ensemble de classes ω_1. La seconde étape sélectionne, à partir d'un décodage global basé sur la matrice de décomposition Ψ_1, la classe qui a la plus haute probabilité estimée. La probabilité de chaque classe est estimée à partir d'une des quatre principes de décodage décrits à la section 6.1.2.1. Nous les désignerons par d_{vote}, d_{Price}, d_{Hastie} et d_{Wu} respectivement en fonction de la réalisation ou de la détermination des équations (6.18), (6.20), (6.21) ou (6.22).

Pour la sélection des k_c classes fortement probables à la fin de la première étape, nous avons défini deux critères.

Critère c_{3c}

Le premier critère, désigné par c_{3c}, correspond à la synthèse d'un ensemble d'observations expérimentales. Nous avons mesuré, pour le schéma *un-contre-tous*, le rang correspondant à la bonne prédiction de la classe d'un exemple lorsque les labels de classe sont triés par ordre décroissant de la probabilité estimée (*cf.* expression (6.12)). A partir de cette notion de rang d'un exemple relativement à la prédiction de sa classe, nous avons déterminé le pourcentage d'exemples pour chacun de ces rangs. La figure 6.7 illustre un des résultats obtenus. Nous avons constaté qu'il existe une forte probabilité pour que le label correct d'un exemple soit situé entre le rang 1 et le rang 3. A partir de cette remarque, la sélection de l'ensemble des classes fortement probables s'effectue en conservant les trois classes qui ont les plus fortes probabilités estimées ($k_c = 3$). Ce critère à également l'avantage de forcer l'utilisation d'un schéma *un-contre-un* de taille réduite.

Critère $c_{\delta_{\max}}$

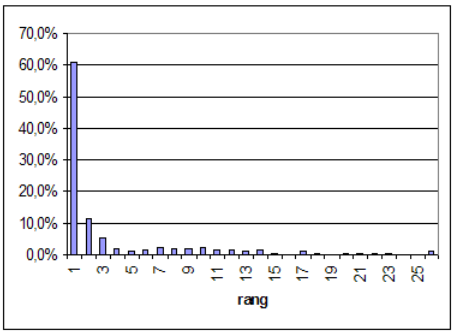


FIG. 6.7 – *Pourcentage d'exemples correctement classés pour chaque rang avec un schéma un-contre-tous. La base de test est* **Letter** *et la sélection des hyper-paramètres des SVM utilisés dans ce schéma est identique à celle de la section 6.2.1.2.*

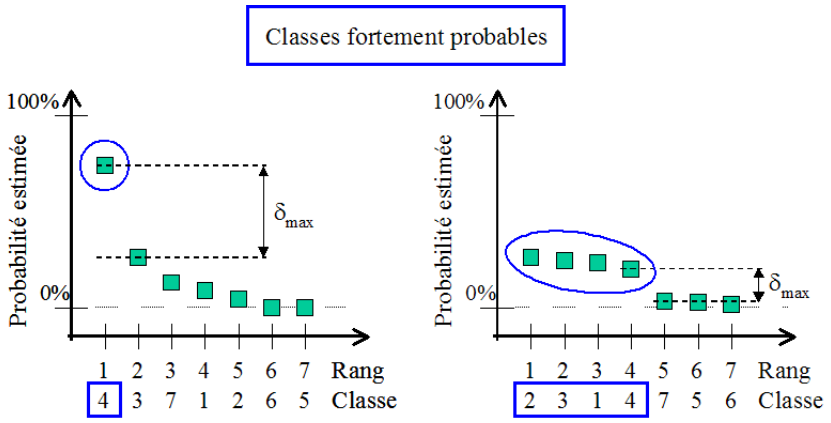


FIG. 6.8 – *Sélection dynamique du nombre de classes fortement probables pour chaque exemple. Le seuil* $\delta_{\max}$ *définit la séparation des classes en deux ensembles par rapport à leurs rangs respectifs.*

Le deuxième critère, désigné par $c_{\delta_{\max}}$, réalise également la sélection des classes fortement probables par rapport aux rangs qui leur sont associés, mais ne fixe pas à l'avance la valeur de k_c. Il utilise les probabilités estimées $p(\omega_i|x)$ pour déterminer le nombre de classes à sélectionner parmi celles qui ont les plus fortes valeurs $p(\omega_i|x)$. Soit σ_i la permutation telle que les valeurs $p(\omega_{\sigma_i}|x)$ soient décroissantes en fonction de i et $\delta_i = p(\omega_{\sigma_i}|x) - p(\omega_{\sigma_{i+1}}|x)$ l'écart entre deux valeurs consécutives de ces estimations. Notre second critère utilise la valeur maximum des δ_i pour choisir le nombre k_c de classes à sélectionner :

$$k_c = \arg \max_{1 \leq i \leq n_c} \delta_i \tag{6.26}$$

avec par convention $p(\omega_{\sigma_i}|x) = 0$ si $i > n_c$. La figure 6.8 illustre ce principe de sélection avec $k_c = 1$ pour le premier exemple (à gauche) et $k_c = 4$ pour le second exemple (à droite). Dans le cas où $k_c = 1$, la deuxième étape de notre schéma hybride n'est pas réalisée.

Pour un grand nombre d'applications les valeurs des probabilités estimées relatives à chaque classe sont d'autant plus importantes que la sélection de la classe est plus probable.

Nous avons vu en section 6.1.2.1 différentes formulations pour l'estimation des $p(\omega_{\sigma_i}|\mathbf{x})$ avec un schéma de décodage *un-contre-un* et *un-contre-tous*. Notre méthode hybride, bien qu'exploitant ces formulations, pose problème pour une estimation globale de ces probabilités à cause du processus en deux étapes. Nous avons défini une méthode simple permettant de combiner les probabilités estimées pour ces deux étapes. Notons $p_1(\omega_{\sigma_i}|\mathbf{x})$, les n_c probabilités estimées avec un schéma *un-contre-tous* par la première étape de notre méthode hybride ($\sum p_1\left(\omega_{\sigma_i}|\mathbf{x}\right) = 1$ avec $1 \leq i \leq n_c$) et $p_2(\omega_{\sigma_i}|\mathbf{x})$ les k_c probabilités estimées avec un schéma *un-contre-un* ($\sum p_2\left(\omega_{\sigma_i}|\mathbf{x}\right) = 1$ avec $1 \leq i \leq k_c$) pour les k_c premières classes de la permutation σ (σ conserve la même signification que précédement). Nous considérons que les probabilités estimées par la deuxième étape pour les k_c classes les plus probables sont plus pertinentes que leurs estimations avec la première étape. Les valeurs des $p(\omega_{\sigma_i}|\mathbf{x})$ pour notre schéma hybride sont alors définie par:

$$p\left(\omega_{\sigma_i}|\mathbf{x}\right) = \begin{cases} p_2\left(\omega_{\sigma_i}|\mathbf{x}\right) \cdot \sum_{j=1}^{k_c} p_1\left(\omega_{\sigma_j}|\mathbf{x}\right) & \text{si } i \leq \mathrm{k_c} \\ p_1\left(\omega_{\sigma_i}|\mathbf{x}\right) & \text{sinon} \end{cases} \tag{6.27}$$

L'expression $\sum_{j=1}^{k_c} p_1\left(\omega_{\sigma_j}|\mathbf{x}\right)$ dans (6.27) permet de vérifier la contrainte $\sum_{i=1}^{n_c} p\left(\omega_{\sigma_i}|\mathbf{x}\right) = 1$.

6.2.1.2 Expérimentations

Ces expérimentations ont pour but de comparer notre schéma hybride à plusieurs schémas classiques de combinaison de fonctions de décision binaires. Les schémas classiques utilisés sont les suivants: *un-contre-tous*, *un-contre-un*, IPCCC et ECOC. Pour les décompositions *un-contre-tous*, *un-contre-un* et IPCCC, les principes de décodage décrits en section 6.1.2.1 ont été utilisés pour l'estimation des probabilités (pour la décomposition *un-contre-un* les 4 procédures relatives à d_{vote}, d_{Price}, d_{Hastie} et d_{Wu} sont utilisées pour l'estimation des probabilités). Pour la création de la matrice de décomposition relative au schéma ECOC, nous avons utilisé la méthode RHC (*Randomized Hill Climbing*) décrite dans [DIETTE95].

Procédure de sélection de modèle

Pour l'ensemble des problèmes binaires impliqués dans un ou plusieurs des schémas de combinaison précédement énumérés, une recherche de modèle (C,γ) est réalisée pour le choix des hyper-paramètres des SVM. Toutes les fonctions de décision binaires $D(\boldsymbol{\psi},\cdot)$ construites par les SVM s'appuient sur un noyau gaussien avec γ la largeur de bande ($K(\mathbf{u},\mathbf{v}) = \exp\left(\gamma \left\|\mathbf{u}-\mathbf{v}\right\|^2\right)$) . La méthode de Platt [PLATT99B] est utilisée pour déterminer les estimations de probabilités relatives à chaque classificateur binaire utilisé dans un schéma de décomposition. Nous avons utilisé le même genre de procédure que dans [WU04] pour réaliser la sélection de modèle. Cette procédure consiste à réaliser une validation croisée en 5 parties pour la sélection d'un couple (C,γ) parmi l'ensemble de couples $\{2^{-5},2^{-3},\ldots,2^{15}\}\times\{2^{-5},2^{-3},\ldots,2^{15}\}$. Pour chaque couple ($C$,$\gamma$) testé, les valeurs[7] de a_1 et de a_2 sont déterminées avec la méthode de [LIN03A] (*cf.* section 4.1.5.4) pour chacune des cinq étapes de la procédure de validation croisée (voir [WU04] pour plus de détails).

7. Relatives à : $D(\boldsymbol{\psi},\mathbf{x}) = \frac{1}{1+\exp(a_1 \cdot f(\boldsymbol{\psi},\mathbf{x})+a_2)} \approx p(\omega = +1|\boldsymbol{\psi},\mathbf{x})$.

Base	t	*un-contre-un*	*un-contre-tous*	IPCCC	ECOC
Satimage	1	3,6±5,4%	13,7±11,1%	8,2±8,1%	10,8±8,7%
Satimage	2	3,1±4,5%	11,9±10,2%	7,3±6,9%	9,4±5,2%
USPS	1	0,9±1,5%	10,3±6,8%	7,2±8,1%	8,9±7,3%
USPS	2	0,7±0,9%	18,1±5,1%	5,5±6,2%	7,1±5,8%
Letter	1	6,0±7,2%	29,6±11,9%	19,9±16,1%	22,7±12,8%
Letter	2	2,4±2,6%	20,3±8,6%	13,6±12,4%	13,9±9,2%

TAB. 6.1 – *Valeur moyenne de $e_v^t(\Psi)$ pour l'ensemble des fonctions de décision binaire utilisé par une décomposition donnée Ψ.*

Les fonctions binaires impliquées dans les schémas de décomposition

Les bases d'apprentissage et de validation utilisées sont issues des trois bases **Satimage**, **USPS** et **Letter** (*cf.* annexe A). Elles ont respectivement 6, 10 et 16 classes. Pour des raisons pratiques de temps d'entraînement, celles utilisées pour ce chapitre sont de tailles différentes. Nous avons utilisé les mêmes bases que Wu et al [WU04][8]. Deux types d'échantillonnage ont été réalisés. Les bases d'apprentissage Z^A et de validation Z^V produites ont respectivement suivant ces deux cas : 1) 300 et 500 exemples et 2) 500 et 800 exemples. Pour chacun de ces types d'échantillonnage et pour chaque base d'origine, 20 bases différentes sont produites. Nous noterons B_i^t l'échantillon numéro i de type t de la base B (B étant une des trois bases de référence **Satimage**, **USPS**, **Letter**). Pour chacune des bases B_i^t produites par la procédure d'échantillonnage, les taux d'erreur $e_v(B_i^t,\psi)$ (mesurés à partir de l'échantillon de validation), des fonctions de décision binaires $D(\psi,\cdot)$ produites sont mesurés sur les bases de validation correspondantes. La valeur moyenne $e_v^t(\psi) = \frac{1}{20}\sum e_v(B_i^t,\psi)$ pour un problème binaire ψ donné est ensuite déterminée. Pour chaque schéma de décomposition Ψ exploité dans cette section, la valeur moyenne (et l'écart type) $e_v^t(\Psi)$ est déterminée à partir des valeurs $e_v^t(\psi)$ des fonctions binaires $D(\psi,\cdot)$ intervenant pour ce schéma de décomposition. Le tableau 6.1 donne les résultats obtenus pour les schémas de décomposition *un-contre-un*, *un-contre-tous*, IPCCC et ECOC. Ces résultats illustrent que la décomposition *un-contre-un* produit l'ensemble de problèmes binaires les plus simples à résoudre et la décomposition *un-contre-tous* ceux les plus difficiles; les décompositions IPCCC et ECOC se situant entre les deux. Ces résultats sont logiques et corroborent les remarques émises en section 6.2.1.1. L'augmentation de la taille de la base d'apprentissage permet de réduire les écarts entre les capacités de généralisation de ces ensembles, mais globalement le classement des performances moyennes des fonctions binaires relatif à ces différentes décompositions reste le même.

L'ensemble des fonctions des décision binaires créées à cette étape est ensuite utilisées pour évaluer les performances de plusieurs schémas de combinaison classiques ainsi que celles de notre schéma hybride.

Performances des schémas de combinaison classiques

Nous avons mesuré le taux d'erreur moyen e_v^t pour différents principes de décodage

8. Elles sont accessibles à l'adresse suivante: http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/svmprob/svmprob-1.0.tgz.

	(%)	*un-contre-un*				*u-c-t*	IPCCC	ECOC
B	t	d_{vote}	d_{Price}	d_{Hastie}	d_{Wu}			
Satimage	1	14,7±1,8	14,7±1,8	14,7±1,7	14,7±1,8	14,6±1,7	14,3±1,8	14,5±1,8
Satimage	2	12,1±0,9	11,8±0,9	11,8±0,8	11,7±0,8	11,5±0,8	11,3±1,0	11,3±1,2
USPS	1	13,2±1,3	12,8±1,2	13,0±1,1	12,6±1,1	11,9±1,3	10,1±1,0	10,5±1,2
USPS	2	9,2±0,8	8,9±0,9	9,2±1,1	9,0±0,9	8,8±1,3	7,4±1,0	7,8±1,3
Letter	1	40,5±3,2	40,5±3,0	43,3±2,8	41,2±2,9	41,9±3,3	33,4±3,1	37,4±3,4
Letter	2	21,8±1,6	21,4±1,7	24,9±1,8	22,8±1,6	22,1±1,3	19,4±1,3	19,9±1,6

TAB. 6.2 – *Valeur moyenne de e_v^t pour différents principes de décodage (*u-c-t *est l'abréviation de* un-contre-tous*). Toutes les valeurs de ce tableau sont des pourcentages.*

global présentés en section 6.1.2.1. Cette mesure est réalisée à partir des 20 échantillons (relatif à chaque type t donné) de validation pour chacune des 3 bases utilisées lors de la phase de création des fonctions de décision binaires. Ces mesures sont réalisées pour les principes de décodage exploitant les décompositions *un-contre-un* (avec les 4 variantes d_{vote}, d_{Price}, d_{Hastie} et d_{Wu}), *un-contre-tous* (*cf.* relation (6.12)), IPCCC (*cf.* relation (6.15)) et ECOC (*cf.* relation (6.23)). Le tableau 6.2 donne les résultats obtenus pour ces différents principes de décodage.

Ces résultats illustrent que le nombre plus élevé et la spécialisation à seulement deux classes des fonctions de décision binaire fait que le décodage de l'ensemble des informations relatives à la décomposition *un-contre-un* n'est pas forcément la meilleure, bien que les fonctions de décision binaires (*cf.* tableau 6.1) utilisées avec un schéma *un-contre-un* soient plus performantes en généralisation que celles utilisés avec schéma *un-contre-tous*. Globalement ces deux schémas de combinaison ont des performances similaires. Les schémas IPCCC et ECOC sont plus performants que les deux précédents, mais ils nécessitent l'exploitation d'un nombre plus important de sous-problèmes binaires. Les temps d'entraînement et de décision sont donc beaucoup plus importants. De plus ces temps augmente avec le nombre de classes.

En ce qui concerne les 4 types de décodage relatifs à la décomposition *un-contre-un*, les différences de performances sont assez faibles. Le décodage d_{Hastie} est celui qui est le moins régulier. Pour les trois autres, ils ont des performances fortement similaires. Par contre, les décodages d_{vote} et d_{Price} ont l'avantage d'être plus rapides car la détermination des probabilités $d(\omega_i)$ liées à ces deux décodages est directe, alors que pour le décodage d_{WU} (et d_{Hastie}) un problème d'optimisation doit être résolu pour chaque décision relative à un nouvel exemple[9].

Performances de notre schéma hybride

Nous avons ensuite mesuré le taux d'erreur moyen e_v^t (et l'écart type) pour les mêmes

9. Wu et al n'arrivent pas aux mêmes classement entre ces décodages dans leur article [WU04], car ils utilisent les mêmes paramètres (C,γ) pour toutes les fonctions binaires d'un même schéma de décomposition. Par contre, les paramètres (C,γ) sélectionnés sont propres à chaque schéma de décomposition. Il peut donc y avoir pour un même sous-problème binaire des fonctions de décision différentes suivant le schéma dans lequel ce sous-problème intervient. Ces différences méthodologiques peuvent expliciter ces différences de classement.

	(%)	décodage hybride avec critère c_{3c}				$\Delta\bar{e}$
B	t	d_{vote}	d_{Price}	d_{Hastie}	d_{Wu}	
Satimage	1	14,5±1,6	14,5±1,5	14,4±1,8	14,5±1,7	-0,22
Satimage	2	12,1±0,9	11,9±0,9	11,7±0,8	11,7±0,7	-0,00
USPS	1	12,6±1,3	12,4±1,2	12,5±1,2	12,4±1,2	-0,42
USPS	2	8,9±0,8	8,9±0,9	8,7±0,8	8,8±0,7	-0,25
Letter	1	40,5±3,9	40,5±3,8	40,4±3,8	40,3±3,2	-0,95
Letter	2	21,2±1,3	20,8±1,2	20,9±1,3	20,8±1,2	-1,80

TAB. 6.3 – *Valeur moyenne de e_v^t pour le décodage hybride avec le critère c_{3c}. La colonne $\Delta\bar{e}$ correspond à l'écart moyen pour e_v^t entre notre schéma hybride et le décodage classique* un-contre-un *du tableau 6.2.*

	(%)	décodage hybride avec critère $c_{\delta_{\max}}$				$\Delta\bar{e}$	k_c
B	t	d_{vote}	d_{Price}	d_{Hastie}	d_{Wu}		
Satimage	1	14,5±1,9	14,5±1,9	14,5±2,0	14,5±1,7	-0,20	1,7±0,9
Satimage	2	12,0±0,9	11,6±0,9	11,8±0,9	11,6±0,9	-0,10	1,5±0,7
USPS	1	12,2±1,2	12,2±1,1	12,2±1,1	11,9±0,9	-0,77	2,3±1,4
USPS	2	8,6±0,7	8,6±0,7	8,6±0,7	8,4±0,8	-0,52	1,9±1,1
Letter	1	38,5±3,2	38,4±3,1	38,5±3,1	38,2±2,9	-2,97	5,1±3,2
Letter	2	20,6±1,3	20,5±1,3	20,7±1,3	20,4±1,1	-2,17	4,2±2,3

TAB. 6.4 – *Valeur moyenne de e_v^t pour le décodage hybride avec le critère $c_{\delta_{\max}}$. La colonne $\Delta\bar{e}$ correspond à l'écart moyen pour e_v^t entre notre schéma hybride et le décodage classique* un-contre-un *du tableau 6.2. k_c représente le nombre moyen de classes sélectionnées avec ce critère.*

échantillons que précédemment avec notre schéma hybride. Les mêmes fonctions de décision binaires que celles exploitées par les principes de décodage *un-contre-un* et *un-contre-tous* ont été utilisées par la procédure de décodage de notre schéma hybride. Les deux critères c_{3c} et $c_{\delta_{\max}}$ pour la sélection d'un sous-ensemble de classes de confiance à l'étape 1 ont été testés. Nous avons utilisé, pour chacun de ces deux critères, les 4 principes de décodage relatifs à une décomposition *un-contre-un* lors de la deuxième étape de notre schéma. Les tableaux 6.3 et 6.4 donnent respectivement les résultats obtenus avec les critères c_{3c} et $c_{\delta_{\max}}$. La colonne $\Delta\bar{e}$ dans ces deux tableaux permet d'avoir une estimation de l'accroissement des performances en généralisation entre notre schéma de combinaison hybride et un schéma de combinaison classique *un-contre-un* en mesurant l'écart moyen sur les taux d'erreurs e_v^t relatifs aux 4 décodages d_{vote}, d_{Price}, d_{Hastie}, d_{Wu} avec ces deux schémas.

Les résultats issus de ces deux tableaux illustrent que l'accroissement des performances est d'autant plus significatif que le nombre de classes dans le problème de classification augmente. Le schéma hybride est plus performant que les deux schémas *un-contre-un* et *un-contre-tous* avec la base **Letter**. Pour les deux autres bases, le schéma *un-contre-tous* n'est pas plus performant, ce qui semble indiquer que les bénéfices de notre schéma hybride ne sont perceptibles que lorsque le nombre de classes est conséquent. Le critère $c_{\delta_{\max}}$ est globalement plus performant que le critère c_{3c} pour la sélection des classes fortement probables intervenant à l'étape 2 de notre schéma hybride. La supériorité de la sélection dynamique du nombre de classes par rapport à un nombre fixé de classes est

vérifiée expérimentalement. Cette supériorité est d'autant plus marquée que le nombre de classes dans le problème augmente. Le nombre de classes k_c sélectionné reste relativement faible, même s'il croît avec le nombre de classes. Les temps de décision de notre schéma sont alors très proches de ceux d'un schéma *un-contre-tous*.

Les différences entre les principes de décodage d_{vote}, d_{Price}, d_{Hastie}, d_{Wu} pour la deuxième étape de notre schéma hybride sont très faibles. La diminution du nombre de fonctions binaires utilisées et la pertinence de ces différentes fonctions par rapport aux classes à discriminer produisent des problèmes de décodage *un-contre-un* plus faciles à réaliser. Ces deux effets expliquent les comportements fortement similaires de ces quatre principes. Les décodages d_{vote} et d_{Price} étant plus rapides à réaliser, il est donc préférable de les utiliser avec notre schéma hybride.

Par rapport aux schémas IPCCC et ECOC, notre schéma hybride est moins performant en généralisation, mais l'écart tend à diminuer par rapport à ceux relatifs aux schémas *un-contre-un* et *un-contre-tous*. Comme les temps d'entraînement et de décision sont plus faibles, notre schéma hybride représente donc une alternative intéressante pour la construction de fonctions de décision à partir de SVM binaires.

6.2.1.3 Discusion

Les résultats obtenus avec notre schéma de combinaison hybride sont encourageants, mais plusieurs remarques peuvent être formulées pour améliorer ce schéma.

Le critère $c_{\delta_{\max}}$ pour la sélection des classes fortement probables ne tient pas compte de la performance individuelle de chaque fonction de décision binaire impliquée dans la décomposition *un-contre-tous*. Le tableau 6.1 illustre qu'elles peuvent avoir des écarts importants de performances entre elles. De plus le recours à $\delta_{\max}$ pour réaliser cette sélection correspond à une façon de procéder *ad hoc*. L'exploitation de la théorie des fonctions de croyance [QUOST04, QUOST05] devrait permettre de définir une méthode de sélection plus performante pour la détermination de l'ensemble des classes fortement probables, car elle pourrait exploiter à la fois les sorties des fonctions de décision binaires pour un exemple donné et les connaissances sur les capacités de généralisation de chacune de ces fonctions de décision.

L'influence de la sélection des modèles relatifs à chaque fonction de décision binaire sur la fonction de décision multi-classe (*cf.* seconde approche en section 6.2.2) devra être mesurée pour réaliser un classement plus précis des différents schémas de combinaison par rapport à notre schéma hybride.

Les classes peuvent avoir une organisation hiérarchique, par exemple pour des problèmes de types cellulaires (*cf.* section 7.1.3). Il peut alors être intéressant de généraliser le principe de notre schéma hybride pour que la fusion de classes fortement similaires soit réalisée à la première étape et leur distinction dans la seconde. La limitation à seulement deux étapes est également trop restrictive dans ce cas de figure et notre schéma hybride devra être étendu pour avoir autant d'étapes qu'il y a de niveaux dans l'arbre hiérarchique des classes.

Les schémas IPCCC et ECOC peuvent également être hybridés en généralisant notre approche hybride afin de réduire la complexité calculatoire de ces deux schémas. Les détails d'une telle hybridation restent à définir.

6.2.2 Selection multi-modèle par algorithme évolutionnaire

Nous avons expliqué, en section 6.1.3, que la sélection des hyper-paramètres pour les SVM intervenant dans un schéma de décomposition multi-classe est actuellement réalisée suivant deux protocoles. L'un considère que les mêmes hyper-paramètres sont communs à tous les problèmes binaires, l'autre considère qu'ils sont différents pour chaque problème binaire. Pour le second protocole, la sélection des hyper-paramètres est réalisée à partir des performances estimées de chaque problème binaire pris séparément. La question qu'il est possible de se poser pour ce second cas est : "produit-on le meilleur schéma multi-classe possible avec ce protocole par rapport à l'espace des modèles envisageable" ?

En effet, avec le premier protocole, l'espace des modèles est certainement plus réduit, mais les hyper-paramètres communs à l'ensemble des problèmes binaires sont sélectionnés à partir de l'estimation des performances de généralisation du schéma de combinaison. Ceci est réalisé sans tenir compte de la mesure des performances de chaque fonction de décision binaire pour chaque problème binaire exploité. Il semble cependant logique que si le schéma de combinaison multi-classe produit avec cette procédure de sélection est performant, alors la majorité des fonctions de décision binaires exploitées par ce schéma le sont également. Mais rien ne garantit la réciprocité, c-à-d que le fait de combiner des fonctions de décision binaires les plus performantes, produira forcément le schéma de combinaison les exploitant le plus performant. Nous proposons alors de vérifier expérimentalement s'il est possible de produire des contre-exemples. Nous avons choisi d'utiliser le même cadre expérimental que celui de la section 6.2.1.2, sauf que chaque couple $(C,\gamma) \in \{2^{-5},2^{-3},\ldots,2^{15}\} \times \{2^{-5},2^{-3},\ldots,2^{15}\}$ relatif à l'apprentissage d'un SVM pour un problème binaire donné n'est plus sélectionné par rapport aux capacités de généralisation pour le problème binaire considéré, mais par rapport aux capacités de généralisation du schéma de combinaison l'exploitant.

Soit n_c le nombre de couples possibles ($n_c = 121$ dans le cas considéré) et n_d le nombre de fonctions de décision binaires intervenant dans un schéma de combinaison, alors le nombre de modèles potentiel à tester est de $n_c^{(n_d)}$. Il est évident qu'une recherche exhaustive n'est pas réalisable à cause d'un problème d'explosion combinatoire. Nous avons alors décidé d'utiliser une approche méta-heuristique pour réaliser cette sélection. L'exploitation d'un algorithme évolutionnaire (*cf.* section 3.2) a été retenue pour réaliser l'optimisation d'un schéma de combinaison des fonctions de décision binaires lorsque ce schéma utilise des SVM avec un noyau gaussien.

6.2.2.1 Détails relatifs au synopsis de AE

Le synopsis général de l'Algorithme Evolutionnaire (AE) utilisé pour déterminer les hyper-paramètres des SVM d'un schéma de combinaison est le même que celui décrit à la section 3.2.1. Les détails spécifiques à la représentation des individus et à leur sélection, croisement et mutation sont les suivants.

Représentation des individus

Une solution $\boldsymbol{\theta}_j$, caractérisant un individu j de la population, est un vecteur de n_d modèles $\theta_i \equiv (C_i,\gamma_i)$ avec ($1 \leq i \leq n_d$). Chaque modèle θ_i représente les valeurs des

hyper-paramètres utilisées pour produire à partir de l'algorithme des SVM la fonction de décision D_i relative au problème binaire ψ_i. Nous noterons $\theta_{i,j}$ le modèle θ_i relatif à l'individu j, $D_{i,j}$ la fonction de décision D_i relative au modèle $\theta_{i,j}$ et $\tilde{D}_j \equiv \tilde{D}(\boldsymbol{\theta}_j)$ la fonction de décision multi-classe correspondant à la combinaison des fonctions binaires $D_{i,j}$ à partir d'un modèle $\boldsymbol{\theta}_j$.

Initialisation de l'AE

Les solutions $\boldsymbol{\theta}_j$ relatives à la population initiale sont produites par un tirage aléatoire uniforme sur l'espace des modèles : $\theta_i = (2^{u_i-5},2^{v_i-5})$ avec $u_i,v_i \in \{0,1,\ldots,10\}$. La taille λ de la population est une constante qui est fixée à l'initialisation de l'AE. Le nombre maximum d'itérations $n_{\max}$ de l'AE est également fixé à l'initialisation.

Performance des individus

La performance d'un individu j est déterminée à partir de l'estimation des capacités de généralisation de la fonction de décision $\tilde{D}_j$ produite à partir de la solution $\boldsymbol{\theta}_j$. Nous notons e_j le taux d'erreur lié à la fonction de décision $\tilde{D}_j$ et $e_{i,j}$ le taux d'erreur estimé lié à la fonction de décision binaire $D_{i,j}$ pour le problème binaire ψ_i. La performance f_j d'un individu j est déterminée à partir de la relation :

$$f_j \equiv f(\boldsymbol{\theta_j}) = \frac{1}{1-\frac{1}{n_\omega}}\left(1-\frac{1}{n_\omega}-e_j\right) \tag{6.28}$$

On a donc $f_j = 1$ si la fonction de décision $D(\boldsymbol{\theta}_j)$ est parfaite ($e_j = 0$) et $f_j = 0$ si la fonction de décision $D(\boldsymbol{\theta}_j)$ n'est pas plus performante qu'un tirage aléatoire ($e_j = 1-\frac{1}{n_\omega}$). Les performances $f_{i,j}$ des fonctions de décision binaires $D_{i,j}$ sont déterminées également à partir de la relation (6.28) en remplaçant e_j par $e_{i,j}$ ($n_\omega = 2$).

Sélection des individus

La sélection des individus participant à la phase de reproduction et survivant à la prochaine itération de l'algoritme évolutionnaire est réalisée en exploitant la règle de proportionnalité (3.7) avec une pression de sélection égale à deux et une stratégie de répartition SUS (*cf.* section 3.2.4.1). Le nombre d'individus *enfants* μ produit par le croisement de deux individus *parents* est égal au nombre d'individus *parents* λ dans la population initiale. Parmi ces individus (*parents* et *enfants*) seulement λ individus survivent pour la prochaine itération.

Opérateurs de croisement

Deux types d'opérateurs de croisement stochastiques ont été utilisés, nous les noterons $\oplus_1$ et $\oplus_2$. Soit $\boldsymbol{\theta_1}$ et $\boldsymbol{\theta_2}$ les solutions relatives aux deux *parents* intervenant dans un croisement et $\boldsymbol{\theta}$ la solution qui décrit l'*enfant* produit ($\boldsymbol{\theta} = \boldsymbol{\theta}_1 \oplus \boldsymbol{\theta}_2$). Le principe commun à ces deux types d'opérateurs de croisement est d'affecter à chaque modèle θ_i de la solution codant l'*enfant* le modèle $\theta_{i,1}$ ou $\theta_{i,2}$ de l'un des deux parents en fonction d'un tirage

aléatoire. La probabilité de recopier l'un de ces deux modèles dépend des performances liées aux deux *parents*.

– **Opérateur de croisement** $\oplus_1$
Le premier opérateur de croisement exploite seulement les performances globales f_1 et f_2 des deux schémas de combinaison relatifs aux *parents* et a pour relation :

$$p(\theta_i = \theta_{i,1}) = \frac{f_1^2}{f_1^2 + f_2^2} \tag{6.29}$$

L'idée directrice pour aider à la création d'*enfants* performants est qu'il faut qu'ils exploitent plus de modèles en commun avec le *parent* le plus performant qu'avec le *parent* le moins performant.

– **Opérateur de croisement** $\oplus_2$
Le deuxième opérateur de croisement exploite lui les performances internes $f_{i,1}$ et $f_{i,2}$ des fonctions binaires intervenant dans les deux schémas de combinaison relatifs aux *parents* et a pour relation :

$$p(\theta_i = \theta_{i,1}) = \frac{f_{i,1}^2}{f_{i,1}^2 + f_{i,2}^2} \tag{6.30}$$

L'idée directrice est dans ce cas plus fine. Puisque le principe de sélection favorise les croisements avec les *parents* les plus performants, l'utilisation des valeurs f_1 et f_2 n'est pas essentielle et si l'on veut accroître les performances des *enfants* produits, il est préférable de favoriser la sélection des modèles qui correspondent à l'incorporation de fonctions binaires performantes dans le schéma de combinaison résultant. Chaque *parent* est alors mis à contribution suivant la qualité des fonctions binaires qu'il incorpore.

Remarque : L'intérêt de recopier l'un des deux modèles $\theta_{i,1}$ ou $\theta_{i,2}$ pour les opérateurs de croisement, au lieu de créer un nouveau modèle θ_i différent de $\theta_{i,1}$ et $\theta_{i,2}$ à partir de ces deux modèles réside dans le fait que la fonction de décision relative au modèle choisi a déjà été créée et qu'il n'y aura pas besoin de réaliser de nouvelle phase d'entraînement avec les SVM[10].

Opérateurs de mutation

Chaque individu à une probabilité p_m de subir une mutation à chaque itération de l'AE. Si un individu subit une mutation, chaque modèle θ_i a la même probabilité p_m d'être modifié. La modification correspond à un tirage aléatoire uniforme identique à celui utilisé pour l'initialisation des individus.

10. Pour être plus précis, comme une phase de validation croisée en k parties est utilisée pour réaliser les estimations des f_j et $f_{i,j}$, il n'y a pas une fonction de décision, mais k fonctions de décision attachées à un modèle θ_i particulier.

Schéma	un-contre-tous		un-contre-un	
λ	$\oplus_1$	$\oplus_2$	$\oplus_1$	$\oplus_2$
5	49,0%	42,6%	48,8%	46,8%
10	43,0%	41,2%	45,2%	36,2%
20	38,4%	38,0%	44,2%	36,6%
50	38,2%	36,4%	42,6%	34,8%

TAB. 6.5 – *Taux d'erreur $\bar{e}_v$ avec une sélection de modèles par AE pour différentes tailles de population et les deux types d'opérateur de croisement ($p_m = 0{,}01$, $n_{\max} = 100$). La base d'apprentissage (et de test) utilisée correspond au premier échantillonnage (type 1) issu de la base* **Letter**.

6.2.2.2 Expérimentations

Nous avons choisi de mesurer les performances des schémas de combinaison *un-contre-un* (avec le décodage d_{Price}) et *un-contre-tous* en utilisant l'AE décrit à la section précédente pour la sélection de modèle. Nous avons utilisé les mêmes bases de données que celles utilisées avec le schéma hybride (*cf* section 6.2.1.2). Les estimations des performances (6.28) relatives aux fonctions de décision binaires D_i et multi-classes $\tilde{D}$ sont réalisées à partir des taux d'erreur mesurés par une procédure de validation croisée qui est la même que celle utilisée pour notre schéma hybride (*cf.* section 6.2.1.2). Une fonction de décision $\tilde{D}^*$ est ensuite réalisée en utilisant le meilleur modèle $\boldsymbol{\theta}^*$ (*i.e.* l'individu de performance maximale produit par l'AE) pour produire l'ensemble des fonctions de décision binaires nécessaires au schéma multi-classe utilisé. Le taux d'erreur e_v de la fonction $\tilde{D}^*$ est ensuite mesuré à partir de la base de validation.

Le nombre maximum d'itérations $n_{\max}$, la taille de la population λ et le taux de mutation p_m n'ont pas été définis pour l'AE proposé. Plusieurs expérimentations ont été réalisées pour choisir les valeurs de ces constantes. Le tableau 6.5 donne les résultats obtenus avec différentes tailles de la population, un taux de mutation fixé à 1% et un nombre maximum d'itérations fixé à 100. Ces résultats illustrent que la taille de la population a une influence importante sur les performances en généralisation des fonctions multi-classes produites. Ces résultats illustrent également que l'opérateur de croisement $\oplus_2$ est plus performant que $\oplus_1$. D'autres expérimentations relatives au changement du taux de mutation ou du nombre maximum d'itérations ont montré que les performances de l'AE sont moins sensibles aux variations de ces paramètres[11]. De plus, la durée nécessaire à la sélection d'un modèle $\boldsymbol{\theta}^*$ et la taille mémoire nécessaire au stockage des fonctions de décision binaire augmente essentiellement avec la taille de la population. En effet, à cause de la nature des opérateurs de croisement, les phases d'entraînement des SVM sont essentiellement réalisées pendant la première itération de l'AE. Le nombre de fonctions de décision à stocker est, quant à lui, proportionnel à la taille de la population (sauf modèle θ_i commun entre plusieurs individus).

A partir de ces différentes observations, nous avons choisi les valeurs suivantes pour les 3 constantes de notre AE : $\lambda = 50$, $p_m = 0{,}01$, $n_{\max} = 50$. L'opérateur de croisement utilisé est $\oplus_2$, car il est globalement plus performant. Notre AE est ensuite utilisé pour réaliser la sélection de modèles pour l'ensemble des bases exploitées (les 2×20 échan-

11. Bien entendu, une valeur trop grande du taux de mutation ($p_m \geq 5$) ou un nombre trop faible d'itérations ($n_{\max} \leq 20$) dégrade fortement les performances de l'AE.

	un-contre-tous				un-contre-un			
	$t=1$		$t=2$		$t=1$		$t=2$	
	$\bar{e}_V$	$\Delta\bar{e}_V$	$\bar{e}_V$	$\Delta\bar{e}_V$	$\bar{e}_V$	$\Delta\bar{e}_V$	$\bar{e}_V$	$\Delta\bar{e}_V$
Satimage	14,5±2,0%	-0,1%	11,6±1,0%	+0,1%	14,5±2,1%	-0,2%	11,8 ± 1,0%	-0,0%
USPS	11,2±1,5%	-0,7%	8,5±1,6%	-0,3%	11,0±1,8%	-1,8%	8,4±1,6%	-0,5%
Letter	36,3±3,3%	-5,6%	19,7±1,8%	-2,4%	35,9±2,9%	-4,1%	18,6±2,1%	-2,8%

TAB. 6.6 – *Taux d'erreur moyen et écart type des valeurs estimées de e_v avec une sélection de modèles par AE (λ, $p_m = 0{,}01$, $n_{\max} = 100$). La base d'apprentissage (et de test) utilisée correspond au premier échantillonnage (type 1) issu de la base* **Letter**.

tillons des 3 bases) à la section 6.2.1.2. Les deux schémas de combinaison *un-contre-un* et *un-contre-tous* ont été testés avec l'ensemble de ces bases. Le tableau 6.6 donne pour chaque base de référence la valeur moyenne $\bar{e}_v$ (et l'écart type) relatif aux deux tailles de base d'apprentissage. La colonne $\Delta\bar{e}_V$ de ce tableau correspond à l'écart entre le taux d'erreur moyen de la sélection de modèles avec notre AE et celui de la sélection de modèles classique utilisée en section 6.2.1.2. L'ensemble de ces résultats illustre que la sélection multi-modèle $\boldsymbol{\theta}^*$ avec notre AE est la plus performante. Cet effet est d'autant plus important que le problème comporte un nombre important de classes. On constate également une légère augmentation de la variance des performances en généralisation. Néanmoins, en tenant compte du caractère stochastique de la méthode de sélection et de la dimension de l'espace multi-modèle exploré, cette augmentation reste raisonnable.

6.2.2.3 Discusion

Nous avons, à partir de ces premiers résultats, obtenu des contre exemples qui illustrent que la sélection indépendante du modèle optimal pour chaque sous-problème binaire ne produit pas forcément la meilleure fonction de décision pour un schéma optimal de combinaison à base de SVM. Deux raisons peuvent être raisonnablement mises en avant pour cet effet:

1. L'estimation des performances d'un SVM pour un modèle est forcément une approximation de ses vraies performances. Le cumul de toutes ces approximations doit alors être suffisant pour que la combinaison de ces fonctions binaires dites *optimales* ne produise pas la fonction de décision multi-classe optimale; l'effet de variance est accru.
2. Théoriquement rien ne s'oppose à ce que deux fonctions de décision qui ne produisent pas exactement les mêmes décisions aient cependant les mêmes capacités de généralisation (*i.e* elles ne réalisent pas leurs erreurs sur les même exemples, mais produisent, en moyenne, exactement le même nombre d'erreurs). Supposons qu'il soit possible pour chaque fonction binaire d'un schéma de décomposition de produire deux fonctions de ce type[12]. Il y a dans ces conditions $2^{(n_d)}$ fonctions multi-classes différentes et il est difficile de supposer qu'elles auront toutes les mêmes performances en généralisation quel que soit le principe de décodage utilisé. Cette

12. Plusieurs figures de la section 4.2.3.2 illustrent qu'un grand nombre de modèles produisent des fonctions de décision binaires de performances très similaires. Rien ne prouve par contre qu'elles soient différentes dans leurs décisions sans une étude plus approfondie.

possibilité pourrait justifier l'existence d'une variance sur les capacités de généralisation intrinsèque au principe même de combinaison indépendamment des capacités de généralisation de l'ensemble des classificateurs binaires utilisés.

Ces deux raisons expliqueraient également que l'écart entre les deux stratégies de sélection de modèle testées (AE et individuel) augmente avec le nombre de classes.

Si la sélection multi-modèle d'un schéma de combinaison à base de SVM semble donc plus performante au regard des résultats de la section précédente, les temps nécessaires pour réaliser cette sélection sont actuellement très importants. L'utilisation d'un critère d'arrêt prématuré et de technique d'*alpha seeding*[13] devrait permettre dans un premier temps de réduire les temps d'entraînement.

Les fonctions de décision binaires combinées sont globalement toutes performantes en généralisation, même si elles ne sont pas toutes les meilleures. Notre AE peut actuellement produire des fonctions de décision binaires de performances très médiocres à cause de la procédure d'initialisation des modèles $\boldsymbol{\theta}$. L'utilisation d'une technique comme celle exposée à la section 5.3.2.2 (*i.e.* sélection rapide d'hyper-paramètres par quantification vectorielle) permettrait de ne conserver que les modèles θ_i les plus probables pour chaque sous-problème binaire intervenant dans un schéma de combinaison. Cela permettrait de réduire de façon significative la dimension de l'espace de recherche. Les opérateurs de mutation et de croisement pourraient être modifiés pour prendre en compte ces changements afin d'améliorer globalement l'AE.

6.2.3 Conclusion

Dans ce chapitre, nous avons rappelé les différentes techniques utilisées pour construire une fonction de décision à base de SVM lorsque le problème d'apprentissage comporte plus de deux classes. Nous avons dressé un état de l'art des différents schémas de décomposition et des différents principes de décodage proposés ces dernières années. Nous proposons ensuite deux approches liées à la problématique qu'est la production de fonctions multi-classes à partir de la combinaison de fonctions binaires. La première correspond à la définition d'un schéma hybride de combinaison qui exploite les avantages et inconvénients des schémas de combinaison *un-contre-tous* et *un-contre-un*. Nous montrons au travers de différentes expérimentations que ce schéma de combinaison est plus performant que les deux schémas de base qu'il utilise et que cet effet est d'autant plus marqué que le nombre de classes du problème traité est important. Nous montrons également qu'il a l'avantage d'être de complexité moindre que ceux qui lui sont supérieurs, même s'il n'est pas le schéma de combinaison le plus performant en généralisation. La deuxième approche est liée à la sélection d'un multi-modèle optimisant les capacités de généralisation d'un schéma de combinaison à base de SVM. Nous proposons un algorithme évolutionnaire pour réaliser cette sélection. Nous montrons expérimentalement que notre méthode est supérieure à celle classiquement utilisée en ce qui concerne les capacités de généralisation des fonctions de décision multi-classes produites. Nous montrons également que cet effet est d'autant plus marqué que le nombre de classes du problème

13. Les techniques du chapitre 4 n'ont pas été utilisées, car les tests étaient antérieurs au développement de celles-ci. De plus, les remarques dans [DECOST00] sur d'autres types d'utilisation des techniques d'*alpha seeding* sont appropriées à la sélection de modèle par AE et devront faire l'intention de futurs travaux.

traité est important. Plusieurs remarques au travers de discussions insistent sur les raisons des performances de nos deux approches et proposent des améliorations futures de ces approches. Ces remarques indiquent également que ces deux approches (avec leurs améliorations respectives) pourront être combinées pour produire des schémas de classificateurs binaires encore plus performants. La combinaison de classificateurs binaires est un problème qui peut être considéré comme un cas particulier du problème général de sélection/combinaison de classificateurs [KUNCHE04].

CHAPITRE 7

APPLICATIONS

Sommaire

Les résultats présentés dans ce chapitre se rapportent à deux types d'applications. La première est liée au domaine médical à travers la constitution d'un système de tri cellulaire et la seconde à la réalisation d'un expert informatique en qualité visuelle des images couleur.

7.1 Imagerie microscopique médicale

7.1.1 Introduction

Le LUSAC (Laboratoire Universitaire des Sciences Appliquées de Cherbourg) travaille depuis de nombreuses années maintenant à la réalisation d'un système de vision en imagerie microscopique médicale. Les thèses d'Olivier Lezoray [LEZORA00] et de Cyril Meurie [MEURIE05A] ont toutes deux traité ce type de problème par des approches différentes et complémentaires. On peut résumer le fonctionnement d'un tel système en plusieurs blocs logiciels principaux que sont l'acquisition d'images, la segmentation d'images, l'extraction de caractéristiques cellulaires, la catégorisation de cellules. La figure 7.1 présente un tel système de vision ainsi que la façon dont un être humain peut interagir avec celui-ci (paramétrage, annotation et expertise). Nous ne détaillons pas plus ici l'application microscopique et le lecteur intéressé pourra se référer à [LEZORA03, MEURIE05A, MEURIE05B] pour de plus amples informations. Dans cette thèse, nous nous sommes intéressés plus particulièrement à la segmentation d'images couleur et à la catégorisation de cellules. Pour cela nous avons utilisé des séparateurs à vaste marge exploitant les techniques que nous avons développées dans cette thèse (voir chapitres 5 et 6)

7.1.2 Segmentation d'images microscopiques médicales

Plusieurs de nos travaux précédents [SCHUPP00, LEZORA02, MEURIE05A] ont montré que des schémas de segmentation hybrides combinant classification de pixels et opérations morphologiques permettent d'obtenir des segmentations d'images microscopiques médicales de très bonne qualité. Les images microscopiques médicales que nous avons à

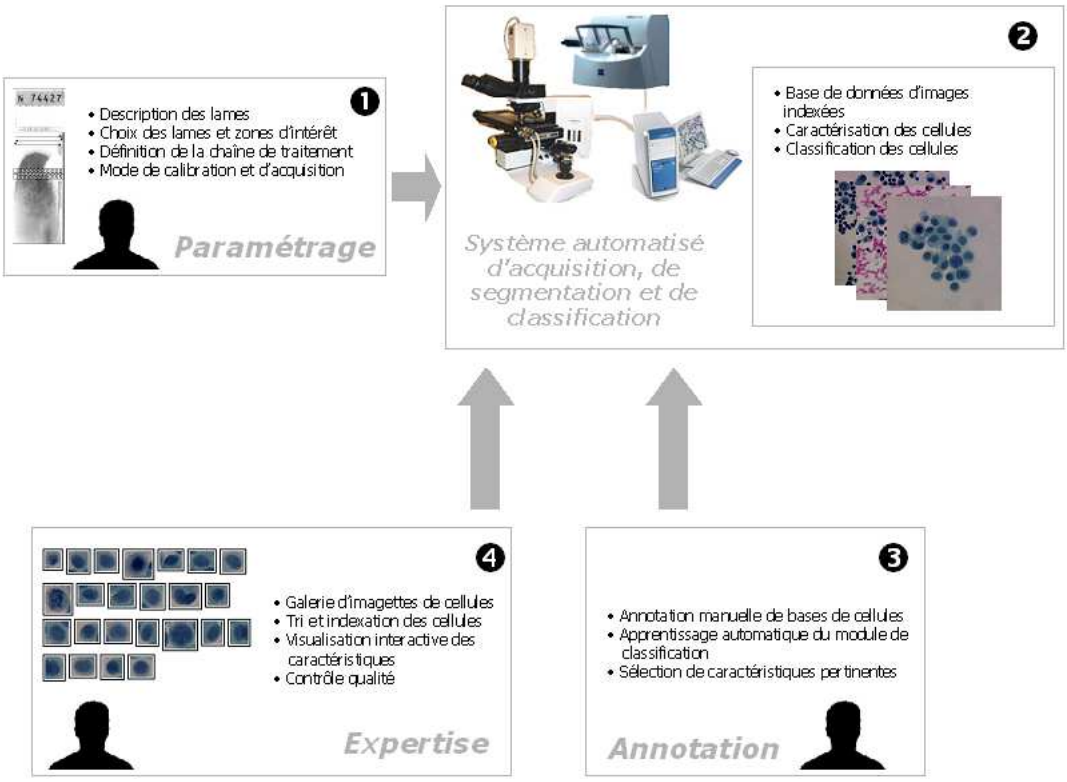
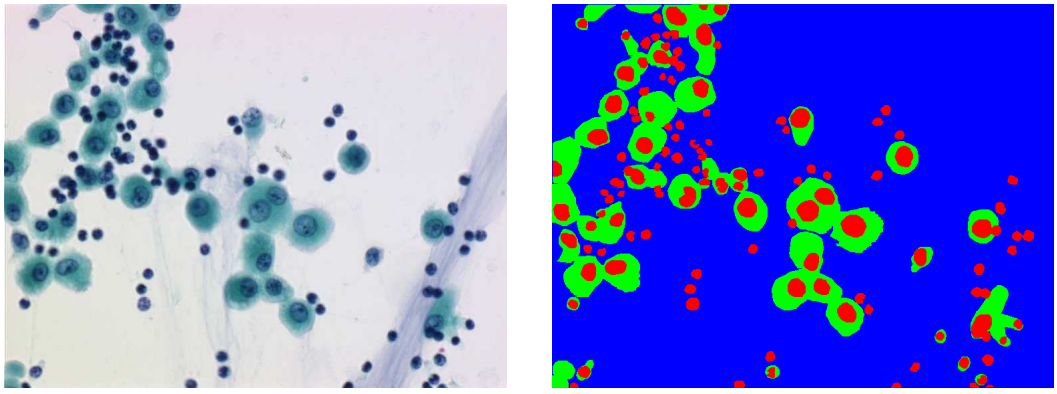


FIG. 7.1 – *Composants du système de vision en imagerie microscopique médicale.*

traiter sont des images provenant d'expectorations bronchiques. Sur ces images on trouve différents types d'objets cellulaires dont la distinction est facilitée par la coloration de Papanicolaou [PAPANI54] qui est appliquée aux lames cytologiques contenant les liquides d'expectorations. Grâce à la coloration des lames, il est possible de classer visuellement les différents pixels d'une image microscopique en trois catégories différentes : le fond, les cytoplasmes et les noyaux (voir figure 7.2). Nous avons pu nous apercevoir, lors de la constitution de vérités terrain, que cette classification ne peut être parfaite même lorsqu'elle est effectuée par un expert. La figure 7.2(b) présente une telle vérité terrain pour la figure 7.2(a) et l'on remarque que certains débris (tels que le mucus coloré en bleu et présent dans le fond) peuvent gêner la différentiation entre les différentes catégories de pixels. Si cela est difficile pour un expert en cyto-pathologie, cela l'est également pour un expert informatique qui aurait à classer les pixels d'une image en trois différentes catégories. Il faut noter de plus que l'utilisation d'informations sémantiques est prépondérante lors de l'évaluation d'une image microscopique par un expert humain de ce champ d'application. En effet, certains pixels présents dans le cytoplasme et le noyau peuvent avoir exactement la même couleur, ce qui montre que ces deux catégories de pixels ont un fort taux de recouvrement. Cependant, dans la thèse de Cyril Meurie [MEURIE05A], plusieurs schémas de classification de pixels ont été étudiés et ont montré leur robustesse même dans le cas d'ambiguités telles que celles que nous venons de soulever. Cyril Meurie a optimisé différents classificateurs supervisés ou non supervisés pour classer les pixels d'une image de microscopie médicale. La couleur d'un pixel étant une information capitale pour ce type de classification, il a étudié l'influence de l'espace couleur sur la méthode de classification utilisée. Il a pu alors montrer que les SVM permettent d'obtenir des classifications pixellaires parmi les meilleures mais l'utilisation des SVM comme classificateurs de pixels



(a) Une image couleur d'un champ microscopique d'une lame d'expectoration bronchique.

(b) La classification manuelle des pixels effectuée par un expert.

FIG. 7.2 – *Une image microscopique médicale (a) et la vérité terrain associée (b).*

implique un fort accroissement de la complexité calculatoire. Ceci est d'un intérêt crucial dans le cadre d'applications industrielles. En effet, même si les SVM permettent d'obtenir des taux de classification correcte très élevés, cela n'est pas le seul critère d'évaluation de la qualité de la fonction de décision. Une fonction de décision exploitée dans le cadre de la classification de pixels est interrogée pour chaque pixel de l'image, une image microscopique comportant un grand nombre de pixels, la complexité calculatoire d'un classificateur de pixels à base de SVM peut rapidement devenir prohibitive. En effet, les SVM sont des algorithmes très performants pour produire des fonctions de décision qui ont de fortes capacités de généralisation, mais dont la complexité croît avec la taille de la base d'apprentissage. L'utilisation des SVM sur des bases de pixels n'est donc pas directement exploitable pour la classification de pixels. Nous avons alors défini une nouvelle méthode qui permet d'utiliser des SVM pour une classification rapide de pixels.

7.1.2.1 Schéma général de segmentation

Le schéma général de segmentation que nous avons retenu est classique et repose sur nos travaux précédents [SCHUPP00, LEZORA02, MEURIE05A]. Il se décompose de la manière suivante :

1. Classification rapide de pixels par SVM.
2. Lissage gaussien des probabilités estimées.
3. Extraction de germes par seuillage des probabilités.
4. Filtrage morphologique.
5. Croissance de régions.

A partir d'une image couleur, nous effectuons une classification en trois classes (fond, cytoplasme et noyau) et nous obtenons trois images de probabilité d'appartenance de chaque pixel à chacune de ces classes. Ces images de probabilités sont lissées puis seuillées afin d'extraire des germes de régions qui sont filtrés par des érosions morphologiques. Enfin, une croissance de régions par ligne de partage des eaux est effectuée sur une image de probabilité correspondant au maximum des trois probabilités précédemment estimées. Dans les sections suivantes, nous détaillons ces différentes étapes.

7.1.2.2 Classification rapide de pixels par SVM

Nous avons défini une nouvelle méthode qui permet d'utiliser des SVM dans le cadre de la classification de pixels. Cela consiste à construire des fonctions de décision binaires de discrimination entre les différentes classes. Pour cela nous disposons d'une base de données initiale constituée de pixels qui ont été manuellement étiquettés par un expert en cytopathologie.

Espace couleur hybride

La couleur des pixels des images numérisées est généralement exprimée dans l'espace couleur (R,V,B). Cependant, cet espace couleur n'est pas nécessairement le plus approprié pour réaliser une classification de pixels et le choix d'un espace couleur adapté permet, dans la pluspart des cas, d'améliorer les performances de la classification de pixels. On peut par exemple choisir de combiner plusieurs classifications effectuées dans différents espaces couleur par des SVM optimisés pour chaque espace couleur [CHARRI06A]. Cependant, la création d'un espace couleur hybride à partir d'un ensemble d'espaces couleur permet généralement d'améliorer la classification de pixels par rapport à l'utilisation de l'espace couleur initial des images [VANDEN98, VANDEN00B, VANDEN03, GUÉ04]. Nous avons choisi ce type d'approche qui consiste à ne pas déterminer quel espace couleur (forcément tri-dimensionnel) permet de produire les meilleurs résultats, mais à déterminer un espace couleur hybride approprié pour chaque fonction de décision binaire. La création d'un espace couleur hybride est réalisée par sélection d'un ensemble de composantes couleur issues de divers espaces couleur [VANDEN03]. Un grand nombre d'espaces couleur existent et ont des propriétes physiques, colorimétriques ou physiologiques spécifiques [VANDEN00B, VANDEN98]. Pour notre application, nous avons retenu des espaces couleur classiques à savoir [TREMEA03]:[1] (R,V,B), (X,Y_1,Z), (L^*,a^*,b^*), (L^*,u^*,v^*), (L_1,C,H_1), (Y_2,Ch_1,Ch_2), (I_1,I_2,I_3), (H_2,S,L_2), (Y_3,C_b,C_r).
Par définition, un espace couleur hybride H_E^β, défini comme suit, correspond à un sous-espace comportant n_β composantes issues d'un espace E qui regroupe n_E composantes couleur provenant de e espaces couleur.

$$H_E^\beta = \bigcup_{i\in[1,\ldots,n_E],\beta_i=1} C_E^i \tag{7.1}$$

C_E^i désigne la composante i de l'espace E . Le vecteur β désigne les composantes couleur de E conservées dans l'espace couleur hybride (*i.e.* $\beta_i = 1$ si la composante i de l'espace E est conservée dans H_E^β, $\beta_i = 0$ sinon). Pour notre application, $e = 9$, $n_E = 25$ et

$$\begin{array}{c} E = (R,V,B,X,Y_1,Z,L^*,a^*,b^*,u^*,v^*,C,H_1, \\ Y_2,Ch_1,Ch_2,I_1,I_2,I_3,H_2,S,L_2,Y_3,C_b,C_r) \end{array} \tag{7.2}$$

L'objectif est alors de trouver l'espace couleur hybride H_E^β (les valeurs du vecteur β) qui améliore la qualité des fonctions de décision produites par des SVM. Cela s'apparente donc à de la sélection d'attributs pertinents. Les composantes C_E^i avec $i > 3$ pour un pixel

1. Nous ajoutons des indices à certaines composantes couleur pour différencier celles qui ont le même nom dans différents espaces couleur mais pas les mêmes formules de calcul.

d'une image sont déterminées à partir des composantes couleur R, V et B du même pixel par des transformations linéaires ou non linéaires [VANDEN00A]. Le coût en temps de calcul de ces transformations est plus ou moins important suivant la nature de la transformation et le fait qu'elle soit ou non implantée directement sur une puce électronique. Soit κ_i le coût de la transformation pour obtenir la composantes C_E^i. Pour une conversion logicielle, $\kappa_i \neq 1, \forall i$ et pour une conversion matérielle $\kappa_i = 1, \forall i$. Le coût associé à l'espace couleur hybride H_E^β est alors défini par : $\text{coût}(\beta) = \sum_{i=1}^{n_E} \beta_i \kappa_i$.

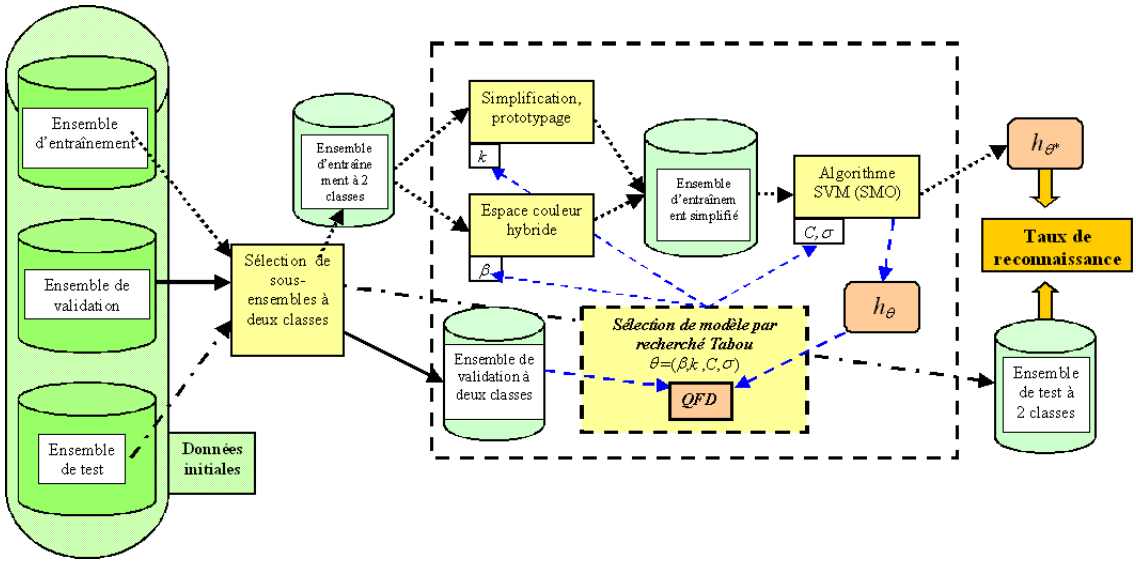


FIG. 7.3 – *Principe général du schéma de classification de pixels par sélection de modèle.*

Principe général

La base de données de pixels dont nous disposons est très grande et très redondante notamment car beaucoup de pixels d'une même classe sont très similaires. La simplification de cette base d'apprentissage est donc nécéssaire pour sélectionner les exemples pertinents et obtenir une classification rapide. Pour effectuer une classification de pixels d'images microscopiques médicales, nous devons donc à la fois simplifier la base d'apprentissage et en extraire les exemples pertinents, simplifier la représentation des pixels pour construire un espace hybride couleur et sélectionner les attributs pertinents, et nous devons également sélectionner les hyper-paramètres optimaux de chaque Fonction de Décision Binaire (FDB) induite par la classification par des SVM en trois classes. Ceci correspond exactement au cadre général de sélection de modèle que nous avons défini au chapitre 5. Nous pouvons, dans le cas de la classification de pixels, résumer son principe général par la figure 7.3. A partir de la base des données initiales, sont sélectionnés des sous-ensembles de données concernant uniquement deux classes (par une approche *un-contre-tous* ou *un-contre-un*). On obtient donc un ensemble de données d'entraînement, de test et de validation pour chaque FDB à base de SVM qui doit être construite. Pour chaque fonction de décision, on sélectionne le modèle optimal par un algorithme à base de recherche Tabou. Cette sélection de modèle permet de déterminer le niveau de simplification de la base (k), l'espace couleur couleur hybride (β), les paramètres de la FDB à base de SVM (C, σ). Pour obtenir

des probabilités par chaque FDB, la méthode Platt est utilisée [PLATT99A].

Résultats

Nous avons donc appliqué notre méthode de sélection de modèle à la classification de pixels issus d'images microscopiques. Les bases d'entraînement Z^E, de test Z^T et de validation Z^V sont construites à partir de 8 images couleurs microscopiques de tumeurs bronchiques (codées en RVB, 574×752 pixels). Pour chaque image, une segmentation manuelle : fond (classe 1), cytoplasme (classe 2), noyau (classe 3) est réalisée par un expert. La base de données a été découpée en trois sous-ensembles d'entraînement, de test et de validation avec respectivement les images suivantes : les deux premières, les deux suivantes et les quatres restantes. Comme la répartition du nombre de pixels dans chaque classe n'est pas équilibrée dans les images (1: 89%, 2: 7%, 3: 4%), seul un sous-ensemble des pixels des classes 1 et 2 a été sélectionné aléatoirement pour la constitution de Z^E et Z^T, ceci afin que chaque classe possède le même nombre d'exemples (60000 par classe). La sélection de 3 modèles θ^i ($i \in [1,2,3]$) est réalisée avec notre méthode à partir des trois bases Z_i^E induites par les décompositions *un-contre-tous* ou *un-contre-un* appliquées à la base Z^E. Pour chaque modèle θ_i une FDB D_θ^i est produite. Pour éviter tout biais lié à la sélection d'un modèle, le taux de reconnaissance est évalué pour une FDB D_θ^i à partir de la base Z_i^V.

$\mathbf{c_{VS}}$	**FDB**	**ECH**	$\mathbf{n_{VS}}$	**k**	$\mathbf{T_R}$
0,003	D_θ^1	RY_1	12	3	96,57
0,003	D_θ^2	VLI_2	2623	11	87,02
0,003	D_θ^3	B	6	2	89,85
0,010	D_θ^1	u^*S	2	1	96,24
0,010	D_θ^2	u^*Ch_1	8	3	84,94
0,010	D_θ^3	B	2	0	89,56

TAB. 7.1 – *Sélection de modèle pour différentes FDB : D_θ^1 (fond contre (cytoplasme et noyau)), D_θ^2 (cytoplasme contre (fond et noyau)), D_θ^3 (noyau contre (fond et cytoplasme)). Les colonnes présentent le niveau du simplification (k), le nombre de vecteurs support (n_{VS}), le taux de reconnaissance (T_R), l'espace couleur hybride (ECH) pour une FDB et un niveau de pénalité (c_{VS}).*

Le tableau 7.1 illustre la sélection de modèle pour des FDB suivant une décomposition *un-contre-tous* avec $c_\beta = 0.01$ et $\kappa_i = 1$, $\forall i$. On remarque tout d'abord que l'augmentation de la pénalité c_{VS} associée au nombre de vecteurs de support (n_{VS}) permet de réduire fortement ce nombre avec une faible diminution du taux de reconnaissance (T_R). Ainsi, il est possible de produire des FDB efficaces et de complexité réduite. On remarque également que le niveau de simplification (k) et la sélection d'un espace couleur hybride (ECH) dépendent du problème de discrimination. Par exemple, la discrimination cytoplasme contre (fond et noyau), qui est la FDB D_θ^2, est la plus difficile et la diminution de la pénalité permet d'augmenter le taux de reconnaissance mais nécessite plus de prototypes représentatifs de la base d'entraînement. Cela se traduit par une augmentation du temps de décision associé à une telle FDB.

	D^1_θ		D^2_θ		D^3_θ	
ECC	$\mathbf{T_R}$	$\mathbf{n_{VS}}$	$\mathbf{T_R}$	$\mathbf{n_{VS}}$	$\mathbf{T_R}$	$\mathbf{n_{VS}}$
RVB	95,22	2	84,35	13	89,74	4
XY_1Z	95,29	2	84,46	12	90,10	5
$L^*a^*b^*$	94,86	2	83,58	7	88,68	2
$L^*u^*v^*$	**96,41**	**4**	84,90	10	89,25	8
LCH_1	95,86	4	84,76	60	89,25	4
$Y_2Ch_1Ch_2$	96,00	2	85,74	46	89,82	4
$I_1I_2I_3$	95,73	2	**85,41**	**6**	89,76	4
H_2SL_2	95,93	3	84,94	6	89,90	9
$Y_3C_BC_R$	95,38	2	86,17	149	**90,13**	**5**
Moyenne	95,63	2,6	84,92	34,3	89,62	5,0

TAB. 7.2 – *Taux de reconnaissance et nombre de vecteurs support pour des fonctions de décision binaires par notre sélection de modèle pour 9* ***espaces couleur classiques*** *(ECC) (*$c_{VS} = 0{,}01$*). Les meilleurs taux de reconnaissance de chaque FDB sont indiqués en gras.*

Le tableau 7.2 présente des résultats similaires à ceux du tableau 7.1, mais cette fois nous avons fixé les composantes de l'espace couleur hybride (ECH) avec des composantes d'espaces couleur classiques (ECC). Pour chaque sélection de modèle, cette fois, seuls le niveau de simplification et les hyper-paramètres des FDB à base de SVM doivent être ajustés. Pour avoir une comparaison possible avec les précédentes expérimentations, nous avons fixé $c_{VS} = 0.01$. La décomposition considérée est toujours de type *un-contre-tous*. On remarque que chaque FDB, une fois optimisée, nécessite un espace couleur spécifique soit 9 composantes ($L^*u^*v^*$ pour D^1_θ, $I_1I_2I_3$ pour D^2_θ et $Y_3C_BC_R$ pour D^3_θ) au total pour les trois FDB. De même, chaque FDB nécessite des vecteurs de support différents, soit 15 au total. Avec la recherche d'un espace couleur hybride, comme cela est décrit dans le tableau 7.1, seules 4 composantes couleurs différentes sont nécéssaires (u^*, S, Ch_1, B) et seulement 12 vecteurs de support, tout ceci avec un taux de reconnaissance très proche (environ 0,5% de différence). Nous pouvons donc dire que, même si des schémas basés sur l'exploitation d'espaces couleur classiques peuvent être efficaces, ceux-ci sont globalement plus complexes que ceux construits par notre approche réalisant une sélection d'un espace couleur hybride, même si elle s'accompagne d'une très faible différence en termes de taux de reconnaissance pour cette application.

Nous venons de voir, à travers les deux précédentes expérimentations, qu'un compromis entre efficacité et complexité permet de sélectionner un modèle performant. Cependant, nous avons considéré uniquement une décomposition de type *un-contre-tous* et l'utilisation d'un autre schéma de décomposition (*un-contre-un* par exemple) peut avoir des conséquences sur les résultats que nous venons de commenter. Pour comparer les deux approches, nous utilisons les probabilités estimées par chacune d'entre elles pour effectuer une décision de classification et donc classer les pixels. Dans le cas d'une approche *un-contre-tous* la probabilité maximum est utilisée pour désigner la classe retenue et dans le cas d'une approche *un-contre-un*, nous avons retenu l'approche de Price [PRICE94] pour sa rapidité. Afin de comparer ces deux approches de décomposition, nous avons mesuré le taux de reconnaissance du schéma global (T_R), le nombre total de vecteurs de support (n_{VS}), le temps d'entraînement total (TE), le temps moyen de classification d'une image complète ($\overline{TCI}$). Pour le nombre de vecteurs support et les temps d'entraînement, ce sont

les sommes des quantités associées à chaque fonction de décision binaire. Les tableaux 7.3 et 7.4 montrent les résultats de ces expérimentations. Tout d'abord on remarque que les deux schémas de décomposition produisent des solutions dont les taux de reconnaissance sont proches, mais le temps moyen de classification des pixels d'une image est beaucoup plus important avec une approche *un-contre-tous*. La principale raison est que dans un schéma *un-contre-un*, les FDB utilisent un ensemble d'entraînement beaucoup plus réduit et leur discrimination est généralement plus simple. Les temps d'entraînement sont également plus réduits avec une approche *un-contre-un*, ceci s'accentuant avec la recherche d'un espace couleur hybride.

ECC	$\mathbf{T_R}$	$\mathbf{n_{VS}}$	**TE**	$\overline{\mathbf{TCI}}$
RVB	86,55	479	2639	10,32
XY_1Z	86,80	1364	12017	29,22
$L^*a^*b^*$	86,74	745	3856	16,80
$L^*u^*v^*$	86,35	2680	5761	61,98
LCH_1	85,97	1239	6785	27,40
$Y_2Ch_1Ch_2$	87,09	303	6404	6,58
$I_1I_2I_3$	86,85	2589	4760	54,11
H_2SL_2	86,02	2520	2899	55,52
Y_3CBCR	86,67	519	2668	11,08
Average	86,56	1382	5310	30,34
ECH	$\mathbf{T_R}$	$\mathbf{n_{VS}}$	**TE**	$\overline{\mathbf{TCI}}$
RVY_1LI_2 ($\kappa_i = 1$)	87,18	2641	75158	63,97
$RVBY_1L$ ($\kappa_i \neq 1$)	86,97	2532	76256	54,28

TAB. 7.3 – *Taux de reconnaissance global (T_R), nombre de vecteurs de support (n_{VS}), temps d'entraînement (TE) en seconde, temps de classification moyen d'une image ($\overline{TCI}$) en seconde pour une décomposition* ***un-contre-tous*** *avec un espace couleur classique (ECC) ou bien un espace couleur hybride (ECH). Les constantes de pénalisation sont* $c_{\mathrm{VS}} = 0{,}003$ *et* $c_\beta = 0{,}01$.

Pour terminer ces expérimentations, nous avons, pour une décomposition *un-contre-un*, étudié l'influence du compromis entre taux de reconnaissance et temps de traitement d'une image. Pour cela, nous avons augmenté la pénalité c_{VS}. Ces résultats sont décrits par le tableau 7.5. On peut constater que les solutions privilégiant une conversion logicielle ($\kappa_i \neq 1, \forall i$) sont plus rapides que celles privilégiant une conversion matérielle ($\kappa_i = 1, \forall i$), mais le taux de reconnaissance est moins bon. Nous avons cependant retenu comme classificateur de pixels cette dernière configuration (décrite par la première ligne du tableau 7.5) qui opère une décomposition *un-contre-un* avec conversion matérielle (pas de pondération de chaque composante couleur). Nous avons donc privilégié une classification rapide. Ceci s'explique par le fait que pour de faibles variations du taux de reconnaissance, les classifications obtenues produiront la même segmentation finale car le schéma de segmentation utilise des étapes de filtrage des germes extraits par la classification qui annulent les différences entre ces classifications [LEBRUN04B].

ECC	$\mathbf{T_R}$	$\mathbf{n_{VS}}$	TE	$\overline{\mathbf{TCI}}$
RVB	86,86	988	968	21,38
XY_1Z	87,06	880	1258	20,18
$L^*a^*b^*$	86,69	50	1269	1,97
$L^*u^*v^*$	80,00	13	517	1,07
LCH_1	86,24	128	812	3,81
$Y_2Ch_1Ch_2$	84,57	35	429	1,42
$I_1I_2I_3$	86,57	851	1488	19,04
H_2SL_2	86,36	146	762	3,81
Y_3CBCR	86,28	36	731	1,51
Average	85,63	347	915	8,24
ECH	$\mathbf{T_R}$	$\mathbf{n_{VS}}$	TE	$\overline{\mathbf{TCI}}$
RVY_1LI_2 ($\kappa_i = 1$)	86,97	12	4622	2,13
$RVBY_1L$ ($\kappa_i \neq 1$)	86,13	11	4843	1,87

TAB. 7.4 – *Taux de reconnaissance global (T_R), nombre de vecteurs de support (n_{VS}), temps d'entraînement (TE) en seconde, temps de classification moyen d'une image ($\overline{TCI}$) en seconde pour une décomposition **un-contre-un** avec un espace couleur classique (ECC) ou bien un espace couleur hybride (ECH). Les constantes de pénalisation sont $c_{\mathrm{VS}} = 0{,}003$ et $c_\beta = 0{,}01$.*

ECH	$\mathbf{T_R}$	$\mathbf{n_{VS}}$	TE	$\overline{\mathbf{TCI}}$
Ru^*v^* ($\kappa_i = 1$)	86,67	10	1069	1,71
RVB ($\kappa_i \neq 1$)	85,97	15	1128	0,81

TAB. 7.5 – *Taux de reconnaissance global (T_R), nombre de vecteurs support (n_{VS}), temps d'entraînement (TE) en seconde, temps de classification moyen d'une image ($\overline{TCI}$) en seconde pour une décomposition **un-contre-un** avec un espace couleur hybride (ECH). Les constantes de pénalisation sont $c_{\mathrm{VS}} = 0{,}01$ et $c_\beta = 0{,}01$.*

7.1.2.3 Segmentation : étapes suivantes

La classification rapide de pixels par SVM permet de produire les cartes de probabilité d'appartenance de chaque pixel à l'une des classes. Les images 7.4(b) à 7.4(d) illustrent les résultats obtenus avec l'image microscopique 7.4(a). La figure 7.4(e) présente le résultat obtenu en affectant chaque pixel à sa classe la plus probable, les probabilités étant calculées avec la méthode de Price [PRICE94]. Les étapes suivantes de la segmentation ont différents paramètres qui doivent être réglés efficacement afin d'améliorer la qualité de la segmentation finale. L'étape de lissage gaussien des probabilités estimées nécessite de spécifier la largeur de bande du filtre gaussien (w_s). L'étape d'extraction des germes par seuillage des probabilités nécessite de déterminer les seuils (T_i) à partir desquels un pixel est considéré comme appartenant à un germe potentiel d'une classe. L'étape de filtrage morphologique nécessite de déterminer combien d'itérations (n_e) du filtrage morphologique sont effectuées afin d'éliminer les artefacts éventuels. Tous ces paramètres peuvent être réglés a priori mais il est préférable de chercher à les optimiser étant donné leur nombre (5 paramètres différents). Nous avons donc utilisé une recherche tabou qui opti-

mise un critère de qualité de la segmentation d'images microscopiques médicales.

Qualité de la segmentation d'images microscopiques médicales

Ce critère de qualité, noté q_{seg}, doit prendre en compte l'adéquation q_{forme} entre la forme des objets produits par une segmentation automatique (I_a) et celle d'un expert (I_e). Ce critère doit également prendre en compte le nombre d'artefacts introduits ($\eta_{artefact}$) ainsi que le nombre d'objets non segmentés ($\eta_{manques}$). Le critère que nous proposons est alors le suivant :

$$q_{seg} = q_{forme} + \lambda(\eta_{manques}) + (1-\lambda)\eta_{artefacts} \tag{7.3}$$

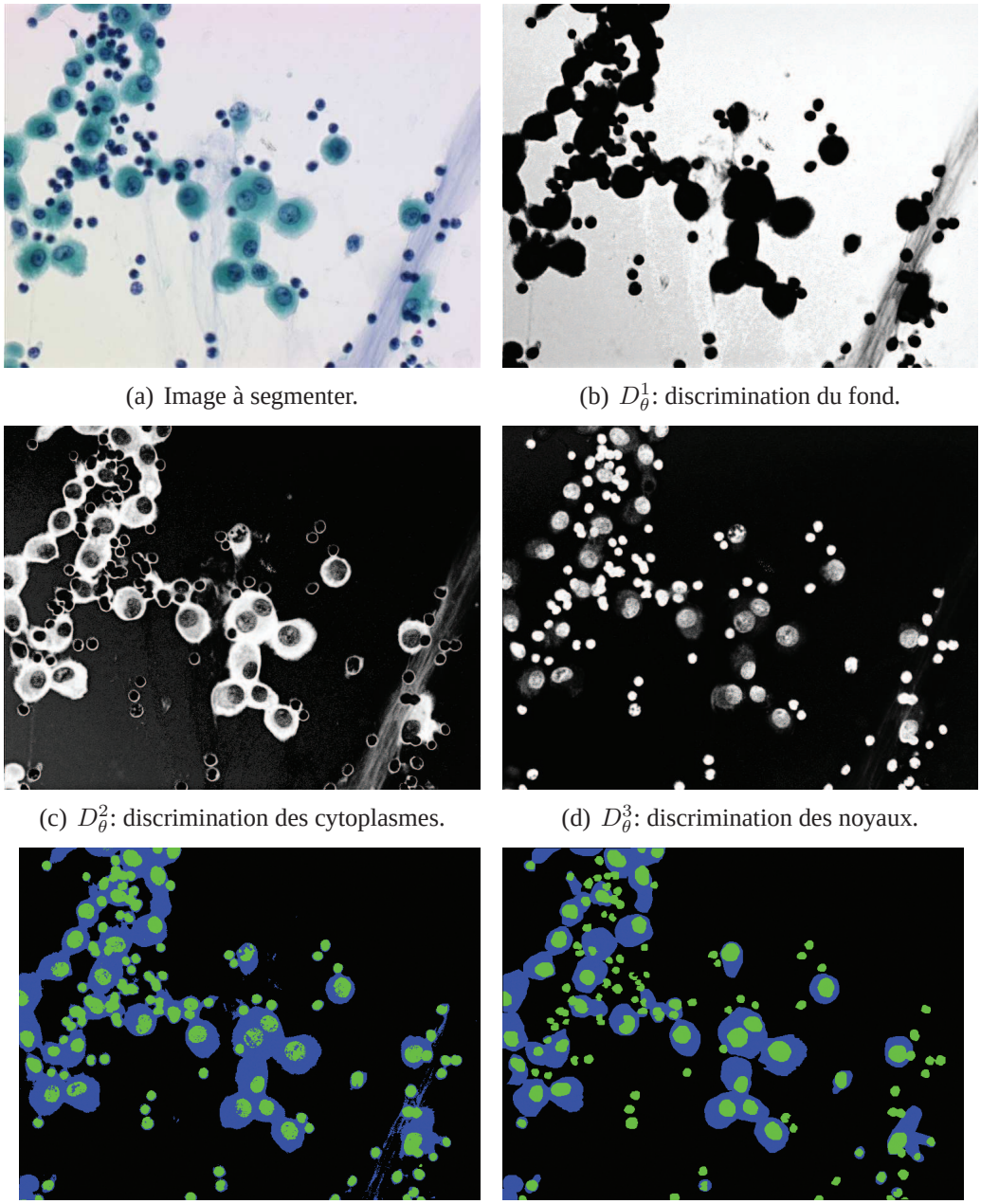
avec $\lambda \in [0,1]$. La constante λ permet de favoriser une segmentation limitant le nombre d'objets non segmentés en comparaison du nombre d'artefacts. Dans le cas de la segmentation de cellules, il est essentiel de ne manquer aucune cellule, même si cela amène à considérer quelques artefacts étant donnée la faible proportion de cellules cancéreuses parmi les cellules normales. Nous avons donc fixé $\lambda = 0.9$. L'adéquation q_{forme} est définie comme suit :

$$q_{forme} = \frac{1}{|I_a|} \sum_{I_a(p) \neq I_e(p)} min\left(d_e(p,I_e), d_{max}\right)^2 \tag{7.4}$$

où $d_e(p,I_e)$ correspond à la distance entre un pixel p de la segmentation automatique I_a et le plus proche voisin de p correspondant à une frontière dans la segmentation de l'expert I_E. d_{max} sert à ne pas trop s'éloigner des frontières et est fixé à 4. A l'aide d'une optimisation tabou de q_{seg}, nous avons pu déterminer l'ensemble des valeurs des paramètres à savoir : $w_s = 3$, $T_1 = 0.42$, $T_2 = 0.64$, $T_3 = 0.56$ et $n_e = 2$.

La figure 7.5(e) présente un exemple de segmentation sur l'image couleur 7.5(a). On peut constater que la segmentation produite est très proche de celle de l'expert (figure 7.4(f)). Si l'on compare le résultat de la segmentation avec celui de la classification de pixels seule (figure 7.5(c)), on s'aperçoit de l'amélioration obtenue avec les étapes de segmentation. Pour illustrer ceci, à partir de la classification de l'expert (figure 7.4(f)), on peut en déduire une carte de pondération des erreurs (figure 7.5(b)) selon la formulation de q_{forme} (les erreurs sont prises en compte dans un voisinage de distance maximum d_{max} à chaque pixel de frontière dans I_e). Les figures 7.5(d) et 7.5(f) présentent alors les valeurs de q_{forme} pour chaque pixel, respectivement pour la classification rapide de pixels et pour la segmentation finale. On remarque que la segmentation finale est la plus proche de celle de l'expert.

Afin de quantifier le gain produit par les étapes de la segmentation qui suivent la classification de pixels, le tableau 7.6 présente une comparaison entre le schéma complet de segmentation avec optimisation des paramètres et la classification rapide de pixels seule en combinant les probabilités par la méthode de Price [PRICE94]. Pour toutes les images, la segmentation permet d'améliorer les résultats au niveau global (q_{seg}), mais également au niveau de la forme des objets cellulaires (q_{forme}). Ceci illustre bien la nécessité de raffiner la classification de pixels car elle n'utilise aucune information de connexité spatiale dans l'image. La figure 7.6 présente des résultats de segmentation d'images en comparaison aux vérités terrains réalisées par un expert. On constate la qualité des segmentations produites par notre approche.

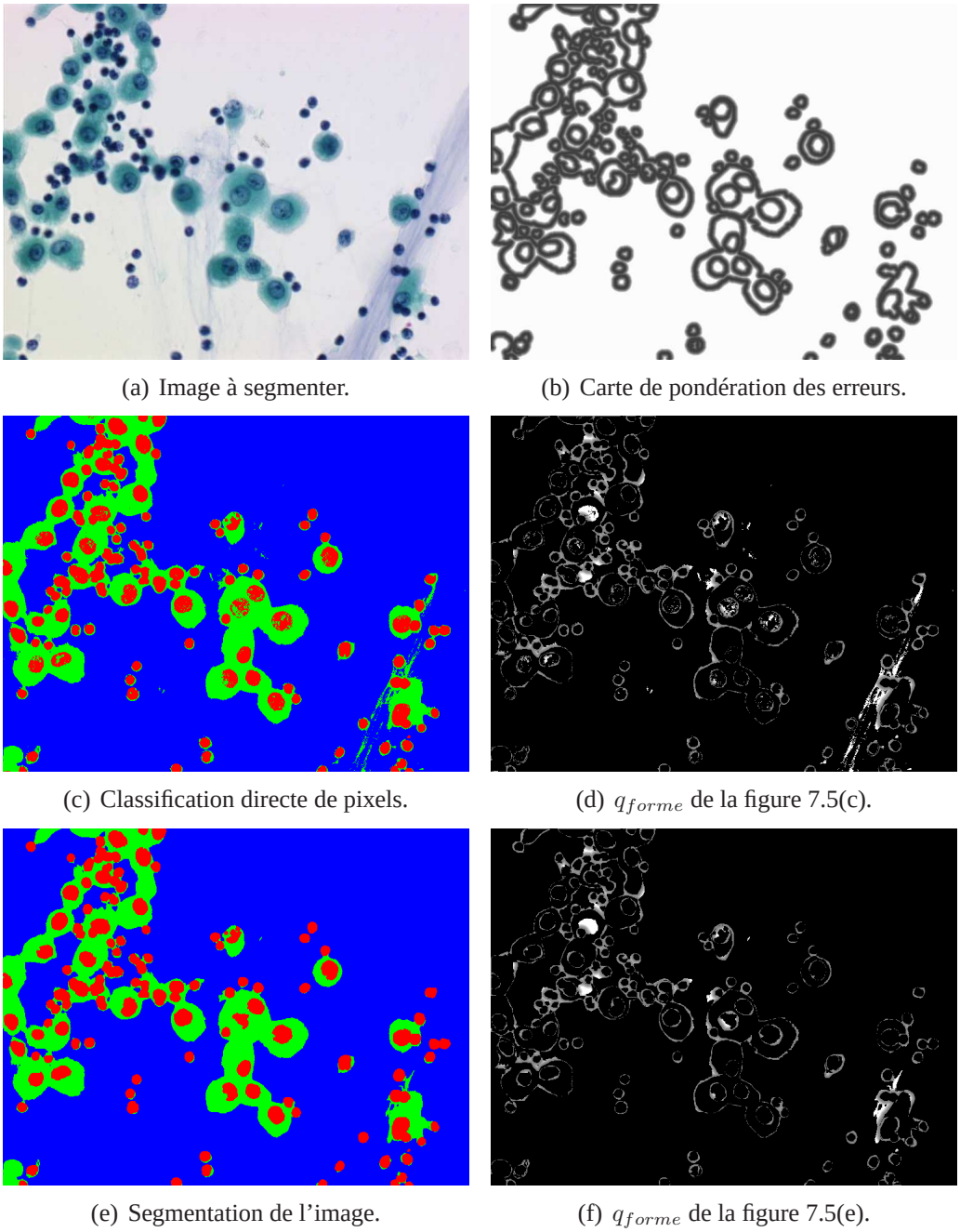


(a) Image à segmenter. (b) D_θ^1: discrimination du fond.

(c) D_θ^2: discrimination des cytoplasmes. (d) D_θ^3: discrimination des noyaux.

(e) Classification de pixels. (f) Classification de l'expert.

FIG. 7.4 – *Classification de pixels produite par la combinaison des fonctions de décision (D_θ^1, D_θ^2, D_θ^3).*



(a) Image à segmenter. (b) Carte de pondération des erreurs.

(c) Classification directe de pixels. (d) q_{forme} de la figure 7.5(c).

(e) Segmentation de l'image. (f) q_{forme} de la figure 7.5(e).

FIG. 7.5 – *A partir de la segmentation de l'expert (figure 7.4(f)), une carte de pondération des erreurs est construite (figure 7.5(b)) selon la formulation de* q_{forme}*. La classification directe de pixels (figure 7.5(c)) a un* $\overline{q_{forme}} = 0.769$ *et la segmentation de l'image a un* $\overline{q_{forme}} = 0.492$*.*

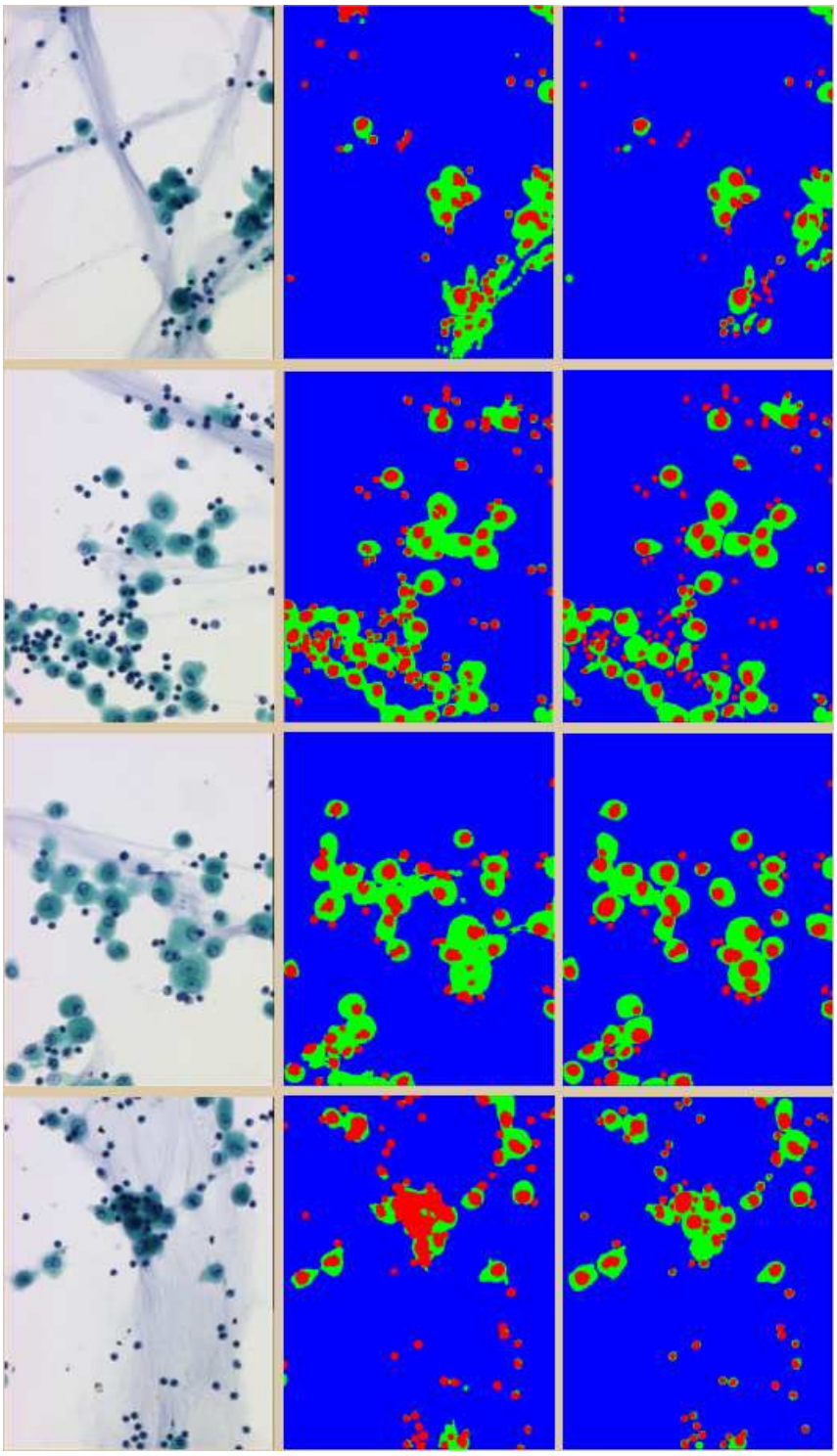


FIG. 7.6 – *Galerie d'images microscopiques médicales couleur (première colonne), les segmentations obtenues (deuxième colonne) et les segmentations des experts (dernière colonne).*

	Classification		**Segmentation**	
Image	q_{seg}	q_{forme}	q_{seg}	q_{forme}
0	17,24	1,04	4,60	0,80
1	6,11	1,01	4,36	0,96
2	11,00	0,90	6,26	0,66
3	24,59	2,19	4,89	1,49
4	6,34	0,24	2,30	0,20
5	6,42	0,52	2,45	0,45
6	7,33	0,73	4,58	0,68
7	13,99	0,79	3,17	0,57
moyenne	11,63	10,70	4,08	0,73

TAB. 7.6 – *Comparaison de la qualité de la segmentation (q_{seg}) et de la qualité des formes des objets (q_{forme}) pour 8 images microscopiques médicales avec la classification de pixels seule et l'approche globale de segmentation.*

7.1.3 Tri cellulaire avec des SVM

La première étape (hors acquisition) pour la réalisation d'un système d'aide au tri cellulaire par analyse d'image nécessite l'extraction des objets à trier dans une image microscopique. La segmentation décrite précédemment permet de réaliser rapidement l'extraction de ces objets[2]. La seconde étape correspond à la reconnaissance de la classe des cellules extraites. Il s'agit d'une problématique de reconnaissance des formes. Pour que cette reconnaissance soit automatique, il est nécessaire de réaliser une base de données représentative des différentes cellules à identifier. La constitution de cette base a été réalisée pendant la thèse de Olivier Lezoray [LEZORA00]. Chaque cellule est décrite par un ensemble de 46 attributs numériques qui appartiennent à trois groupes (forme, texture, couleur) et par son type cellulaire déterminé par un expert en cytopathologie (l'oracle du chapitre 2). La figure 7.7 donne des statistiques sur cette base, ainsi qu'un exemple visuel de chaque type cellulaire. Certains types cellulaires ne sont pas présents dans la base car la probabilité de les observer est très faible. Le nombre de types cellulaires dans la base est donc actuellement[3] de 12.

L'objectif principal du système de tri cellulaire est l'aide au diagnostic pour le *screening* des lames cytologiques. Le tri entre cellule maligne et bénigne est donc essentiel. Cependant les experts médicaux pensent que l'obtention du type cellulaire de chaque cellule est une information supplémentaire importante pour le diagnostic médical complet.

Deux problèmes de classification supervisée ont donc été formulés à partir de ces données. Le premier correspond à une discrimination entre cellule maligne et cellule bénigne et le second en une reconnaissance des types cellulaires[4]. Pour ces deux problèmes, la

2. Pour être plus précis, il est nécessaire de traiter également le problème de découpage des cellules qui se touchent, nous ne rentrerons pas ici dans ces détails et laissons le lecteur intéressé se reporter à [LEZORA00] pour plus de détails sur l'étape complète d'extraction.

3. Dans des travaux futurs, il sera nécessaire de tenir compte de la possibilité de faire évoluer dynamiquement cette base, au mininum pour l'ajout d'exemples des types cellulaires non encore observés, mais pour les expérimentations de cette thèse elle est considérée comme invariante.

4. Actuellement s'il y a contradiction entre les deux processus décisionnels induits par ces deux problèmes, la cellule sera considérée comme suspecte.

base a été divisée en deux parties, la première constitue la base d'apprentissage pour les différentes expérimentations et la seconde est réservée à la validation des fonctions de décision produites par ces expérimentations. La description des bases d'apprentissage et de validation relatives à ces deux problèmes est donnée en annexe A sous les noms respectifs de **Serous-2c** et **Serous**.

Classe cellulaire	Malin	Nombre (3957)
Polynucléaires neutrophiles	Non	360
Polynucléaires éosinophiles	Non	279
Polynucléaires altérés	Non	217
Lymphocytes non activés	Non	754
Lymphocytes activés	Non	357
Macrophages	Non	631
Sidérophages	Non	0
Mésothéliales	Non	600
Mésothéliales dystrophiques	Non	38
Mésothéliomes	Oui	27
Adénocarcinomes	Oui	399
Carcinomes épidermoïdes	Oui	0
Lymphomes	Oui	0
Sarcomes	Oui	0
Mélanomes malins	Oui	0
Mitose Anormales	Oui	33
Mitoses Normales	Non	0
Lobe de Polynucléaire	Non	262

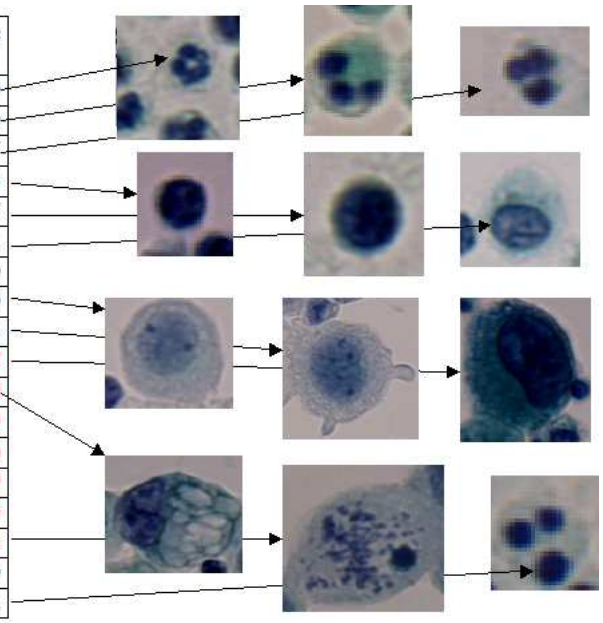


FIG. 7.7 – *Constitution de la bases de données relative au problème de tri cellulaire.*

7.1.3.1 Classification maligne et bénigne

Le détection de la malignité des cellules avec des SVM[5] a un taux de reconnaissance élevé de 97,8% (voir tableau A.5). Si l'on tient compte du fait que le nombre d'exemples de cellules malignes (90) est plus faible que celui de cellules bénignes (694) dans la base de validation[6], le taux de reconnaissance balancé (1-e_{BER}, *cf.* section 5.3.2.1) baisse légérement (97,4%), mais reste de très bonne qualité[7]. Le tri des cellules malignes peut donc être réalisé automatiquement avec de très bonnes performances. L'extraction de l'ensemble des caractéristiques des cellules a par contre un coût calculatoire important. Il est alors important de se poser la question de l'apport de ces attributs.

Importance des attributs

Cette expérimentation a pour objectif de mesurer l'importance des attributs. L'algorithme SFFS (*sequential forward floating selection*) [PUDIL94] est utilisé pour réaliser la sélection d'attributs avec des SVM (noyau gaussien). Une procédure de validation croisée en 5 parties est utilisée pour réaliser la sélection des hyper-paramètres (C,γ) des

5. La sélection hyper-paramètres a été réalisée par une procédure de validation croisée avec $(C,\gamma) \in \{2^{-5},2^{-3},\ldots,2^{15}\} \times \{2^{-5},2^{-3},\ldots,2^{15}\}$.

6. Il y a environ la même proportion avec la base d'apprentissage.

7. Olivier Lezoray obtient un taux de reconnaissance balancé de 96,7% [LEZORA00] avec l'architecture MONNA [LEZORA00].

SVM parmi l'ensemble de couples $\{2^{-5},2^{-3},\ldots,2^{15}\} \times \{2^{-5},2^{-3},\ldots,2^{15}\}$[8]. L'initialisation de SFFS est réalisée sans aucun attribut sélectionné[9] et pendant la procédure SFFS la meilleure fonction de décision pour chaque cardinalité de l'ensemble d'attributs utilisée est sauvegardée[10]. Pour l'ensemble de ces fonctions de décision le taux de reconnaissance balancé est déterminé à partir de la base de validation. La figure 7.8 illustre l'évolution de ce taux en fonction du nombre d'attributs. On peut observer qu'au-delà de 13 attributs les performances en généralisation sont quasi-constantes (maximum de 98,9% avec environ 20 attributs), même si une légère baisse se produit lorsque que plus de 20 attributs sont utilisés. Le taux de reconnaissance de 97,4% obtenu avec l'ensemble des ces attributs confirme cette tendance. Plusieurs remarques peuvent être formulées :

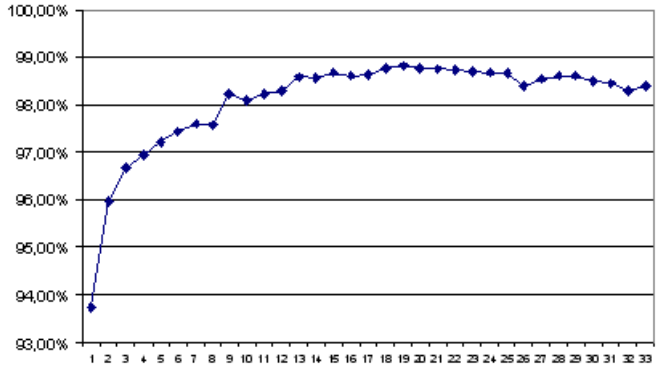


FIG. 7.8 – *Taux de reconnaissance balancé en fonction du nombre d'attributs utilisé pour la discrimination des cellules bénignes et malignes.*

- les SVM ne sont que peu sensibles à la malédiction de la dimensionnalité pour ce problème de discrimination (le nombre d'exemples est suffisant par rapport au nombre d'attributs),
- plusieurs attributs doivent être redondants car le taux de reconnaissance est déjà performant avec seulement 9 attributs sur les 46,
- lorsque peu d'attributs sont utilisés, la majorité sont ceux liés à la texture (voir figure 7.9). Ce résultat est important pour le corps médical car il est reconnu que l'œil humain se base essentiellement sur l'aspect de la texture de la chromatine des cellules pour réaliser la distinction malin/bénin.

L'extraction de l'ensemble de ces attributs étant importante en temps de calcul, il semble logique de chercher à réduire le nombre d'attributs utilisés si le compromis coût d'extraction et taux de reconnaissance reste performant. Dans des travaux futurs notre approche à base de recherche avec tabous (ou plus exactement une variante) presentée

8. Pour accélérer la sélection de modèle, tous les couples (C,γ) ne sont pas testés à chaque ajout ou suppression d'un attributs. La raison est que deux solutions qui partagent un nombre important d'attributs en commun ont des performances quasi-identiques avec les mêmes modèles. Nous ne rentrerons pas plus dans les détails techniques de la méthode utilisées pour ne pas tester tous les modèles.

9. Il n'y pas d'apprentissage dans ce cas et le taux de reconnaissance est fixé à 50%.

10. Une phase SFS et SBS est utilisée à la fin de SFFS pour améliorer les performances des fonctions de décision relatives à des cardinalités peu présentes pendant l'exploration de SFFS.

au chapitre 5.3.2.1 devra être utilisée afin de réduire les coût d'extraction nécessaire à la caractérisation des cellules[11].

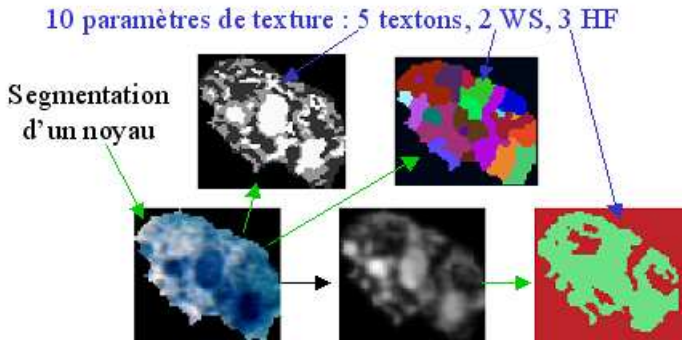


FIG. 7.9 – *Importance des attributs de texture pour la discrimination malin/bénin. Par exemple, lorsque le nombre d'attributs sélectionné est 13, la meilleure fonction de décision exploite: 10 attributs de texture, 2 attributs de forme et 1 de couleur relatifs à la caractérisation du noyau des cellules (voir [*LEZORA00*] pour plus de détails sur la signification des différents attributs).*

7.1.3.2 Classification des types cellulaires

Avant de commenter les résultats obtenus avec des schémas de combinaison de SVM binaires, nous rappelons quelques résultats et remarques relatifs aux travaux de Olivier Lezoray [LEZORA00].

Remarques et résultats préliminaires

Olivier Lezoray montre dans sa thèse que la reconnaissance du type cellulaire est un problème difficile. Il obtient avec un réseau de neurones multi-couche un taux de reconnaissance du type cellulaire de 55,9%. Il montre qu'il est possible d'améliorer le taux de reconnaissance en exploitant une architecture à base de réseaux de neurones nommée MONNA [LEZORA00, LEZORA04]. Cette architecture utilise une décomposition *un-contre-un* et un principe de décodage en cascade. Le taux de reconnaissance est alors de 65,8%. Lorsqu'une sélection d'attributs est réalisée pour chaque réseau de neurones utilisé par l'architecture MONNA le taux de reconnaissance est de 84,55%[12]. Si la nature du système d'apprentissage permet d'augmenter le taux de reconnaissance du type cellulaire, il est constaté que les principales erreurs se situent au niveau de:

- la distinction entre les différents types de polynucléaires,
- la distinction entre les lymphocytes non activés, les lymphocytes activés et les macrophages,
- la distinction entre les macrophages et les mésothéliales.

11. Le fait de devoir réduire le nombre de vecteurs de support est dans ce cas moins important par rapport à celui de réduire le nombre d'attributs.

12. Le taux de reconnaissance (non balancé) malin/bénin, déduit à partir des types cellulaires, est de 98,5%.

Si certaines distinctions sont difficiles à réaliser par les experts lors de l'évaluation des cellules, la constitution de la base suscite certaines interrogations et met en avant l'importance d'équilibrer et d'étoffer le nombre de cellules de chaque classe. Il est également remarqué que le processus décisionnel construit par MONNA lorsqu'une sélection interne des attributs est réalisée ne permet pas de réduire globalement le nombre d'attributs utilisé et donc le coût de la caractérisation des cellules.

Reconnaissance du type cellulaire

Plusieurs protocoles d'apprentissage avec des schémas de combinaison de SVM binaires ont été réalisés. Pour chaque problème binaire, la méthode de Hastie [HASTIE04A] est utilisée pour obtenir une estimation de la probabilité de la classe positive en fonction de la sortie de chaque SVM. Le tableau 7.7 donne les caractéristiques propres à chaque protocole d'apprentissage et le taux de reconnaissance du type cellulaire. La discrimination malin/bénin est déductible du type cellulaire et le taux de reconnaissance malin/bénin correspondant est également donné dans ce tableau.

Les résultats obtenus montrent que les protocoles d'apprentissage ont une influence importante. L'utilisation d'une optimisation multi-modèle du schéma de décomposition ou le choix d'un couple décomposition/décodage approprié permet dans les deux cas d'améliorer significativement le taux de reconnaissance et permet d'obtenir un taux de reconnaissance plus performant que ceux de l'architecture MONNA (sans la phase de sélection d'attributs). L'utilisation conjointe de ces deux approches devrait permettre de produire un processus décisionnel encore plus performant. La sélection des attributs propres à chaque classificateur binaire doit également permettre d'améliorer les performances de l'ensemble, comme cela a été constaté avec MONNA. Les résultats du chapitre 6 laissent supposer que cela peut être réalisé dans le cadre d'une optimisation multi-modèle pour garantir une optimisation du schéma complet de combinaison. De plus, si l'on souhaite également optimiser un compromis entre coût d'extraction et performance en généralisation, un système d'apprentissage capable de réaliser à la fois une sélection globale des attributs tout en réalisant une sélection locale d'attributs propre à chaque sous problème d'apprentissage, doit être défini. Bien que le problème d'optimisation ait une plus grande

décomposition	décodage	modèles $\boldsymbol{\theta}$	Taux TC	Taux MB
un-contre-tous	max $p(\omega_i)$	indentique	64,5%	97,7%
un-contre-tous	max $p(\omega_i)$	individuel	66,3%	97,4%
un-contre-tous	max $p(\omega_i)$	AE	66,9%	97,4%
un-contre-un	Price	indentique	65,0%	97,2%
un-contre-un	Price	individuel	65,9%	97,3%
un-contre-un	Price	AE	68,6%	97,4%
hybride	hybride	individuel	70,4%	97,5%

TAB. 7.7 – *Taux de reconnaissance du type cellulaire (Taux TC) et de la nature maligne ou bénigne (Taux MB) obtenus à partir de différents protocoles d'apprentissage. Pour chaque classificateur multi-classe, le type de décomposition et le principe de décodage peut être différents (voir le chapitre 6 pour plus de précisions).Pour la sélection de modèles (colonne modèles $\boldsymbol{\theta}$) les trois possibilités sont: 1) identique (tous les SVM ont le même modèle $\boldsymbol{\theta}$), 2) individuel (chaque SVM à son propre modèle $\boldsymbol{\theta}$ qui est sélectionné indépendamment des autres) et 3) AE (sélection multi-modèle par AE).*

complexité, un algorithme évolutionnaire du type de celui du chapitre 6 et encore envisageable, même si cela nécessite des adaptations.

Pertinence des exemples de la base

La constitution actuelle de la base de cellules bronchiques semble ne pas être optimale pour réaliser par apprentissage supervisé une bonne catégorisation des types cellulaires (et à moindre mesure de la nature cancéreuse d'une cellule). Nous avons exploité notre méthode qui réalise la sélection d'un sous-ensemble d'exemples pertinents sur cette base et nous avons observé l'évolution du taux d'erreur sur la base de validation en fonction du nombre d'exemples pour les 4 critères de pertinences. La figure 7.10 correspond à cette observation. A nouveau, les critères $[\gamma_{\odot} \searrow]$ et $[\gamma_{\bar{\Gamma}} \searrow]$ sont les plus efficaces. Le taux d'erreur atteint rapidement 35% pour environ seulement 15 % des exemples de la base d'apprentissage. L'observation plus en détail de ces résultats montre que ce sont les exemples de classes minoritaires (en nombre d'exemples) qui sont les plus sélectionnés par notre méthode (environ 95% des mésothéliales dystrophiques, mésothéliomes et mitoses anormales). De plus, peu d'exemples des classes majoritaires sont présents dans l'ensemble final des exemples pertinents (environ 10% de l'ensemble des polynucléaires et lymphocytes). Cependant, le nombre d'exemples dans une classe n'est pas suffisant pour conclure qu'une majorité de ces exemples ne sont pas pertinents (*i.e.* redondants) car certaines classes sont fortement présentes dans l'ensemble final d'exemples pertinents alors qu'elles sont pourtant fortement représentées dans la base intiale (environ 80% des mésothéliales et adénocarcinomes). Globalement la constitution de la base est loin d'être optimale. Pour autant la labellisation de cellules d'un grand nombre d'images a pris un temps conséquent aux experts. Il aurait été préférable de ne labelliser que les cellules pertinentes pour l'apprentissage à partir d'un nombre important d'images plutôt que de labelliser toutes les cellules à partir d'un nombre réduit d'images. Le problème est que les experts n'ont pas accès actuellement à l'information "*cellules pertinentes pour l'apprentissage*".

La création d'une base d'apprentissage à donc un coût élevé et les remarques précédentes conduisent à l'importance de définir des outils pour permettre de réduire les coûts de production des bases d'apprentissage. Nous pensons que nos deux approches pour simplifier les bases d'apprentissage peuvent également permettre de développer de tels outils, si elles sont repensées dans un cadre d'apprentissage semi-supervisé. En effet, ces méthodes ont montré que l'ensemble des exemples productibles à partir d'une base d'images n'ont pas forcément besoin d'être tous labellisés pour permettre la constitution d'un processus décisionnel performant. Le développement de systèmes d'apprentissage qui permettent de réduire les coûts relatifs à la constitution de bases d'apprentissage à partir de bases d'images correspond à un axe de recherche qu'il sera nécessaire de développer.

7.1.4 Prise en compte des relations entre classes

Une analyse plus détaillée de la nature des erreurs propre à chaque problème binaire dans le cas d'une décomposition *un-contre-un* montre que les difficultés de discrimination portent sur les mêmes couples de classes que ceux observés par Olivier Lezoray dans sa thèse. La figure 7.11 illustre ce fait. Il est important de noter que si le principe de décodage peut corriger certaines de ces erreurs, il est également possible, qu'à cause du principe de

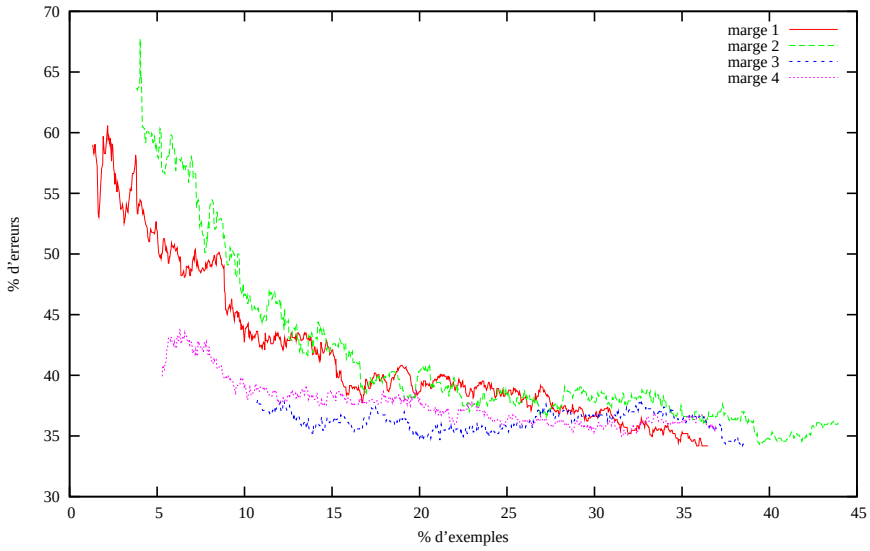


FIG. 7.10 – *Taux de reconnaissance (évalué à partir de la base de validation) en fonction du nombre d'exemples les plus pertinents utilisés pour chacun des quatre critères de pertinence (1 :* $[\gamma \nearrow]$*, 2 :* $[\gamma \searrow]$*, 3 :* $[\gamma_{\odot} \searrow]$ *et 4 :*$[\gamma_{\bar{\Gamma}} \searrow]$*).*

décodage, la classe choisie ne soit pas la bonne alors qu'aucun des classificateurs binaires liés à la bonne classe n'ait réalisé d'erreur de discrimination. Les classificateurs binaires du schéma de décomposition qui sont les moins performants concernent des problèmes à deux classes internes au regroupement de classes bénin ou malin. Ils auront donc peu d'influence sur la classification bénigne ou maligne finale (sauf effet indirect relatif au principe de décodage). Cela explique le bon taux de reconnaissance pour cette discrimination malin/bénin. Cependant, la détermination du type cellulaire n'aide pas forcément à produire une meilleur discrimination malin/bénin que celle produite à partir d'un problème binaire où les classes cellulaires d'un même regroupement (malin ou bénin) ont été préalablement fusionnées. La cause principale est liée au fait que la méthode d'optimisation utilisée n'a aucune connaissance de la signification sémantique de certains regroupements possibles entre les classes, ainsi que le fait que les coûts relatifs à tous les types d'erreurs ne sont

		Discrimination entre deux types cellulaires	
Polynucléaires neutrophiles Polynucléaires éosinophiles	66,8%		
Mésothéliales Mésothéliales dystrophiques	75,2%		
Polynucléaires altérés Lymphocytes non activés	85,0%	Lymphocytes non activés Lymphocytes activés	87,0%
Lymphocytes activés Macrophages	86,3%	Mésothéliales Mésothéliome	97,7%
Mésothéliales dystrophiques Adénocarcinome	86,3%	Mésothéliales dystrophiques Mésothéliome	100,0%

FIG. 7.11 – *Taux de reconnaissance relatifs à plusieurs classificateurs réalisant la discrimination de deux classes dans une décompositions* un-contre-un *(La sélection de modèle est individuelle).*

pas les mêmes. Il est donc actuellement préférable d'avoir deux classificateurs différents pour ces deux problèmes d'identification.

Pour autant, il semble logique que la connaissance du statut malin ou bénin aide à la détermination du type cellulaire car il simplifie le nombre de classes à prendre en compte dans chaque regroupement. Réciproquement, la connaissance du type cellulaire doit aider à la distinction malin/bénin. Nous pensons que pour prendre en compte ces relations, une solution possible est de définir un schéma de combinaison et un principe de décodage qui prend en compte la hiérarchie existante entre les classes. La figure 7.12 donne une illustration de ce que pourrait être cette hiérarchisation des classes des types cellulaires. La structure de ce graphe induit plusieurs types de décomposition, ainsi qu'un principe de décodage hybride qui est global ou en cascade suivant la localisation dans le graphe. Notre schéma de décomposition/décodage hybride devra donc être étendu pour pouvoir prendre en compte ce type de structure. Le système d'apprentissage doit dans ce cas réaliser une optimisation multi-modèle, mais également multi-objectif (les taux de reconnaissance en différents points du graphe et en particulier ceux relatifs aux deux problèmes de discrimination type cellulaire et malin/bénin). L'exploitation de la notion de *front de Pareto* serait une voie possible pour réaliser cette optimisation.

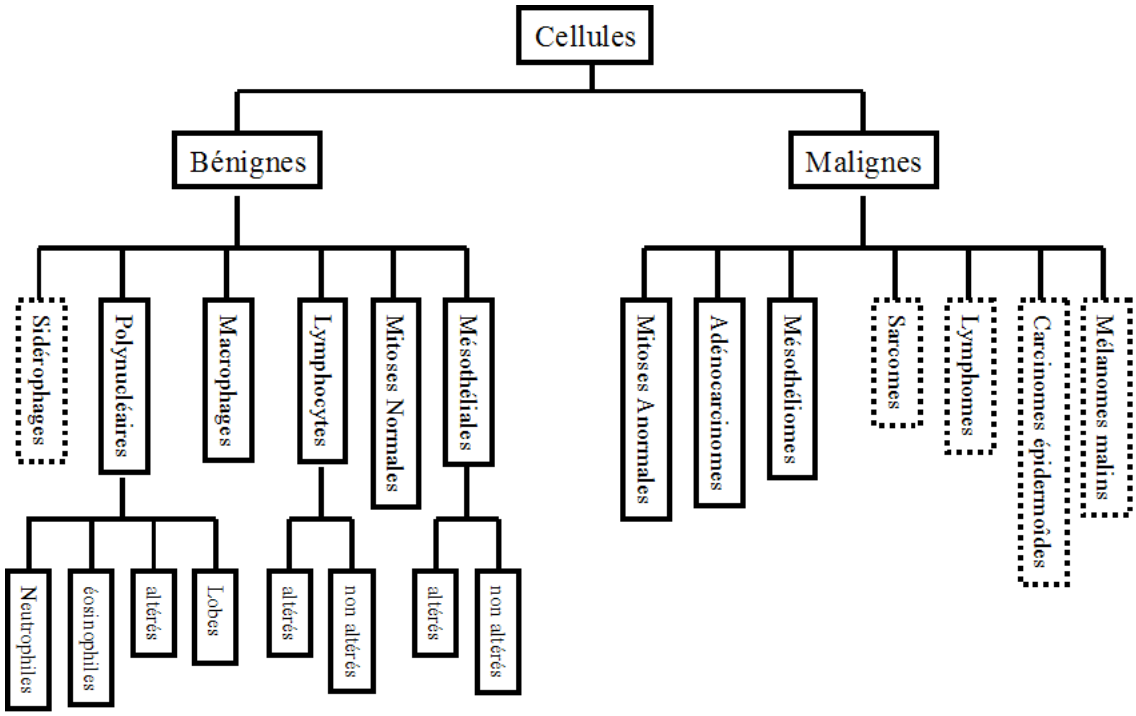


FIG. 7.12 – *Illustration hiérarchique des relations entre les classes et les regroupements de classes (Les noms qui sont entourés par un rectangle en pointillé sont ceux des types cellulaires qui ne sont pas actuellement présents dans la base étudiée).*

L'utilisation d'une telle représentation hiérarchique permettrait également de ne pas avoir à préciser la fonction de coût entre les différents types d'erreurs lorsqu'on cherche à prendre en compte leur différentes natures. En effet, confondre un polynucléaire neutrophile avec un polynucléaire éosinophile est moins grave que de confondre un polynucléaire neutrophile avec un macrophage si la cellule était réellement un polynucléaire neutrophile. Mais quantifier exactement ce coût est problématique, en particulier lorsque le nombre de

classes est important. L'optimisation d'un schéma de décomposition incluant la prise en compte des relations entre les classes ne nécessite par contre que la définition de cette organisation.

Définir un système d'apprentissage capable de réaliser ce type d'optimisation ouvre des perspectives intéressantes car un grand nombre de problèmes multi-classes ont des relations sémantiques plus ou moins marquées entre leurs classes. L'apprentissage automatique de ces relations à partir des données peut également constituer un axe de recherche intéressant à la frontière entre l'apprentissage supervisé et non supervisé.

7.2 Expertise automatique de la qualité des images

7.2.1 Problématique

Toute personne qui cherche à mettre au point une métrique de qualité (qui intègre ou non des caractéristiques de vision bas niveau du système visuel humain–SVH) se retrouve vite confrontée à la même problématique : le calcul de la note finale. Généralement, le développement d'une métrique de qualité s'effectue selon un schéma qui se décompose en deux phases : 1) une transformation colorimétrique de façon à obtenir un triplet de coordonnées décorrélées et 2) une décomposition de ces nouvelles coordonnées en canaux perceptuels (spatiaux et fréquentiels). Ensuite, un calcul d'erreur dans le domaine fréquentiel et spatial est réalisé pour chaque canal ainsi obtenu. Finalement, la note finale est obtenue en moyennant ces erreurs selon la métrique de Minkowski (pondération des erreurs suivant leur intensité).

Cependant, de récentes études [WANG05] ont mis en évidence que cette formulation de la sommation n'est pas adéquate pour le calcul de la note finale, même si elle est largement utilisée par la communauté. En effet, il est possible d'obtenir la même valeur pour deux images dégradées alors que d'un point de vue qualité visuelle les deux images sont très différentes (*cf.* figure 7.13). Ceci est principalement dû à l'hypothèse sur laquelle repose l'utilisation de cette sommation, à savoir une indépendance des signaux. Or, cette hypothèse n'est pratiquement jamais vérifiée lorsque l'on utilise une décomposition en signaux perceptuels. Dans ce cas, la sommation de Minkowski ne permet pas de prendre en compte la disparité visuelle des images.

La nécessité d'aboutir à une note provient du besoin de mesurer la corrélation existante entre les notes ainsi obtenues et les notes distribuées par l'utilisateur lors de tests psychophysiques. Dans ces derniers, l'observateur est amené à attribuer une note à l'image qui lui est présentée sur un écran. La moyenne de toutes les notes obtenues pour une même image est ensuite calculée; on obtient ainsi le MOS (*Mean Opinion Score*) qui sera comparé avec la note obtenue par la métrique. La phase de notation de la qualité des images par un observateur suit un protocole strict défini dans la recommandation UIT-R B500.10

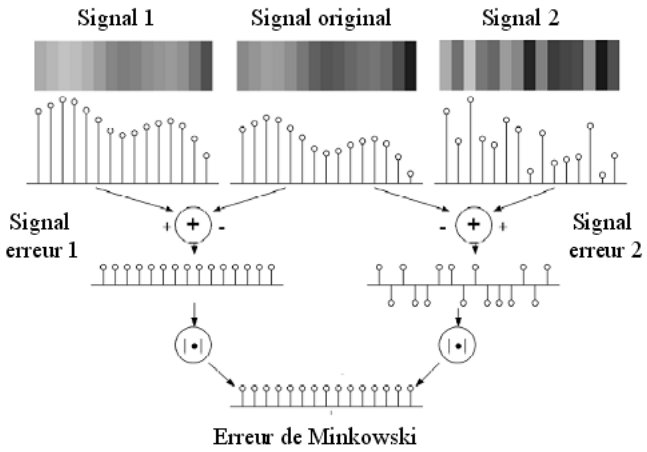


FIG. 7.13 – *Illustration des limites de la sommation de Minkowski.*

[UIT-RR00]. Ainsi, plus la corrélation entre les deux notations est grande, plus la métrique développée est pertinente.

Néanmoins, nous avons constaté précédemment que l'utilisation de la sommation de Minkowski introduit un biais important dans le calcul de la note de qualité finale par la métrique développée. Dans la recommandation de l'UIT, nous observons que les observateurs doivent attribuer une note selon une échelle clairement définie, qui contient les cinq classes. La tableau 7.8 contient ces différentes classes.

TAB. 7.8 – *Echelle de qualité retenue dans la recommandation UIT-R.*

Qualité	
5	Excellente
4	Bonne
3	Relativement bonne
2	Mauvaise
1	Très mauvaise

A la lecture de ce tableau, nous pouvons constater que l'observateur humain effectue une classification de la qualité, notée Q_{OS} (*Quality Opinion Score*), selon cinq classes. Le qualité visuelle d'une image, notée Q_{MOS} (*Quality Mean Opinion Score*), est alors obtenue "par moyennage des numéros de classe".

Nous proposons alors de développer une métrique de qualité basée sur un processus de classification de façon à respecter au mieux le processus de classification utilisé lors des tests psychophysiques. L'objectif est alors d'éviter par la même occasion la limitation posée par l'utilisation de la sommation de Minkowski. Ainsi, au lieu de calculer une note finale de qualité par simple sommation, la méthode aboutie à une classification de la qualité de l'image selon les recommandations de l'UIT.

7.2.2 Caractérisation du système visuel humain

De façon à définir un expert informatique de la qualité, nous avons conçu un vecteur d'attributs issus de caractéristiques du SVH pour réaliser la description des images. Chaque image est décrite par un vecteur composé de 38 attributs $(x_i)_{i\in[1,\ldots,38]}$ correspondant à trois catégories perceptuelles :

1. Sensibilité aux contrastes à travers 31 critères de contraste $((x_i)_{i\in[1,\ldots,31]})$.
2. Sensibilité aux structures à travers 3 critères structurels $((x_i)_{i\in[32,\ldots,34]})$.
3. Sensibilité aux couleurs à travers 4 critères de couleur $((x_i)_{i\in[35,\ldots,38]})$.

Les détails relatifs à ces trois catégories perceptuelles sont donnés dans les sections suivantes.

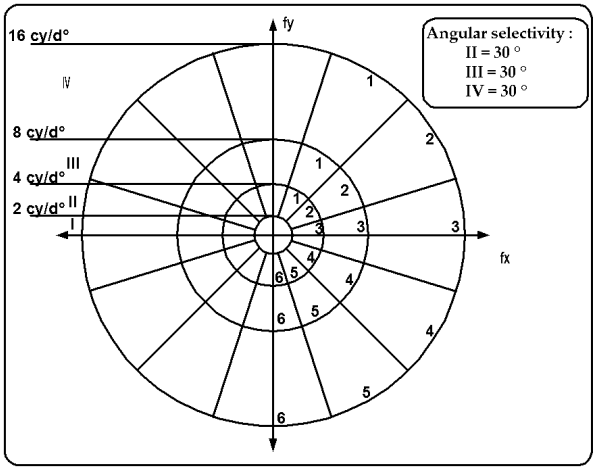


FIG. 7.14 – *Modèles de perception de la luminance selon Daly [*DALY93*].*

7.2.2.1 Critères de contraste local à bande limitée

Pour une bonne définition du contraste, il est nécessaire que les résolutions radiales et fréquentielles soient prises en compte. Peli [PELI90] a introduit le contraste à bande limitée répondant à ce critère. Le constrate développé est local afin d'exprimer la sensibilité de l'observateur aux changements de luminance lorsque la luminance moyenne locale varie. Il est également à bande limitée car la perception des dégradations dépend directement de leur localisation spectrale.

Pour une décomposition telle que celle de la figure 7.14 où les canaux sont sélectifs radialement et angulairement, le contraste à bande limitée est donné par :

$$c_{k,\theta,j}(u,v) = \frac{a_{k,\theta,j}(u,v)}{\sum_{k'=0}^{i} \sum_{\theta'=0}^{n_{k'}} a_{k',\theta',j}(u,v)} \tag{7.5}$$

où $a_{k,\theta,j}(u,v)$ et $c_{k,\theta,j}(u,v)$ représentent respectivement la luminance (voir l'annexe B pour les détails relatifs à la détermination de la luminance) et le contraste au point (u,v) du $j^{\text{ème}}$ canal radial, du $\theta^{\text{ème}}$ secteur angulaire et de la $k^{\text{ème}}$ bande radiale ($n_{k'}$ désigne le nombre de secteurs angulaires dans la $k'^{\text{ème}}$ bande radiale). Le dénominateur de (7.5) représente la luminance moyenne locale correspondant à tous les canaux de support spectral inférieurs à celui du $j^{\text{ème}}$ canal. Le nombre total de valeurs différentes pour le triplet (k,θ,j) est de 31, notons (k_i,θ_i,j_i) l'un de ces triplets avec $i \in [1,\ldots,31]$.

Chaque attribut x_i, relatif à un triplet (k_i,θ_i,j_i) donné, est obtenu à partir de la moyenne des critères de constraste déterminée sur l'ensemble des points de l'image:

$$x_i = \frac{1}{|u|\,|v|} \sum_u \sum_v c_{k_i,\theta_i,j_i}(u,v) \tag{7.6}$$

où $|u|\,|v|$ désigne la taille de l'image.

7.2.2.2 Critères structuraux

Les 31 caractéristiques de contraste prennent en compte la sensibilité de l'observateur humain à la variation de luminance dans l'image et les effets de masquage qu'elle peut

engendrer. Afin de prendre en compte les caractéristiques structurelles de l'image, les critères définis dans la métrique de Wang et Bovik [WANG02] ont été utilisés. Il s'agit de mesurer: 1) la distortion de luminance (x_{32}), 2) la distortion de contraste (x_{33}) et 3) d'effectuer une comparaison de structure des images (x_{34}). Pour calculer ces trois critères, les auteurs proposent de représenter l'image dans un espace image où chaque distortion calculée peut être interprétée comme un vecteur resultant d'une somme vectorielle avec le vecteur représentant l'image originale. Dans cet espace, les deux premières distortions (luminance et contraste) délimitent un plan adapté au vecteur de l'image originale. La rotation de cet espace selon un angle peut être vue comme un changement de structure de l'image.

7.2.2.3 Critères de couleur

A ces trois nouveaux attributs, nous avons ajouté quatre descripteurs couleurs qui sont définis par : 1) la mesure de chrominance locale (x_{35}), 2) la dispersion colorimétrique locale (x_{36}), 3) la MAE (Mean Average Error) (x_{37}) et 4) le PSNR (Peak Signal to Noise Ratio) couleur [PLATON00] (x_{38}). Les deux premiers attributs intègrent des caractéristiques de l'attention visuelle [TRÉM97].

7.2.3 Un expert informatique de la qualité

Nous avons naturellement exploité des SVM afin de mettre au point une métrique de qualité par apprentissage.

7.2.3.1 Modélisation de l'expert

Le problème de la définition d'un expert informatique tel qu'il a été formulé correspond à un problème d'apprentissage supervisé à 5 classes. Plusieurs schémas de combinaison de classificateurs existent pour aborder ce problème avec des SVM (*cf.* chapitre 6). Nous avons utilisé deux schémas : 1) *un-contre-tous* et 2) RO (*Rank Ordering*), permettant de prendre en compte l'ordre existant entre les cinq classes finales. Soit $\boldsymbol{\psi}_i = \{\psi_{i,1}, \ldots, \psi_{i,n_c}\}$ un schéma de décomposition transformant un problème de n_c classes en un problème binaire avec $\psi_{i,j} \in \{+1, -1\}$ (*cf.* chapitre 6).

Les tableaux 7.9 et 7.10 représentent les décompositions en problèmes binaires utilisées respectivement dans chacun des deux schémas[13]. En analysant ces tableaux, les transformations $\boldsymbol{\psi}_1$ et $\boldsymbol{\psi}_5$ du schéma nommé RO *un-contre-tous* sont identiques aux transformations $\boldsymbol{\psi}_1$ et $\boldsymbol{\psi}_4$ du second schéma. Les différences sont essentiellement concentrées sur les autres transformations. Contrairement au schéma *un-contre-tous*, l'information sur le rang des classes est conservée dans le second schéma (i.e. $\forall c_1,c_2 : t_{i,c_1} > t_{i,c_2} \rightarrow c_1 > c_2$). De plus, les fonctions dicriminantes correspondant aux transformations $\boldsymbol{\psi}_2$,$\boldsymbol{\psi}_3$ où $\boldsymbol{\psi}_4$ dans le schéma *un-contre-tous* sont plus compliquées à réaliser lorsque des images de très bonnes et de très mauvaises qualités sont intégrées dans la même classe et utilisées pour identifier des images de relativement bonne qualité.

13. A une transpostion près, il s'agit du même type de représentation que celle du chapitre 6.

classe	ψ_1	ψ_2	ψ_3	ψ_4	ψ_5
5	+1	-1	-1	-1	-1
4	-1	+1	-1	-1	-1
3	-1	-1	+1	-1	-1
2	-1	-1	-1	+1	-1
1	-1	-1	-1	-1	+1

TAB. 7.9 – *Transformation en problèmes binaires utilisée dans le schéma un-contre-tous.*

classe	ψ_1	ψ_2	ψ_3	ψ_4
5	+1	+1	+1	+1
4	-1	+1	+1	+1
3	-1	-1	+1	+1
2	-1	-1	-1	+1
1	-1	-1	-1	-1

TAB. 7.10 – *Transformation en problèmes binaires utilisée dans le second schéma.*

La transformation en problèmes binaires est la première partie du schéma de combinaison. En effet, une décision finale doit être prise à partir de toutes les fonctions de décision. Plusieurs stratégies de combinaison peuvent être adoptées afin d'obtenir la décision finale (*cf.* chapitre 6). Le critère du vote majoritaire est celui que nous avons utilisé dans cette application, car il s'applique facilement à la décomposition RO.

7.2.3.2 Mesure des performances

Base de données

Deux bases sont construites (désignées par **QIjpeg2000** dans l'annexe A) en utilisant les 227 versions compressées des 25 images originales de la base LIVE [LABORA02]. La compression a été réalisée avec le standard de compression JPEG2000. La première base est constituée de 116 images compressées et définit la base d'apprentissage utilisée lors de la phase d'apprentissage de l'expert informatique. La seconde base regroupe les 111 images restantes et définit la base de validation utilisée pour évaluer les performances de notre expert.

La figure 7.15 contient cinq images représentatives de chacune des classes de qualité. Pour chaque image, nous avons relevé le Q_{MOS} des observateurs ainsi que la note moyenne, la note minimale et la note maximale attribuées par les observateurs.

Pour chacune des images compressées, un vecteur contenant 38 attributs a été généré selon les principes définis précédemment. Les 25 images originales ont été utilisées en tant que références dans le calcul des attributs caractérisant les 227 images obtenues à partir d'elles pour différents niveaux de compression avec JPEG2000.

En même temps, respectivement 29 et 25 observateurs ont noté les images de la première et de la seconde base[14]. A partir de ces notations, le Q_{MOS} est calculé. Le tableau

14. Les notes attribués sont disponibles avec la base LIVE.

(a) Q_{MOS}= 5; $\bar{Q}_{\mathrm{OS}}$= 4,68; $Q_{\mathrm{OS}}^{\mathrm{max}}$= 5; $Q_{\mathrm{OS}}^{\mathrm{min}}$= 3

(b) Q_{MOS}= 4; $\bar{Q}_{\mathrm{OS}}$= 4,16; $Q_{\mathrm{OS}}^{\mathrm{max}}$= 5; $Q_{\mathrm{OS}}^{\mathrm{min}}$ = 3

(c) Q_{MOS}= 3; $\bar{Q}_{\mathrm{OS}}$= 2,78; $Q_{\mathrm{OS}}^{\mathrm{max}}$= 4; $Q_{\mathrm{OS}}^{\mathrm{min}}$ = 1

(d) Q_{MOS}= 2; $\bar{Q}_{\mathrm{OS}}$= 1,61; $Q_{\mathrm{OS}}^{\mathrm{max}}$= 3; $Q_{\mathrm{OS}}^{\mathrm{min}}$ = 1

(e) Q_{MOS}= 1; $\bar{Q}_{\mathrm{OS}}$= 1,22; $Q_{\mathrm{OS}}^{\mathrm{max}}$= 3; $Q_{\mathrm{OS}}^{\mathrm{min}}$ = 1

FIG. 7.15 – *Panel représentatif des images présentes dans la base LIVE avec leur* Q_{MOS} *associé ainsi que la note moyenne* ($\bar{Q}_{\mathrm{OS}}$)*, la note maximale* ($Q_{\mathrm{OS}}^{\mathrm{max}}$) *et la note minimale* ($Q_{\mathrm{OS}}^{\mathrm{min}}$) *attribuées par les observateurs.*

7.11 présente le pourcentage de la classification de la qualité des images compressées en fonction de l'échelle de notation telle que définie par l'UIT.

Q_{MOS}	1	2	3	4	5
Base d'apprentissage	12,9%	39,7%	25,9%	16,4%	5,1%
Base de validation	13,5%	36,1%	14,4%	24,3%	11,7%

TAB. 7.11 – *Pourcentage des images classifiées par rapport à l'échelle de notation de l'UIT.*

Afin de mesurer l'efficacité de notre expert informatique, trois mesures de cohérence M ont été introduites à partir de trois fonctions de perte LO :

$$M_a = 1 - \frac{1}{m}\sum_{i=1}^{m} LO_a\left(D(i),Q_{\mathrm{MOS}}(i)\right) \tag{7.7}$$

où $D(i)$ représente la qualité telle que définie par l'expert informatique pour l'image i et $Q_{\mathrm{MOS}}(i)$ désigne le MOS associé. $a \in \{1,2,3\}$ correspond à l'indice de la fonction de perte (*LOss function*) utilisée :

$$LO_1\left(y_1,y_2\right) = \begin{cases} 0 & \text{si } y_1 = y_2 \\ 1 & \text{sinon} \end{cases} \tag{7.8}$$

$$LO_2\left(y_1,y_2\right) = \begin{cases} 0 & \text{si } |y_1 - y_2| \leq 1 \\ 1 & \text{sinon} \end{cases} \tag{7.9}$$

$$LO_3\left(y_1,y_2\right) = \begin{cases} 0 & \text{si } y_1 = y_2 \\ \frac{m}{m_{y_2}} & \text{sinon} \end{cases} \tag{7.10}$$

où m_{y_2} correspond au nombre d'images de la classe y_2 dans la base de référence. M_1 est une mesure classique du taux de reconnaissance. M_2 est une mesure de cohérence de rang pour la prédiction de la qualité. Illustrons l'intérêt de cette mesure par un exemple : supposons que la classe "Excellente" ou "Assez bonne" est la classe prédite par l'expert informatique alors que le Q_{MOS} désigne l'image comme étant de la classe "Bonne", cette différence de classification est tolérable; supposons maitenant que la classe "Mauvaise' ou "Très mauvaise" est la classe prédite, cette différence est par contre intolérable. M_3 est une mesure prenant en compte la proportion relative à la représentation de chacune des cinq classes dans la base de données (*i.e.* taux de reconnaissance balancé). Ceci permet de vérifier que les classes peu représentées ne sont pas négligées par le classificateur.

Variance intra-observateur

Nous avons mesuré pour chaque observateur la stabilité de celui-ci par rapport au Q_{MOS} en se basant sur la notation Q_{OS} lui correspondant. Cette notation est utilisée en tant que fonction de décision D par rapport aux trois mesures de cohérence M. Les tableaux 7.12 et 7.13 montrent respectivement les statistiques de confiance que l'on peut accorder à un observateur par rapport à l'observateur moyen représenté par Q_{MOS}.

Ces résultats montrent que l'opinion des observateurs est très variable et ceci quelle que soit la base utilisée. La mesure M_2 montre que la divergence de classification n'excède que très rarement une classe.

	moyenne	min	Max
M_1	0,558 $\pm$ 0,078	0,405	0,698
M_2	0,989 $\pm$ 0,019	0,914	1,000
M_3	0,594 $\pm$ 0,057	0,462	0,669

TAB. 7.12 – *Statistiques des mesures de cohérence pour la base d'apprentissage.*

	moyenne	min	Max
M_1	0,529 $\pm$ 0,103	0,324	0,712
M_2	0,975 $\pm$ 0,026	0,909	1,000
M_3	0,550 $\pm$ 0,090	0,396	0,706

TAB. 7.13 – *Statistiques des mesures de cohérence pour la base de validation.*

Performances des experts

L'expert informatique est construit en utilisant les deux schémas de combinaison de SVM définis précédemment. Afin de mesurer l'influence des attributs, trois expériences ont été menées :

1. Les 38 attributs sont tous utilisés.
2. Seuls les 31 attributs issus du calcul du masquage à bande limitée ont été sélectionnés.
3. Une recherche des attributs les plus significatifs à partir de l'algorithme *best-first-search* [PEARL84].

Etant donné que les bases sont relativement petites, la sélection du modèle pour chaque SVM binaire (noyau gaussien) est réalisée selon une mesure de validation croisée *leave-one-out*. Nous avons donc utilisé notre méthode d'estimation rapide de l'erreur de *leave-one-out* (*cf.* chapitre 4) pour réduire les temps nécessaires à cette sélection de modèles, en particulier lorsqu'une sélection d'attributs doit être réalisée. Pour chacun des six experts informatiques, les mesures de cohérence ont été calculées à partir des bases d'apprentissage et de validation.

Les résultats obtenus sont présentés dans les tableaux 7.14 et 7.15. Lorsque l'on compare les résultats des tableaux 7.12 et 7.13 aux précédents, nous constatons que les experts que nous avons développés sont de meilleure qualité que le comportement d'un observateur pris individuellement, en particulier pour les mesures M_1 et M_3. Les experts informatiques suivent fortement le comportement moyen des observateurs lorsque l'on examine la mesure M_2. De même, la mesure M_3 permet de mettre en évidence que les experts informatiques ne négligent pas les classes peu représentées.

Les résultats du tableau 7.15 montrent que le schéma RO est plus sensible à l'information de rang que le schéma *un-contre-tous* à travers la mesure M_2. Ces résultats sont cohérents avec les raisons du choix de ce type de décomposition et ils illustrent l'importance d'un schéma de décomposition suivant les relations sémantiques entre les classes.

A partir du tableau 7.15, les valeurs de la seconde colonne montrent l'importance des attributs relatifs aux critères de contraste de l'image et ceci indépendamment du type de

	38 caractéristiques		31 caractéristiques		Sélection de caractéristiques	
	UCT	RO	UCT	RO	UCT	RO
M_1	0,931	0,931	0,922	0,914	0,957	0,965
M_2	0,983	1,000	0,983	1,000	0,991	1,000
M_3	0,873	0,881	0,840	0,863	0,929	0,906

TAB. 7.14 – *Mesures de cohérence calculées sur la base d'apprentissage. Les colonnes UCT et RO désignent respectivement les décompositions* Un-Contre-Tous *et* Rank Ordering.

	38 caractéristiques		31 caractéristiques		Sélection de caractéristiques	
	UCT	RO	UCT	RO	UCT	RO
M_1	0,801	0,801	0,793	0,784	0,693	0,639
M_2	0,982	1,000	0,991	1,000	0,946	1,000
M_3	0,747	0,781	0,741	0,775	0,672	0,598

TAB. 7.15 – *Mesures de cohérence calculées sur la base de validation. Les colonnes UCT et RO désignent respectivement les décompositions* Un-Contre-Tous *et* Rank Ordering.

décomposition. La valeurs de la première colonne montrent que les attributs relatifs aux critères de structure et de couleur permettent une légère amélioration des mesures de performance. Néanmoins, cette remarque est à mettre en opposition avec le faible nombre d'images présentes dans les bases.

La sélection des caractéristiques ne permet pas de réaliser une augmentation des performances. De plus, nous remarquons que le taux de reconnaissance est plus élevé sur la base d'apprentissage que sur la base de validation. Ceci peut être expliqué par un surapprentissage dû au faible nombre d'images présentes dans la base, malgré une sélection de modèles réalisée à partir d'une estimation de l'erreur de *leave-one-out*. Les exploitations de la technique de *boosting* pour la sélection de modèle et d'autres algorithmes pour la sélection des caractéristiques devront être réalisées pour obtenir des résultats plus pertinents sur la possibilité de n'avoir à exploiter qu'un ensemble réduit de ces caractéristiques.

7.3 Conclusion

Nous avons abordé deux domaines d'application relatifs au traitement et à l'analyse d'images. Certaines de nos approches pour définir des systèmes d'apprentissage performants ont été utilisées dans ces deux domaines.

Segmentation d'images de microscopies médicales

Nous avons montré qu'il est possible de définir un classificateur de pixels rapide et performant à partir de notre système de simplification des fonctions de décision par prototypage. Ce système d'apprentissage permet également de choisir des espaces hybrides couleur adaptés à une classification rapide et efficace des pixels. L'exploitation d'un schéma

de décomposition *un-contre-un* permet également d'optimiser le compromis complexité / efficacité. Le principe de décodage de Price sélectionné permet de réaliser l'estimation de la probabilité de présence du noyau, du cytoplasme et du fond pour chaque pixel avec des temps calculatoires relativement faibles. A partir de ces estimations, nous avons défini un schéma complet de segmentation d'images de cellules, ainsi qu'une mesure de la qualité de la segmentation qui puisse prendre en compte l'adéquation de la segmentation automatique avec la segmentation de référence. Nous avons optimisé l'ensemble des paramètres libres du schéma de segmentation proposé à partir d'une recherche avec tabous. Les résultats expérimentaux obtenus montrent que ce schéma de segmentation permet d'augmenter la qualité de l'extraction des objets cellulaires dans les images par rapport à l'exploitation directe de la classification de pixels. L'examen visuel des segmentations obtenues confirme la bonne qualité des segmentations produites.

Tri cellulaire

Nous avons montré que les SVM permettent de réaliser une discrimination efficace des cellules malignes et bénignes avec une légère augmentation du taux de reconnaissance par rapport aux résultats de référence. Cette discrimination peut être réalisée majoritairement à partir d'attributs de texture. La reconnaissance du type cellulaire dépend fortement du protocole d'apprentissage utilisé, en particulier que nos méthodes hybride et à base d'algorithme évolutionnaire permettent d'avoir un meilleur taux de reconnaissance car elle sont à même de produire des schémas de combinaison plus performants sans nécessiter l'utilisation d'un nombre très important de décompositions. Nous avons également montré que la base d'apprentissage n'est pas optimale par rapport à sa constitution et que cela peut réduire les performances en généralisation. Cette étude a permis de développer deux réflexions. La première porte sur la nécessité de développer des outils semi-supervisés pour réduire les coûts en temps de production des bases d'apprentissage et permettre d'accroître les possibilités de réaliser des inductions performantes à partir d'elles. La seconde porte sur la définition de systèmes d'apprentissage plus performants pour prendre en compte les relations entre les classes et regroupements. Dans les deux cas nous montrons que les méthodes développées dans cette thèse peuvent constituer des points de départ à ces problématiques. Nous signalons également que la réduction des coûts d'extraction des caractéristiques propres aux cellules, tout en garantissant un taux de reconnaissance élevé du type cellulaire, est un problème de sélection multi-modèle et multi-objectif très difficile. Les différentes idées et approches proposées dans cette thèse vont dans la direction d'une solution à ce problème, mais demanderont un travail de recherche supplémentaire pour être concrétisées.

Expertise automatique de la qualité des images

Nous avons montré que l'analyse de la qualité des images peut être abordée comme un problème d'apprentissage supervisé en traitant le problème de la détection automatique de la qualité visuelle des images couleur compressées avec la norme JPEG2000. Nous avons réalisé les choix nécessaires à la caractérisation de ces images. L'ensemble des résultats obtenus montre que l'expert informatique produit par notre système d'apprentissage est capable de simuler le comportement moyen d'un panel d'observateurs avec une grande fidélité. Ces résultats justifient également l'importance du choix d'une caractérisation basée sur la sensibilité au contraste du système visuel humain. Nous montrons à nouveau, à

partir de ces travaux, que la définition d'un schéma de décomposition peut être fortement liée aux relations sémantiques entre les classes.

CHAPITRE 8

CONCLUSION GÉNÉRALE ET PERSPECTIVES

Sommaire

8.1 Apports de cette thèse

8.1.1 Définition de systèmes d'apprentissage performants

Les apports de cette thèse concernent essentiellement le développement de nouveaux systèmes d'apprentissage supervisé performants et correspondant à des problèmes d'optimisation difficiles tels qu'ils ont été définis en section 1.3. Pour cela, de nouvelles heuristiques ou de nouvelles formulations de méta-heuristiques ont été définies. Celles-ci exploitent essentiellement des principes d'inférence ayant de solides bases théoriques ainsi que des méthodes éprouvées pour mesurer expérimentalement les capacités de généralisation d'un processus décisionnel. Les détails des apports dans ce domaine sont les suivants.

Il est montré qu'il est possible d'estimer de façon rapide et efficace l'erreur de généralisation avec des techniques de validation croisée lorsque l'entraînement d'un SVM est réalisé avec l'algorithme SMO et que les valeurs d'initialisation de cet algorithme sont correctement choisies. Pour cela, une nouvelle méthode d'*alpha seeding* est proposée. L'importance de changer le critère d'arrêt de SMO pour amplifier l'accélération de la méthode d'*alpha seeding* est mise en évidence et une procédure permettant de choisir la valeur de ce critère est proposée. Plusieurs expérimentations avec un noyau gaussien montrent l'efficacité de la méthode pour trois procédures de validation croisée. Dans [LEBRUN06B], nous proposons également des résultats obtenus avec un noyau polynomial pour l'estimation rapide de l'erreur *leave-one-out*. L'ensemble de la méthode proposée permet de réduire très significativement les coûts calculatoires des systèmes d'apprentissage à base de SVM lorsqu'ils doivent sélectionner, parmi un ensemble de processus décisionnels potentiels, le plus performant.

Il est montré également que la sélection d'un sous-ensemble d'exemples pertinents dans une base de données permet, dans la majorité des cas, d'améliorer les performances en généralisation des SVM. De plus, cela permet de réduire significativement la complexité des fonctions de décision produites. Un algorithme générique est proposé pour

réaliser cette sélection, ainsi que quatre critères pour définir une relation d'ordre relative à la notion de pertinence des exemples. Il est montré que deux de ces critères sont plus performants si le critère d'arrêt de notre méthode générique est modifié pour forcer la sélection d'un ensemble plus réduit d'exemples. Un critère d'arrêt prématuré différent est proposé pour chacun des deux critères de pertinence. A partir de ces résultats, nous proposons un nouveau système d'apprentissage en cascade capable de réaliser une sélection de modèles qui inclut la sélection d'un sous-ensemble réduit d'exemples pertinents et des hyper-paramètres des SVM avec un noyau gaussien. Nous montrons expérimentalement qu'il est possible de réduire la complexité des fonctions de décision produites tout en augmentant les capacités de généralisation de ces fonctions.

Nous proposons ensuite une tout autre approche afin de réduire la complexité des fonctions de décision. Elle est basée sur une méthode de prototypage de la base d'apprentissage à partir d'un algorithme de quantification vectorielle afin de simplifier les bases d'entraînement d'un SVM. Nous définissons un nouveau critère de qualité qui correspond à un compromis entre les capacités de généralisation d'un processus décisionnel et la complexité de ce processus. Nous montrons qu'il est possible d'optimiser ce critère de qualité à partir d'une recherche avec tabous qui réalise une sélection multi-modèle comprenant le niveau de simplification de la base d'entraînement, la sélection des attributs pertinents et les hyper-paramètres des SVM. Nous proposons une formulation propre à ce problème de la recherche avec tabous et définissons notamment des stratégies de diversification et d'intensification efficaces pour la sélection du multi-modèle optimal. Les expérimentations montrent qu'il est possible de réaliser cette optimisation complexe dans des temps raisonnables et que les processus décisionnels produits réalisent un compromis généralisation/complexité très performant. L'ensemble de cette approche permet de définir un système d'apprentissage qui a la particularité de produire des processus décisionnels performants avec une complexité calculatoire réduite. L'importance de cette réduction peut être, dans une certaine proportion, définie par l'utilisateur.

Nous montrons enfin qu'il est possible d'améliorer les performances en généralisation des schémas de combinaison de classificateurs binaires sans augmenter significativement la complexité calculatoire du processus décisionnel produit. Deux approches sont proposées pour réaliser cette amélioration. La première correspond à prendre en compte les relations de plus ou moins grande similarité entre les classes à travers la définition d'un schéma hybride de combinaison qui incorpore dans son principe de décodage (et de décomposition) l'existence de ces relations. La seconde correspond à vérifier, à partir de la production de contre-exemples, que les méthodes de sélection de modèles communément utilisées pour l'optimisation des performances de schémas de combinaison de SVM ne sont pas optimales. Nous proposons pour cela une autre formulation du problème d'optimisation à réaliser qui correspond à une sélection de modèles dans un espace de grande complexité. Nous donnons ensuite la définition des opérateurs d'un algorithme évolutionnaire permettant de réaliser l'optimisation d'un tel problème. Expérimentalement, nous montrons que la sélection de modèle réalisée permet d'augmenter les capacités en généralisation des schémas de combinaison et que cet effet est d'autant plus important que le nombre de classes du problème d'apprentissage est élevé. L'exploitation de ces deux types d'approches permet la définition de systèmes d'apprentissage qui ont des capacités de généralisation accrues pour des problèmes d'apprentissage avec un nombre de classes élevé.

8.1.2 Systèmes d'apprentissage adaptés aux problèmes de traitement et d'analyse d'images

Les apports de cette thèse ont concerné également deux domaines d'applications correspondant à la définition d'un système d'analyse d'images pour le tri cellulaire des cellules cancéreuses et l'expertise automatique de la qualité visuelle des images compressées. Les apports dans ces domaines sont de deux natures : 1) la mise en œuvre de certaines des techniques proposées précédemment dans le cadre de la définition de systèmes d'apprentissage adaptés aux problèmes de traitement et d'analyse d'images abordés. 2) des outils et analyses propres aux domaines des applications traitées.

8.1.2.1 Segmentation et tri cellulaire

Nous avons défini un schéma de segmentation basé sur une classification pixellaire rapide qui a de bonnes performances pour l'extraction de cellules (noyau et cytoplasme) présentes dans des images de microscopie médicale. Notre système d'apprentissage, qui réalise une simplification par prototypage des fonctions de décision, a fortement contribué à la production d'un classificateur rapide de pixels dont le rôle est prépondérant dans notre schéma de segmentation.

Nous avons défini un nouveau critère de qualité pour évaluer la performance d'une segmentation automatique par rapport à celle d'un expert. Ce critère est basé essentiellement sur une mesure d'adéquation avec les formes cellulaires de référence à identifier. Ce critère prend également en compte le fait que des cellules sont manquées ou des artefacts créés. Nous avons montré que, grâce à l'exploitation de ce critère, il est possible d'optimiser les performances d'un schéma de segmentation en déterminant les bons réglages à réaliser.

Nous avons souligné que la base de données de cellules dont nous disposions a des possibilités limitées pour réaliser une très bonne reconnaissance du type cellulaire et que sa constitution devra être améliorée. Nous avons mis en évidence la nécessité de développer des outils spécifiques pour permettre aux experts médicaux d'optimiser les coûts homme/machine relatifs à la constitution de bases d'apprentissage performantes. Nous mettons également en évidence le fait que la production de processus de décision performants dans la reconnaissance de cellules doit prendre en compte les relations hiérarchiques entre les différentes classes. De plus, si l'on cherche à réduire la complexité calculatoire du processus décisionnel à partir des images, nous mettons en évidence la nature du problème d'optimisation correspondant et insistons sur la complexité d'un tel problème.

8.1.2.2 Expertise automatique de la qualité des images compressées

Nous avons défini la problématique propre à l'estimation de la qualité visuelle des images. Nous avons montré qu'elle pouvait être abordée comme un problème d'apprentissage supervisé. Nous avons défini une caractérisation du système visuel humain afin de pouvoir produire une base d'apprentissage à partir d'un ensemble d'images et de l'avis moyen d'un panel d'observateurs sur la qualité de cet ensemble d'images. Nous avons montré qu'il est possible de produire par apprentissage un processus décisionnel qui si-

mule le comportement d'un panel d'observateurs relativement à la qualité visuelle d'images compressées avec JPEG2000. L'utilisation des techniques précédemment proposées et la prise en compte de différentes réflexions sur l'importance des relations sémantiques entre les classes ont permis de définir un système d'apprentissage simulant un tel expert informatique en qualité visuelle d'images compressées.

8.2 Perspectives

L'ensemble de ces travaux a mis en évidence le fait que la production d'un système d'apprentissage artificiel à partir de données supervisées correspond à la résolution d'un problème d'optimisation complexe car cela nécessite de prendre en compte simultanément six points, à savoir :

1. Le problème initial doit être généralement décomposé en plusieurs sous-problèmes pour améliorer la décision finale. Le choix des décompositions à réaliser est un problème difficile.
2. Les données sont plus ou moins pertinentes pour le problème initial, mais également pour chaque sous-problème considéré. Il est donc nécessaire de sélectionner celles qui seront utilisées pour entraîner un algorithme particulier d'apprentissage pour éviter de réduire ses capacités de généralisation.
3. Les données sont plus ou moins redondantes et il est nécessaire de les résumer pour réduire les temps d'apprentissage et de décision.
4. La combinaison des différents processus décisionnels, relatifs à chaque sous-problème résolu, pour obtenir une décision finale est un problème difficile. En particulier parce qu'il est nécessaire de prendre en compte les relations existant entre les classes.
5. Les différents algorithmes utilisés dans un système d'apprentissage ont des paramètres internes qu'il faut sélectionner correctement.
6. La qualité d'un processus décisionnel dépend à la fois de ses capacités en généralisation et de sa complexité. Le domaine d'application concerné peut influencer grandement l'importance relative de ces deux notions contradictoires.

Les méthodes à base de méta-heuristiques sont une voie à approfondir pour réaliser des systèmes d'apprentissage ayant la possibilité de prendre en considération tous ces éléments simultanément. Les approches proposées illustrent que cet objectif est atteignable en partie car elles ne considèrent pas tous ces éléments dans un même système d'apprentissage mais seulement un sous-ensemble de ces éléments. De nouveaux travaux devraient permettre de considérer l'ensemble de ces éléments en améliorant les approches proposées.

L'exploitation de schémas d'apprentissage en cascade est également une voie à approfondir. Si la mise en cascade ne permet pas une optimisation globale du problème, elle permet d'en réduire la complexité. La difficulté est alors de pouvoir évaluer le compromis qualité/complexité induit par de tels schémas. Nous avons également proposé des approches dans cette direction et montré qu'elles peuvent être efficaces.

Quelle que soit l'approche choisie, celles proposées dans cette thèse permettent de contribuer à la réalisation de tels systèmes, mais plusieurs perspectives peuvent être envisagées pour les améliorer.

Les techniques d'*alpha seeding* permettent de définir des procédures de validation croisée rapides, mais elles pourraient être étendues pour réduire les nombreuses phases d'apprentissage des SVM dans des systèmes d'apprentissage. Nous avons indiqué en section 5.3.2.7 que notre méthode de sélection d'attributs pertinents pourrait exploiter notre technique d'*alpha seeding*. Il est cependant possible de prendre en compte des modifications autres que l'ajout ou la suppression d'exemples. Un changement minime du modèle correspondant à l'ajout ou la suppression d'un attribut ou bien à une variation suffisamment faible des valeurs des hyper-paramètres devrait produire une nouvelle fonction de décision ayant des corrélations importantes avec l'ancienne. Notre méthode de recherche avec tabous procède de cette manière. La définition d'une technique d'*apha seeding* permettant de construire une nouvelle fonction de décision plus rapidement, à partir de la connaissance de la première, permettrait d'accélérer le recherche avec tabous. Des problèmes d'optimisation relatifs à des problèmes d'apprentissage plus difficiles pourraient alors être résolus dans le même ordre de temps. Les systèmes d'apprentissage à base d'algorithmes évolutionnaires pourraient également tirer profit de telles techniques en exploitant la variété de leurs populations à partir de principes similaires à celui de la méthode d'*apha seeding*/*boostrap* (*i.e.* rechercher dans la population une solution produite à partir d'un modèle proche de celui à traiter). Nous avons également mise en évidence le rôle essentiel de la valeur de ε dans la technique d'*apha seeding* proposée. Correctement exploitée, elle devrait permettre de réduire les coûts calculatoires des systèmes d'apprentissage. Dans la section 4.2.6, nous décrivons un cadre possible de son exploitation avec la recherche avec tabous ou avec les algorithmes évolutionnaires.

Nous avons signalé l'importance de l'optimisation des schémas de combinaison et proposé en particulier une méthode permettant de réaliser une optimisation globale du problème à base d'algorithmes évolutionnaires. L'approche proposée dans cette thèse ne réalise pas la sélection des attributs pertinents, ni la simplification des fonctions de décision produites, ni la sélection des décompositions les plus pertinentes (*i.e.* construction du graphe entre les classes). L'un des écueils majeur reste les temps nécessaire à l'optimisation de tels problèmes. Si l'exploitation de techniques d'*alpha seeding* doit permettre de réduire significativement les coûts calculatoires, nous pensons cependant que la définition de méthodes méta-heuristiques correspondant à une hybridation entre la recherche avec tabous et des algorithmes évolutionnaires est une solution envisageable pour aborder de tels problèmes. En effet, la recherche avec tabous à partir de stratégies de diversification et d'intensification ne chercherait pas à optimiser tous les paramètres d'un modèle en même temps, mais seulement ceux relatifs à des sous-ensembles de ceux-ci. Le fait de resoudre un sous-ensemble donné ou d'en changer correspondrait alors respectivement à la mise en œuvre de la stratégie d'intensification ou de diversification. La fait de paralléliser[1] la recherche sur plusieurs ordinateurs permettrait de développer plusieurs stratégies d'intensification qui seraient globalement pilotées par la recherche avec tabous et conduites sur chaque ordinateur par un algorithme évolutionnaire.

Nous avons exploité certaines des méthodes proposées dans cette thèse dans le cadre de la classification pixellaire d'images de microscopie médicale. Cela a permis mettre en évidence l'importance de la sélection d'un modèle approprié lors de l'exploitation des

1. Le développement des *clusters* d'ordinateur et les prix relativement faibles des micro-ordinateurs fait que, depuis plusieurs années, l'exploitation de plusieurs dizaines de micro-ordinateurs est parfaitement envisageable.

SVM. Nous l'avons montré par la construction de processus décisionnels performants et de complexités réduites. Nous pensons pouvoir exploiter facilement nos méthodes dans le cadre d'autres problématiques plus générales de vision par ordinateur. Nous en présentons ici quelques-unes en nous focalisant sur la base de données d'images segmentées manuellement nommée "The Berkeley Segmentation Dataset and Benchmark (BSDB)" [MARTIN01] qui est à présent bien connue dans le domaine du traitement d'images. Cette base contient 300 images segmentées manuellement par 30 humains. 200 de ces images servent de base d'apprentissage et les 100 restantes de base de test. Chaque image contient 154401 pixels, ce qui amène à manipuler des bases respectivement d'environ 30 et 15 millions d'exemples. La figure 7.1 présente quelques exemples provenant de cette base : les images originales, les cartes de contours résultant de la superposition de ceux des experts, les images mosaïques (recolorisées selon les régions des segmentations expertes et les images originales), une segmentation des images obtenue par une méthode extraite de [MEURIE05A]. Un article de référence [MARTIN04] concernant cette base de données montre que l'on peut utiliser des processus décisionnels afin d'extraire les contours des objets de façon automatique. Les auteurs concluent rapidement que les SVM ne sont pas du tout adaptés à ce type de problématique étant donné leur complexité (25% des exemples de la base d'apprentissage sont vecteurs de support). Cependant, ils ne cherchent pas à optimiser globalement leur modèle, ce qui rend le temps de traitement d'une seule image prohibitif (plusieurs heures). Nous avons pu montrer le contraire pour la classification de pixels d'images de microscopie dont les classes ont pourtant un fort taux de recouvrement. La BSDB correspond donc exactement au cadre de nos méthodes : une redondance très prononcée tant au niveau des exemples que des attributs. De plus, les hyper-paramètres des classificateurs doivent être optimisés. En nous basant sur la BSDB, nous projetons donc de concevoir des classifieurs à base de SVM permettant :

1. D'apprendre à détecter les contours dans une image. Cette extraction peut se faire automatiquement par le calcul d'une carte de probabilité de contours à l'aide d'un SVM binaire distinguant les pixels contours des pixels non contours. Le prototypage de la base permettra de réduire grandement le nombre d'exemples. La sélection des attributs pertinents permettra de sélectionner les facteurs importants décrivant un contour (luminance, couleur ou texture). Enfin, le compromis entre complexité et performance permettra d'atteindre des temps de traitements acceptables.

2. De modéliser les regroupements perceptuels dans une image. Les sous-ensembles de pixels (les régions) à l'intérieur d'une même zone dans la segmentation de référence constituent ce que l'on appelle un regroupement perceptuel. A partir d'une sur-segmentation (dernière ligne de la figure 8.2), il s'agira alors de déterminer par apprentissage si deux régions font partie du même groupement perceptuel, ceci afin de fusionner automatiquement des régions similaires dans le but de simplifier une segmentation afin qu'elle soit le plus proche possible de celles des experts (troisième ligne de la figure 8.2).

Nous avons signalé dans le chapitre 7 que la constitution d'une base d'apprentissage est difficile et qu'un nombre plus ou moins important d'exemples de ces bases est inutile voire contre-productif pour un processus d'apprentissage. Lorsque la constitution de la base nécessite l'intervention d'un expert humain pour réaliser la labellisation des exemples, le coût de cette labellisation peut être important. Si une grande partie de la base n'est pas exploitée et que cela requiert à nouveau une intervention des experts pour améliorer incré-

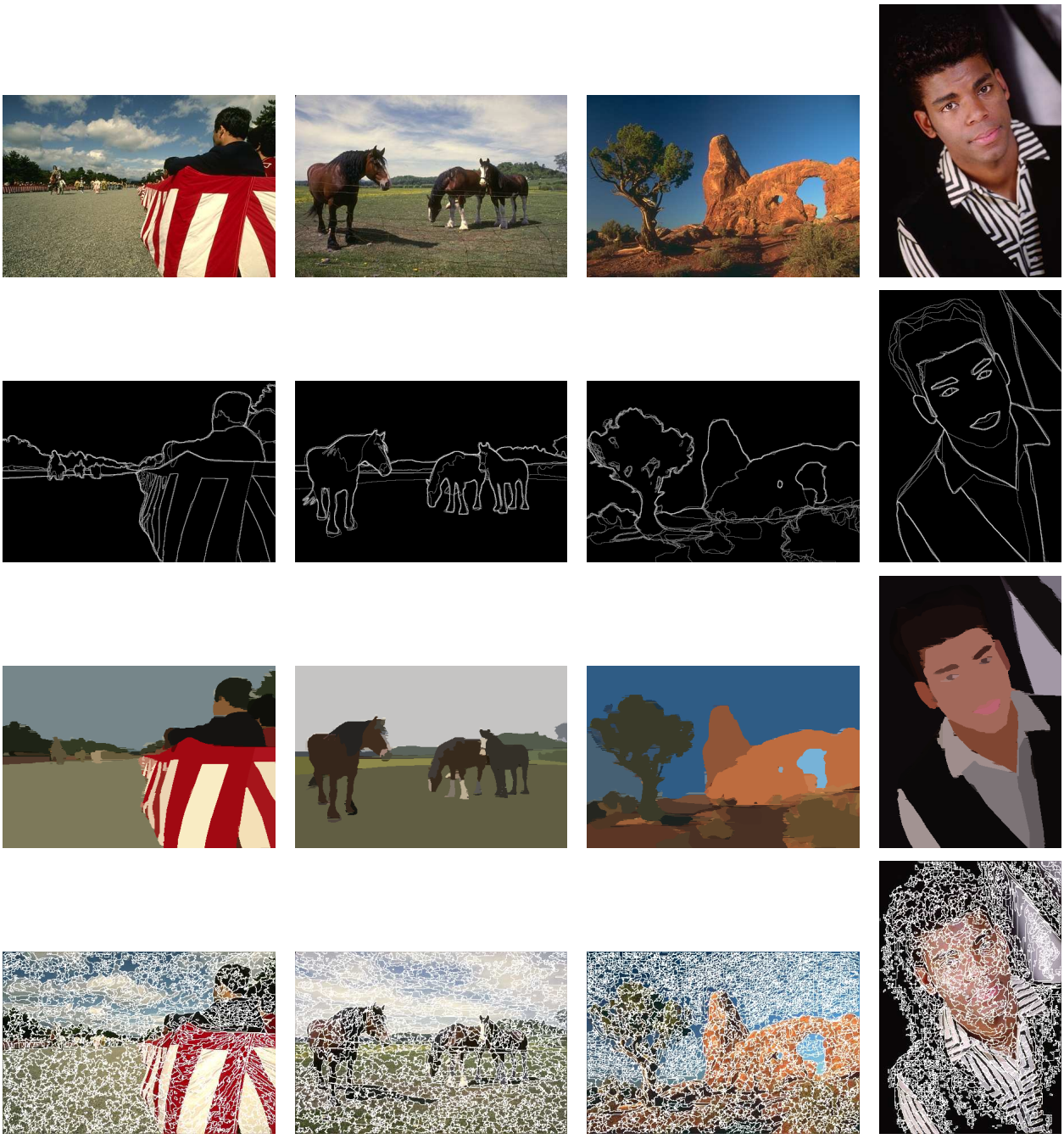
mentalement les données, le coût global est encore augmenté. La réalisation de systèmes d'apprentissage semi-supervisés conçus pour aider les experts à produire des bases d'apprentissage plus performantes avec des coûts de développement réduits est donc un axe de recherche à approfondir.

Le domaine médical produit de grandes quantités d'images et il est souvent nécessaire de pouvoir les archiver efficacement en exploitant des techniques de compression efficaces. La préservation de la qualité visuelle de ces images est alors essentielle. Le problème réside dans la définition de méthodes de compression adaptées. Si la nature des images médicales peut être très variée dans l'absolu (microscopie, radiographie, etc.), ceci n'est plus vrai si l'on se focalise sur une problématique particulière. Par exemple, les images médicales microscopiques de prélèvements bronchiques correspondent à un sous-ensemble très restreint des images médicales de toutes natures. La prise en compte des particularités de ce sous-ensemble peut permettre de définir un système d'archivage spécifique avec des taux de compression importants et une qualité visuelle élevée. Ceci est à mettre en relation avec le fait que nous avons montré qu'il est possible de développer des experts informatiques spécialisés dans la qualité visuelle des images[2]. La production d'un système de compression spécifique peut alors être abordée comme un problème d'apprentissage supervisé où les réglages d'une méthode de compression suffisamment générique[3] correspondent à la sélection d'un modèle dont l'évaluation se fait par le biais d'un expert informatique spécialisé.

L'ensemble des travaux réalisés dans cette thèse doivent donc permettre, dans un futur plus ou moins proche, des développements, selon différents axes relatifs au traitement et à l'analyse d'images. Ceci doit être perçu comme une problématique d'apprentissage supervisé dont la sélection de multi-modèles performants est la pierre angulaire. Rien n'empêche néanmoins d'étendre les approches proposées (ou leurs futures extensions) à d'autres domaines pour lesquels la définition de systèmes d'apprentissage performants serait relative à des problématiques de sélection de modèles.

2. Dans le chapitre 7, l'expertise correspond à des images naturelles, mais il n'y pas d'obstacles majeurs à produire des experts automatiques en qualité visuelle pour des images médicales d'une nature donnée à partir d'un panel d'experts humains de ce domaine.

3. La norme JPEG2000 est suffisamment flexible pour intégrer un grand nombre de spécificités, bien que dans notre cas, rien ne nous oblige à nous restreindre à cette norme.



(o)

FIG. 8.1 – *Images extraites de la "Berkeley Segmentation Dataset and Benchmark (BSDB)" et leurs sur-segmentations [*MEURIE05A*]. Première ligne : images couleur, deuxième ligne : superpositions des segmentation manuelles, troisième ligne : mosaïques des segmentations, quatrième ligne : des sur-segmentations.*

ANNEXE A

BASES DE DONNÉES

Sommaire

Nous donnons dans cette annexe un descriptif des différentes bases de données utilisées dans l'ensemble de cette thèse pour illustrer les performances des différentes méthodes d'apprentissage proposées. Ces bases de données ont été utilisées dans les chapitres 4 à 6.

Nous donnons également dans cette annexe des résultats d'apprentissage avec des SVM en utilisant des méthodes classiques d'apprentissage avec des SVM. Ces résultats illustrent entre autre l'importance de la sélection des hyper-paramètres pour la production de fonctions de décision performantes avec des SVM. L'ensemble de ces résultats sont essentiellement utilisés pour mettre en évidence les effets positifs de notre méthode de sélection d'exemples pertinents (*cf.* section 5).

A.1 Descriptions

Le tableau A.1 et A.2 donnent la composition des différentes bases de données en spécifiant, le nom, la taille de la base d'apprentissage (m_{Z^A}) et la taille de la base de validation (m_{Z^V}). Le nombre d'attributs (n) et le nombre de classes (n_ω) sont également indiqués. Pour information est également donné le nombre d'exemples de chaque classe (dans l'ordre des numéros de labels) pour chaque base d'apprentissage ($m_{\omega_i}(A)$) et pour chaque base de validation ($m_{\omega_i}(V)$). La majorité des bases (**Breast-cancer** à **Web** dans le tableau A.1 et **Australian**, **Heart** et **German** dans le tableau A.2) proviennent de l'*UCI repository* [BLAKE98]. La base USPS a été utilisée initialement dans [HULL94][1]. La base **QIjpeg2M** correspond à une étude de qualité visuelle d'images couleur pour différents niveaux de compression avec JPEG2000 (pour plus détails voir la section 7.2) . La base **Serous** correspond à une étude sur la classification de cellules en cytologie des séreuses [LEZORA00] (pour plus détails voir la section 7.1.3). La base **Classifpixel**[2] correspond à une étude sur la segmentation rapide d'images de cellules en cytologie des séreuses [LEBRUN05B] (pour plus détails voir la section 7.1.2). La base **Adult0** dans le tableau A.2 a été constituée à partir de la base **Adult** par Platt [PLATT99A][3].

1. Elle est disponible à l'adresse: http://algoval.essex.ac.uk:8080/tc5/dataset/dataset.jsp?op=3

2. Cette base est exploitée dans le chapitre 5. Par contre, celle utilisés pour l'application de segmentation du chapitre était différente de celle du chapitre 7.2 (la méthode d'échantillonnage utilisée pour les produire à partir des images était différente dans les deux cas).

3. elle est disponible à l'adresse: http://research.microsoft.com/~jplatt/smo.html

Base	m_{Z^A}	m_{Z^V}	n	n_ω	$m_{\omega_i}(A)$	$m_{\omega_i}(V)$
QIjpeg2M	116	111	38	5	6, 19, 30, 46, 15	13, 27, 16, 40, 15
Serous	3173	784	46	12	289, 224, 174, 604, 286, 505, 481, 31, 22, 320, 27, 210	71, 55, 43, 150, 71, 126, 119, 7, 5, 79, 6, 52
Bcancer	548	135	9	2	356, 192	88, 47
Wine	144	34	13	3	48, 57, 39	11, 14, 9
Bupa	278	67	6	2	117, 161	28, 39
Glass	175	39	9	6	57, 61, 14, 11, 8, 24	13, 15, 3, 2, 1, 5
ionosphere	281	69	34	2	180, 101	44, 25
Iris	120	30	4	3	40, 40, 40	10, 10, 10
Pima	616	152	8	2	401, 215	99, 53
Segment	175	35	19	7	25, 25, 25, 25, 25, 25, 25	5, 5, 5, 5, 5, 5, 5
Vehicle	679	167	18	4	170, 174, 175, 160	42, 43, 43, 39
Satimage	4435	2000	36	6	1072, 479, 961, 415, 470, 1038	461, 224, 397, 211, 237, 470
PageBlocks	4382	1091	10	5	3931, 264, 23, 71, 93	461, 224, 397, 211, 237
PenDigits	7494	3498	16	10	780, 779, 780, 719, 780, 720, 720, 778, 719, 719	363, 364, 364, 336, 364, 335, 336, 364, 336, 336
OptDigits	3064	759	64	10	301, 312, 305, 312, 310, 301, 302, 310, 305, 306	75, 77, 75, 77, 77, 75, 75, 77, 75, 76
Letter	16015	3985	16	26	$\approx$ 615 par classe	$\approx$ 153 par classe
Shuttle	43500	14500	9	6	34108, 37, 132, 6748, 2458, 17	11478, 13, 39, 2155, 809, 6
Adult	30148	15074	123	2	4719, 22729	3625, 11449
Web	44837	4912	300	2	43501, 1336	4769, 143
Classpixel	149758	74878	27	3	133284, 10483, 5991	66641, 5241, 2996

TAB. A.1 – *Descriptif des différentes bases d'apprentissage et de validation utilisées dans le chapitre 5.*

Base	m_{Z^A}	m_{Z^V}	n	n_ω	$m_{\omega_i}(A)$	$m_{\omega_i}(V)$
Adult0	1605	30956	123	2	395, 1210	7446, 23510
Australian	390	300	14	2	174, 216	133, 167
Heart	180	90	13	2	82, 98	38, 52
German	400	600	24	2	108, 292	192, 408
USPS	7291	2007	256	10	$\approx$ 729 par classe	$\approx$ 201 par classe

TAB. A.2 – *Descriptif des différentes bases d'apprentissage et de validation utilisées dans le chapitre 4 et le chapitre 6.*

Les bases du tableau A.1 (sauf les 3 dernières) contenant plus de deux classes ont été transformées en un problème de classification binaires pour la première approche du chapitre 5. Le tableau A.3 donne la nature de la transformation (la chaine de caractères "-2c" est ajoutée pour faciliter leurs identifications). La colonne: (ω_1) - (ω_2), indique quelles classes ont été fusionnées ensemble et quelles classes ont été supprimées[4]. Il y a deux raisons principales à ce choix. Premièrement, les SVM sont des classificateurs binaires.

4. Lorsque l'information était disponible, les classes proches sémantiquement ont été fusionnées ensembles. Pour réduire la taille de certaines bases multi-classes et faciliter leurs utilisations, les bases **PenDigits-2**, **OptDigits-2** et **Letter-2c** utilisent seulement deux classes des bases multi-classes associées.

L'influence de choisir un schéma de combinaison par rapport à un autre n'est pas une opération neutre et la mesure de l'effet plus ou moins positif de techniques de simplification avec des SVM sur des problèmes multi-classes est donc dépendante du schéma choisi. Pour des problèmes binaires, cet effet est annulé et seul l'effet de la simplification avec des SVM est mesuré.
Deuxièmement, il est généralement plus facile de réaliser une bonne généralisation sur un problème à deux classes issues d'un problème multi-classe lorsque plusieurs classes sont fusionnées (en particulier si elles sont proches entre elles) ou écartées (*cf.* chapitre 6). Ces problèmes doivent donc permettre de réduire de façon plus importante le nombre minimum d'exemples nécessaire à la construction d'un classificateur performant. L'observation de ce fait laisse alors penser qu'il est préférable de simplifier chaque SVM binaire, puis de les combiner ensemble plutôt que de réaliser les simplifications globalement pour un schéma multi-classe donné.

Base	m_{Z^A}	m_{Z^V}	n	(ω_1) - (ω_2)	$m_{\omega_i}(A)$	$m_{\omega_i}(V)$
QIJPEG2M-2c	116	111	38	(1-3) - (4,5)	55, 61	56, 55
Serous-c	3173	784	46	(1-9) - (10-12)	2804, 369	694, 90
Wine-2c	144	34	13	(1,3) - (2)	87, 57	20, 14
Glass-2c	175	39	9	(1,2) - (3-6)	118, 57	28, 11
Iris-2c	120	30	4	(1) - (2,3)	40 80	10 20
Segment-2c	175	35	19	(1,4-6) - (2,3,7)	100, 75	20, 15
Vehicle-2c	679	167	18	(1,2) - (3,4)	344, 335	85, 82
Satimage-2c	4435	2000	36	(1,2,5) - (3,4,6)	2021, 2414	922, 1078
PageBlocks-2c	4382	1091	10	(1) - (2-5)	3931, 451	982, 109
PenDigits-2c	1559	727	16	(0) - (1)	780, 779	363, 364
OptDigits-2c	613	152	64	(0) - (1)	301, 312	75, 77
Letter-2c	1245	310	16	(1) - (2)	632 613	157 153
Shuttle-2c	43500	14500	9	(1) - (2-6)	34108, 9392	11478, 3022

TAB. A.3 – *Descriptif des bases multi-classes transformées en des problèmes binaires.*

A.2 Quelques résultats de référence

Pour comparer les résultats de nos différentes méthodes, nous rapellons ici les taux d'erreur sur chacune des bases des tableaux A.1 et A.3 avec les *k*-PPV et les SVM[5] sur chaque base d'apprentissage et de validation. Le tableau A.4 donne les résultats obtenus avec les *k*-PPV pour une valeur de k sélectionnée à partir d'une procédure de validation croisée en 5 parties. On peut remarquer qu'une valeur de $k = 1$ est souvent suffisante dans la majorité des cas. Ces résultats illustrent l'intérêt de la règle *1*-PPV.

5. Nous utilisons la version 2.8 de la *libsvm* [CHANG01]. Pour les problèmes multi-classes, c'est la méthode *un-contre-un* qui est utilisée. La valeur de ϵ pour le critère d'arrêt de SMO est de 10^{-2}, sauf mention contraire. Les expérimentations du chapitre 4 ont montré que c'est une valeur suffisamment faible pour obtenir des solutions très proches de l'optimal avec cet algorithme. Le gain en nombre d'itérations de SMO peut être par contre très significatif.

Base	k	e_{A}	e_{V}
QIjpeg2M	3	18,10%	26,13%
QIjpeg2M-2c	11	0,86%	0,90%
Serous	1	39,08%	45,4%
Serous-2c	1	2,42%	5,48%
Breast-cancer	1	4,92%	5,18%
Wine	1	5,55%	5,88%
Wine-2c	1	6,94%	5,88%
Glass	1	26,85%	20,51%
Glass-2c	1	17,14%	15,38%
ionosphere	1	20,64%	20,28%
Iris	1	5,0%	0,0%
Iris-2c	1	0,0%	0,0%
Pima	3	31,16%	34,21%
Bupa	3	36,33%	38,80%
Segment	1	9,71%	2,85%
Segment-2c	1	4,57%	2,86%
Vehicule	1	29,45%	31,13%
Vehicule-2c	1	5,30%	6,58%
Satimage	1	9,49%	10,55%
Satimage-2c	1	2,84%	2,45%
PageBlocks	1	2,81%	7,14%
PageBlocks-2c	1	2,62%	6,51%
Pendigits-2c	1	0,06%	0,54%
OptDigits-2c	1	0,0%	0,0%
Letter-2c	1	0,08%	0,32%
Pendigits	1	0,56%	2,26%
OptDigits	1	2,41%	3,29%
letter	1	8,44%	7,73%
shuttle	1	0,18%	0,09%
shuttle-2c	1	0,21%	0,11%

TAB. A.4 – *Taux d'erreur avec l'algorithme des* k*-PPV.* e_{A} *et* e_{V} *sont respectivement le taux d'erreur estimé par la procédure de validation croisée en 5 parties et le taux d'erreur sur la base de validation pour cette valeur de* k*.*

Le tableau A.5 donne les mêmes taux d'erreur avec des SVM pour deux procédures de sélection de modèle. Dans les deux cas un noyau gaussien est utilisé (C et σ sont les valeurs sélectionnées du modèle). Les deux procédures de sélection sont les suivantes:

– **Sélection directe**
La première procédure de sélection de modèle, nommé *directe*, choisit un modèle qui ne dépend pas de résultats mesurés à partir de la fonction de décision produite par un SVM, mais réalise sa sélection à partir d'une heuristique. La valeur de C est fixée à 10 (une régularisation moyenne) et la valeur de σ pour le noyau gaussien est déterminée à partir des distances entre les exemples. Le choix de la valeur de σ est basé sur le fait qu'un noyau gaussien définit une notion de similarité entre les exemples:

$$s(\mathbf{x}_i,\mathbf{x}_j) = K(\mathbf{x}_i,\mathbf{x}_j) = \exp\left(-\frac{d(\mathbf{x}_i,\mathbf{x}_j)^2}{2\sigma^2}\right) \tag{A.1}$$

Idéalement $s(\mathbf{x}_i,\mathbf{x}_j) = 1$ si $y_i = y_j$ et zéro dans le cas contraire. Cette notion est utilisée pour guider la construction du noyau approprié [KWOK03] par l'estimation de la corrélation entre la matrice idéale et celle produite par la fonction noyau (l'ensemble des valeurs de similarité entre les exemples). Nous avons défini une heuristique à partir de cette remarque pour la détermination d'une valeur appropriée de σ. L'objectif est que pour les couples d'exemples d'une même classe la valeur de la similarité soit proche de 1 et que pour les couples d'exemples de classes différentes la valeur de la similarité soit proche de zéro. Soit D_1 l'ensemble des distances entre des couples d'exemples de la même classe et D_2 l'ensemble des distances entre des couples d'exemples d'une autre classe. Nous définissons les deux valeurs σ_1 et σ_2 suivantes :

$$\sigma_i = \sqrt{\frac{1}{2\log(2)\,|D_i|}\sum_{d\in D_i} d^2} \tag{A.2}$$

Les remarques précédentes permettent d'écrire les deux contraintes suivantes: $\sigma > \sigma_1$ et $\sigma < \sigma_2$ pour que respectivement la valeur de similarité s soit en moyenne supérieure et inférieure à 0,5 suivant que les deux exemples $\mathbf{x}_i$ et $\mathbf{x}_j$ sont ou ne sont pas de même classe. Nous avons simplement fixé $\sigma = \frac{\sigma_1+\sigma_2}{2}$. Lorsqu'on compare les résultats obtenus avec cette méthode à une sélection classique par validation croisée (en 5 parties) avec technique de "grid search", les résultats obtenus sont très proches, à la fois en généralisation et en nombre de vecteurs de support utilisés.

– **Sélection par minimisation du nombre de vecteurs de support**
Dans la deuxième procédure de sélection (nommé $\min_{\mathrm{SV}}$) nous utilisons une recherche de modèle de type *grid search*. Comme l'objectif de notre méthode est de produire un schéma de compression efficace, nous avons choisi que la sélection de modèle soit réalisée à partir du nombre de vecteurs de support de la solution optimale d'un SVM. Le modèle sélectionné est celui qui en a le moins. Cette façon de procéder correspond à définir un principe d'inférence basé sur la minimisation de la borne supérieure d'erreur de généralisation (2.52). Ce critère est efficace, mais ne produit pas la sélection du meilleur modèle (en généralisation) lorsque les données sont bruitées, parce qu'il impose de choisir des marges géométriques faibles avec des données de cette nature. Malgré tout, si la complexité est suffisamment réduite,

sans réduire significativement les performances en généralisation, ce critère de sélection doit permettre de trouver un bon compromis entre complexité et efficacité.

Les résultats dans le tableau A.5 montrent que la première procédure de sélection est plus perfomante que la seconde pour produire des solutions efficaces en généralisation, bien que la seconde puisse produire des solutions plus efficaces ou équivalentes (base **Segment-2c** par exemple). Les fonctions de décision produites par les SVM avec ces deux procédures de sélection sont globalement plus performantes que celles de la règle *k*-PPV[6]. Le nombre de vecteurs de support utilisés par la fonction de décision est par contre plus faible dans le cas de la seconde procédure de sélection que dans le cas de la première (sauf avec **Pendigits**), ce qui est logique car le principe d'inférence utilisé favorise fortement cet effet. On peut également vérifier la véracité de la borne supérieure (2.52) lorsqu'on compare les valeurs de e_V et $SV\%$ (exception faite de la base **PageBlock2c**). Cependant, tout comme d'autres bornes sur l'erreur de généralisation, cette borne est basée sur une expression probabiliste et elle peut donc être mise en défaut sur un cas particulier. Malgré tout, l'écart est faible, et au regard de l'ensemble des résultats elle est plutôt pessimiste en général, bien que l'écart entre cette borne supérieure et le taux d'erreur diminue avec l'augmentation du nombre d'exemples.

Dans le cas des problèmes multi-classes, c'est une combinaison de classificateurs binaire qui est réalisée. L'utilisation du nombre de vecteurs de support pour estimer la borne supérieure sur l'erreur de généralisation n'est donc pas strictement rigoureuse. Dans le cas d'un schéma multi-classe *un-contre-un* qui utilise $n(n-1)/2$ classificateurs binaires (*cf.* section 6), un exemple qui n'est pas vecteur de support est un exemple qui n'est pas utilisé par les $(n-1)$ fonctions de décision impliquées par la classe de l'exemple. Cette remarque illustre l'éloignement de l'exemple de l'ensemble des frontières de décision concernées et donc le fait que l'ensemble de ces fonctions de décision seront identiques sans sa présence dans la base d'entraînement. D'un point de vue général, la mesure du nombre de vecteurs de support dans le cadre d'un schéma multi-classe correspond à l'estimation de l'efficacité d'un schéma de compression, sauf que la fonction de décision produite n'est généralement pas cohérente avec la base d'apprentissage. La borne (2.51) n'est donc normalement pas directement applicable. En pratique, en regardant le tableau A.5, on remarque que pour certaines bases la fonction de décision des SVM est cohérente avec la base d'apprentissage, ce qui la rend applicable pour ces quelques bases. De plus, pour beaucoup de ces bases la fonction de décision produite est très proche de la cohérence avec la règle PPV. Ces deux remarques laissent supposer le bien fondé d'utiliser le pourcentage d'exemples utilisés par la fonction de décision d'un schéma multi-classe comme borne supérieure de l'erreur de généralisation et par là même de l'utiliser comme principe d'inférence. Cette estimation est par contre dépendante du type de schéma multi-classe utilisé.

6. Une des raisons est liée à la nature fortement discontinue de la frontière de décision produite par cette règle (voir [VINCEN03]).

	directe					$\min_{\mathrm{SV}}$				
Base	C	σ	e_A	e_V	VS %	C	σ	e_A	e_V	VS %
QIjpeg2M	10	1,3	0,86%	22%	60%	1000	7,1	0%	**19**%	**47**%
QIjpeg2M-2c	10	1,6	0%	0,9%	13%	100	2,2	0%	0,9%	**11**%
Iris	10	2	2,5%	0%	22%	1000	7,1	1,7%	0%	**13**%
Iris-2c	10	2,4	0%	0%	3,3%	100	7,1	0%	0%	**2,5**%
Wine	10	1,1	0%	**0**%	28%	1000	7,1	0%	*2,9*%	**17**%
Wine-2c	10	1,2	0%	0%	19%	1000	7,1	0,69%	0%	**11**%
Glass	10	0,88	11%	**21%**	72%	1000	2,2	6,9%	*26*%	**66**%
Glass-2c	10	0,91	10%	13%	40%	1000	2,2	8%	**10**%	**33**%
Segment	10	1,1	1,7%	**2,9**%	58%	1000	2,2	0%	*8,6*%	**47**%
Segment-2c	10	1,3	2,3%	8,6%	24%	1000	2,2	0,57%	**2,9**%	**17**%
Bupa	10	0,82	21%	**31**%	62%	1000	0,71	3,2%	*34*%	**55**%
Ionosphere	10	2,1	2,1%	**4,3**%	33%	1000	2,2	0%	*7,2*%	**26**%
Breast-cancer	10	0,61	2%	2,2%	11%	1000	7,1	2,4%	2,2%	**9,3**%
OptDigits-2c	10	1,9	0%	0%	6%	1000	22	0%	0%	**2,8**%
Pima	10	0,91	20%	**23**%	54%	1000	0,71	1,1%	*32*%	**45**%
Vehicle	10	1,4	12%	**17**%	59%	1000	2,2	4,6%	*19*%	**43**%
Vehicle-2c	10	1,5	0,74%	**1,8**%	20%	1000	2,2	0%	*2,4*%	**8,4**%
Letter-2c	10	1,2	0,08%	**0,97**%	4,3%	1000	7,1	0,24%	*1,6*%	**2,5**%
Pendigits-2c	10	140	0%	1,2%	1,5%	1	71	0,064%	**0,96**%	4,5%
OptDigits	10	1,8	0,033%	**1,6**%	30%	1000	22	0,65%	*2,2*%	**21**%
Serous	10	2	9,8%	35%	62%	1000	7,1	9,7%	*36*%	**53**%
Serous-2c	10	2,2	0,44%	2,2%	4,7%	1000	7,1	0,5%	*2,4*%	**3,4**%
PageBlocks	10	1	2,3%	5,8%	7,7%	1000	2,2	2%	5,7%	**6,5**%
PageBlocks-2c	10	1	2,1%	**6,2**%	7,1%	1000	0,71	1,1%	7,7%	**5,6**%
Satimage	10	100	7,4%	10%	27%	1000	71	0,25%	*11*%	**25**%
Satimage-2c	10	130	1,6%	2,3%	7,2%	100	71	0,2%	*2,6*%	**6,2**%
Pendigits	10	120	0,15%	1,7%	9,9%	100	71	0,013%	1,7%	12%
Letter	10	1,2	3,2%	5,8%	42%	1000	2,2	0,62%	5,4%	**32**%
Shuttle-2c	10	190	0,36%	0,41%	5,9%	1000	71	0,0046%	**0,14**%	**0,45**%
Shuttle	10	250	1,2%	1,2%	8,1%	1000	71	0,0046%	**0,13**%	**0,71**%

TAB. A.5 – *Taux d'erreur et pourcentage de vecteurs de support avec différentes méthodes de sélection des hyper-paramètres des SVM.* e_{A} *et* e_{V} *sont respectivement le taux d'erreur estimé pour la base d'apprentissage et la base de validation pour le modèle* (C,σ) *sélectionné. La colonne* $SV\%$ *indique le pourcentage de vecteurs de support par rapport au nombre d'exemples de la base d'apprentissage.*

ANNEXE B

LUMINANCE MOYENNE LOCALE

Sommaire

Nous détaillons dans cette annexe comment le critère de luminance $a_{k,\theta,i}(u,v)$ utilisé en section 7.2.2.1 est déterminé à partir d'une image RVB afin de permettre la caractérisation de la sensibilité du Système Visuel Humain (SVH) aux variations de contraste.

La processus réalisant la caractérisation de la lumiance moyenne locale se décompose en trois parties :

1. Décomposition colorimétrique dans un espace couleur adapté au SVH.
2. Décomposition en canaux perceptuels modélisant la réceptivité des cellules du cortex visuel.
3. Filtrage de l'image à partir de la décomposition en canaux perceptuels.

B.1 Décomposition colorimétrique

La première étape repose sur une décomposition colorimétrique de l'espace RVB dans l'espace de Krauskopf AC_1C_2 [KRAUSK82] de manière à aboutir à une représentation spatio-colorimétrique dans laquelle la perception uniforme des couleurs par le SVH est respectée. Selon la théorie des couleurs opposées de Young et de Hering, l'information couleur captée par le SVH est transmise au cerveau sous la forme de trois signaux : un signal achromatique (opposition blanc-noir) et deux signaux chromatiques (opposition vert-rouge et bleu-jaune) [KOWALI90].

Les nouvelles coordonnées A, C_1 et C_2 sont obtenues à partir des primaires L, M et S correpondant aux réponses des cônes du système visuel :

$$\begin{aligned} A &= \frac{1}{2}(L+M) \\ C_1 &= \frac{1}{2}(L-M) \\ C_2 &= \frac{1}{2}(S-\frac{L+M}{2}) \end{aligned} \tag{B.1}$$

Les coordonnées d'une couleur exprimée dans l'espace LMS peuvent se calculer à partir des composantes couleur exprimées dans l'espace XYZ à partir de la transformation suivante :

$$\begin{aligned} L &= 0.15514X + 0.54312Y - 0.03286Z \\ M &= -0.15514X + 0.45684Y + 0.03286Z \\ S &= 0.00801Z \end{aligned} \tag{B.2}$$

Les coordonnées dans l'espace XYZ de référence sont obtenues à partir d'une transformation de coordonnées RVB qui dépend des primaires et du blanc de référence [TREMEA03].

B.2 Décomposition en canaux perceptuels

La décomposition en canaux perceptuels est réalisée en utilisant les filtres cortex [WATSON87]. L'avantage de ces filtres réside dans le fait que leur réponse impulsionnelle ressemble à une fonction de Gabor 2D (à l'instar du champ réceptif de la plupart des cellules du cortex visuel). Ces filtres sont obtenus par combinaison linéaire des filtres définissant la sélectivité radiale (filtre *Dom*) et des filtres caractérisant la sélectivité angulaire (filtres *Fan*) :

$$\text{Cortex}_{k,\theta,i}(u,v) = dom_i(u,v).fan_{k,\theta}(u,v) \tag{B.3}$$

La figure B.1 représente le modèle de perception de la luminance défini par Daly [DALY93]. La série de chiffres arabiques spécifie les filtres *fan*, tandis que les chiffres romains désignent les filtres *dom*.

Les filtres *Dom* (*d*ifference *o*f *m*esa) sont générés par différence de deux filtres "mesa" d'indices consécutifs :

$$\text{Dom}_i(u,v) = M_{i-1}(u,v) - M_i(u,v), \tag{B.4}$$

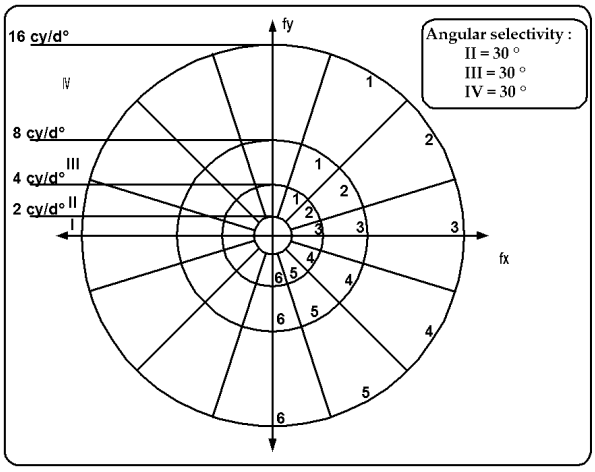


FIG. B.1 – *Modèles de perception de la luminance selon Daly [*DALY93*].*

où u et v représentent les fréquences spatiales cartésiennes. Le filtre *mesa* $M_i(u,v)$ est un filtre passe-bas en fréquences radiales généré à partir du filtre Mesa $M_0(u,v)$:

$$\mathrm{M}_0(u,v) = \left(\frac{\gamma}{f_0}\right)^2 \exp\left[-\pi\left(\frac{\omega\gamma}{f_0}\right)^2\right] \otimes \prod\left(\frac{\omega}{2f_0}\right) \tag{B.5}$$

avec $\omega^2 = u^2 + v^2$.

$\prod\left(\frac{\omega}{2f_0}\right)$ représente la fonction porte 2D à symétrie circulaire centrée en 0 et de rayon f_0. γ est un paramètre de raideur de l'atténuation qui est lié à l'écart-type σ_0 de la gaussienne par $\sigma_0 = \frac{1}{\sqrt{2\pi}}\frac{f_0}{\gamma}$.

Le filtre *mesa* d'indice i est alors défni par:

$$\mathrm{M}_i(u,v) = \mathrm{M}_0(\tau_i u, \tau_i v) \tag{B.6}$$

avec f_i un facteur d'échelle donné par $\tau_i = \prod_{j-1}^{i-1} \tau_j$.

Les filtres *Fan* modélisent les attributs d'orientation de la sélectivité en fréquences spatiales. Ceci est obtenu en appliquant un flou gaussien sur un filtre angulaire idéal. Selon la direction verticale, cette évolution est donnée par :

$$\mathrm{M}'_0(u,v) = \mathrm{H}(v) \otimes \gamma_b \exp(-\pi\gamma_b^2 v^2) \tag{B.7}$$

où $\mathrm{H}(v)$ est le filtre échelon, γ_b un paramètre de raideur. Etant donné que le filtre échelon n'a pas de variation selon l'axe u, la convolution peut s'exprimer par :

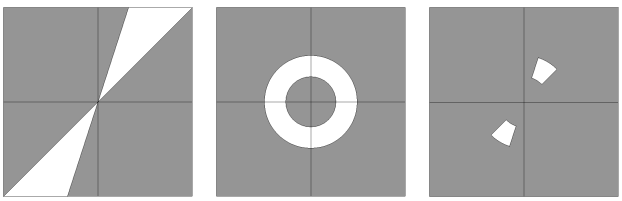
$$\mathrm{M}'_0(u,v) = \mathrm{F}(\gamma_b, v) = \int_{-\infty}^{v} \exp(-\pi\gamma_b^2\omega^2) d\omega \tag{B.8}$$

Pour une orientation θ, l'expression du filtre échelon est alors :

$$\mathrm{M}'_\theta(u,v) = \mathrm{F}(\gamma_b, (v\cos\theta - u\sin\theta)). \tag{B.9}$$

Le filtre *fan* correspondant à la $k^{\text{ème}}$ direction est donné par :

$$fan_{k,\theta}(u,v) = \mathrm{M}'_{\theta k}(u,v) - \mathrm{M}'_{\theta k+1}(u,v) \tag{B.10}$$



(a) Filtre (*Fan*). (b) Filtre (*Dom*). (c) Filtre cortex résultant.

FIG. B.2 – *Le filtre cortex résultant du produit d'un filtre* Dom *et d'un filtre* Fan.

B.3 Calcul des images filtrées

L'image est ensuite filtrée par chacun des filtres cortex de manière à obtenir un sous-ensemble d'images $a_{k,\theta,i}(u,v)$ tel que

$$a_{k,\theta,i}(u,v) = \text{Cortex}_{k,\theta,i}(u,v).S(u,v)$$

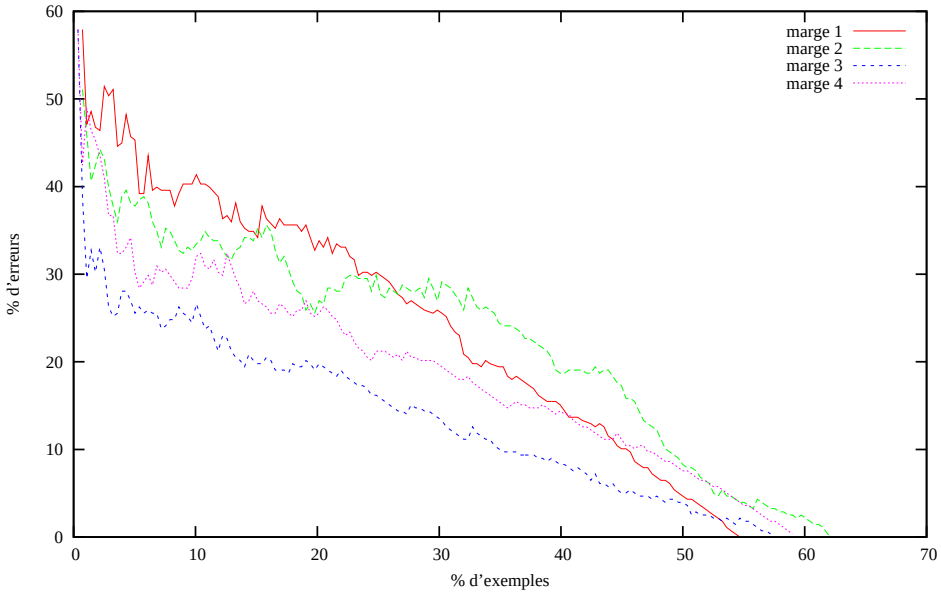
où $S(u,v)$ représente le spectre de l'image et $a_{k,\theta,i}(u,v)$ représente la luminance moyenne locale relatif à un triplet k,θ,i.

ANNEXE C

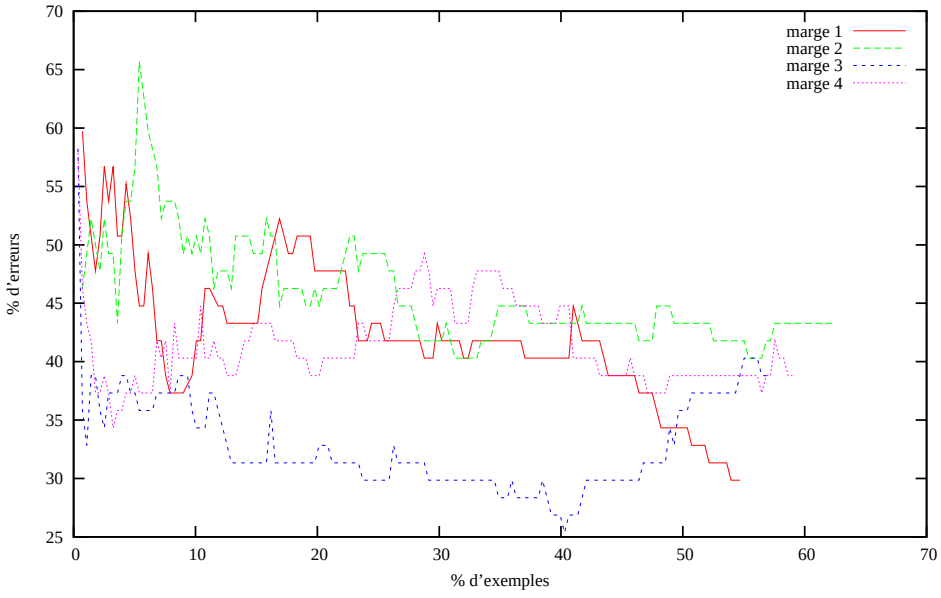
COURBES SUR LA SÉLECTION D'EXEMPLES PERTINENTS

Les figures de cette annexe correspondant à des résultats supplémentaires qui illustrent l'évolution du taux d'erreur en fonction du nombre d'exemples pertinents pour les quatre critères de confiances ((*c.f.* section 5.3.1.3). Les remarques suivantes viennent en complément de la synthèse faite dans la section 5.3.1.4.

Les figures Les figures C.1 à C.10 illustrent que si le critère $[\gamma \nearrow]$ produit des fonctions de décision qui utilisent le moins d'exemples pour être cohérentes avec la base de données, il est celui qui a généralement le taux d'erreur le plus important et ceci pendant la majorité des itérations de notre méthode. Les figures C.1, C.2, C.5, C.7 et C.9 illustrent l'importance de cet effet et elles corroborent les risques évoqués précédemment pour ce critère lorsque les données sont bruitées.
Le critère $[\gamma \searrow]$ a un comportement qui dépend fortement des bases de données utilisées. Les figures C.2, C.4, C.9, illustrent que la sélection avec ce critère produit rapidement une fonction de décision de bonne qualité avant d'avoir atteint la cohérence avec la base d'entraînement. Les figures C.1, C.3, C.5 et C.7 illustrent l'effet contraire. Globalement ce critère n'est pas approprié à l'utilisation d'un critère d'arrêt prématuré.
Le critère $[\gamma_{\odot} \searrow]$ est celui le plus à même de produire la sélection d'un ensemble réduit d'exemples pertinents de façon à ce que la règle PPV soit performante en généralisation à la fois avec la base d'apprentissage et avec celle de validation. L'ensemble des figures C.1 à C.9 illustrent cette possibilité. Cet effet est très marqué sur la figure C.5. Une remarque importante est qu'avec certaines bases de données (voir les figures C.1, C.5 et C.7), le fait de devoir produire un ensemble d'exemples qui soit totalement cohérent avec la base d'apprentissage a pour effet de réduire les capacités de généralisation de la règle PPV produite par les dernières itérations de notre méthode. Ce phénomène doit en effet correspondre à l'ajout d'exemples aberrants lorsque les données d'apprentissage sont significativement bruitées. La proportion d'exemples fortement significatifs peut alors devenir faible par rapport à la proportion d'exemples non significatifs au point de rendre l'apprentissage plus difficile avec cet ensemble réduit d'exemples. L'observation de ce phénomène justifie l'utilisation d'un critère d'arrêt avant cohérence avec la règle PPV pour réduire l'effet indésirable lié à l'ajout de données non significatives (*i.e.* non pertinentes de type 3).
Le critère $[\gamma_{\bar{\Gamma}} \searrow]$, bien que globalement moins performant que le précédent, permet également de produire des fonctions de décision qui utilisent peu d'exemples et qui sont performantes en généralisation à la fois sur les bases d'apprentissage et de validation. L'utilisation d'une règle simple, en l'occurance la moyenne, pour déterminer un critère de marge à partir de l'ensemble des valeurs de $\Gamma(z)$ pour chaque exemple z produit globalement de bons résultats et joue en faveur d'une utilisation plus fine des valeurs de $\Gamma(z)$.

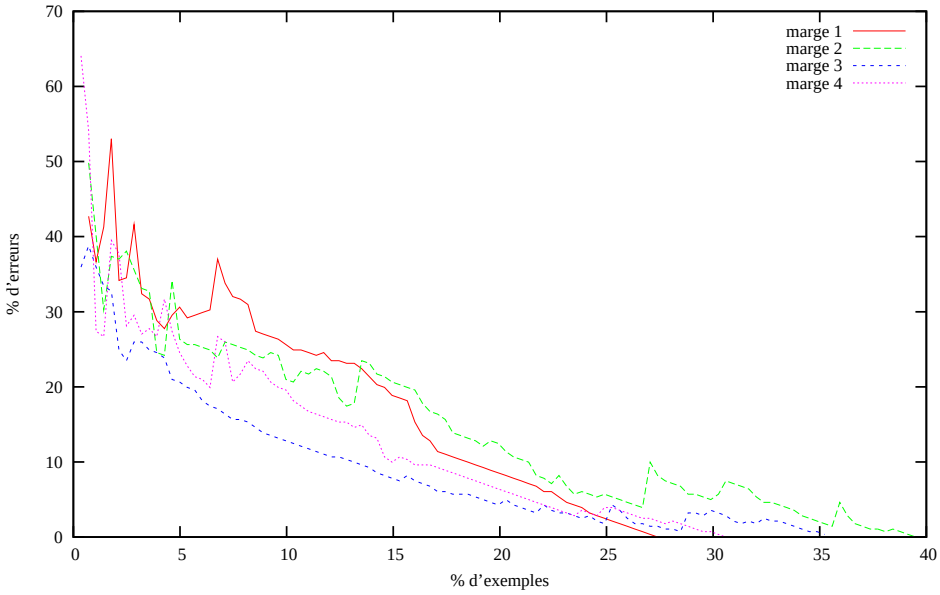


(a) apprentissage

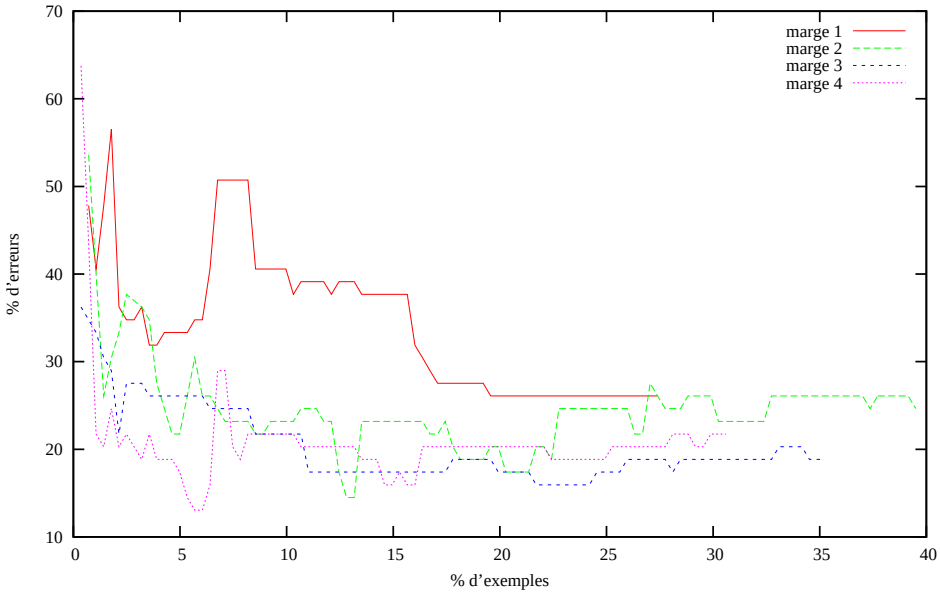


(b) validation

FIG. C.1 – *Taux d'erreur des 4 critères de marge (1 :* $[\gamma \nearrow]$*, 2 :* $[\gamma \searrow]$*, 3 :* $[\gamma_{\odot} \searrow]$ *et 4 :*$[\gamma_{\bar{\Gamma}} \searrow]$*) suivant le pourcentage d'exemples utilisé pour la base* **bupa**.

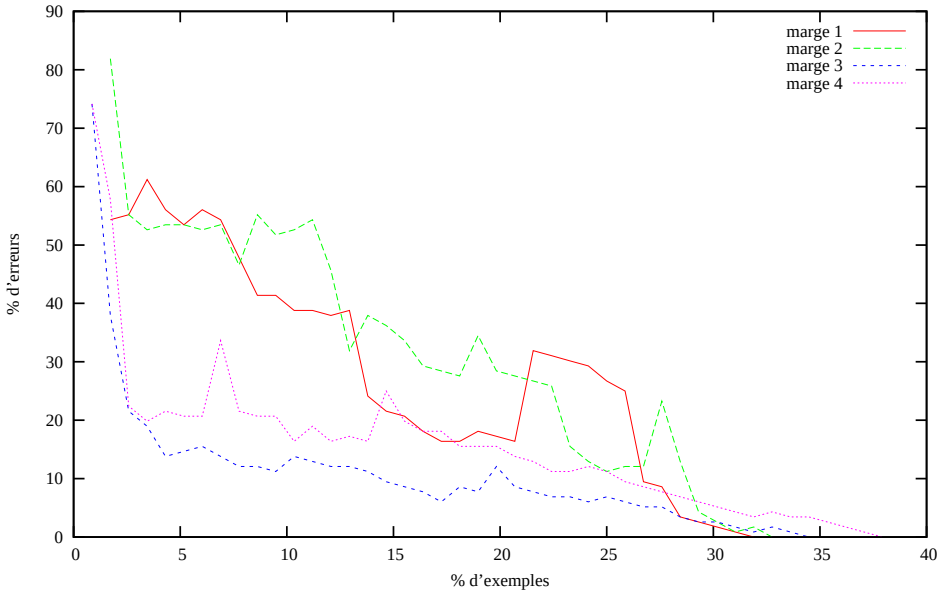


(a) apprentissage

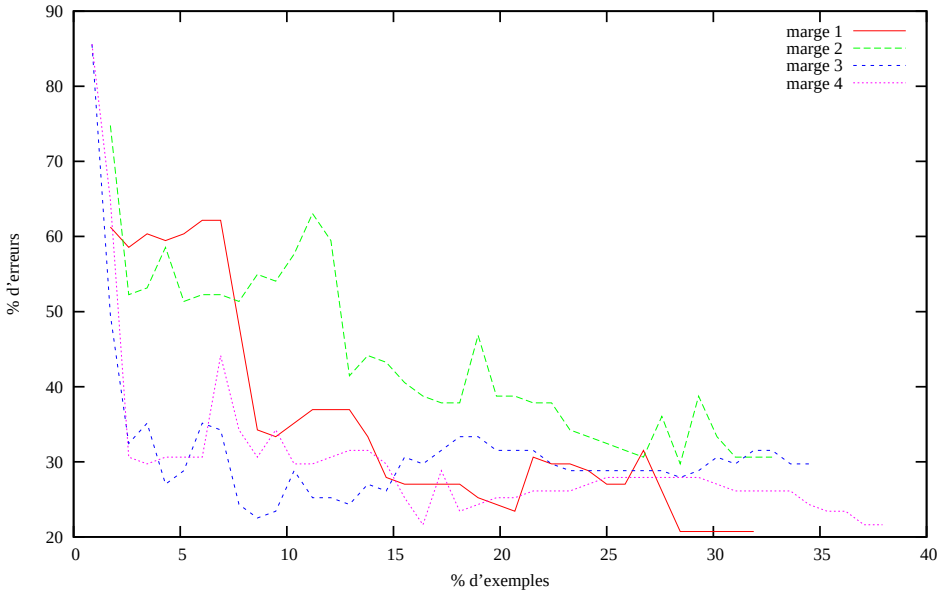


(b) validation

FIG. C.2 – *Taux d'erreur des 4 critères de marge (1 :* $[\gamma \nearrow]$*, 2 :* $[\gamma \searrow]$*, 3 :* $[\gamma_{\odot} \searrow]$ *et 4 :*$[\gamma_{\bar{\Gamma}} \searrow]$*) suivant le pourcentage d'exemples utilisé pour la base* **ionosphere**.

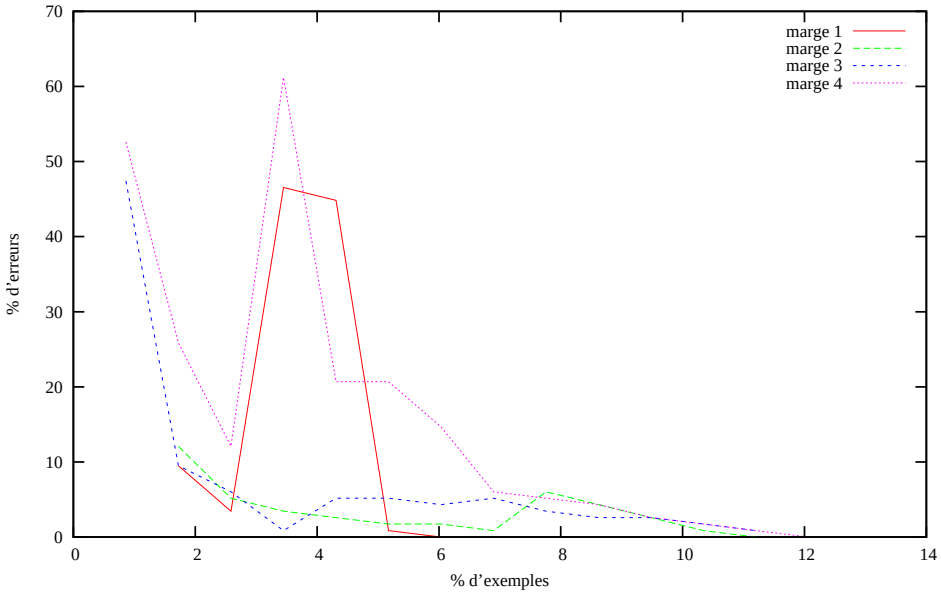


(a) apprentissage

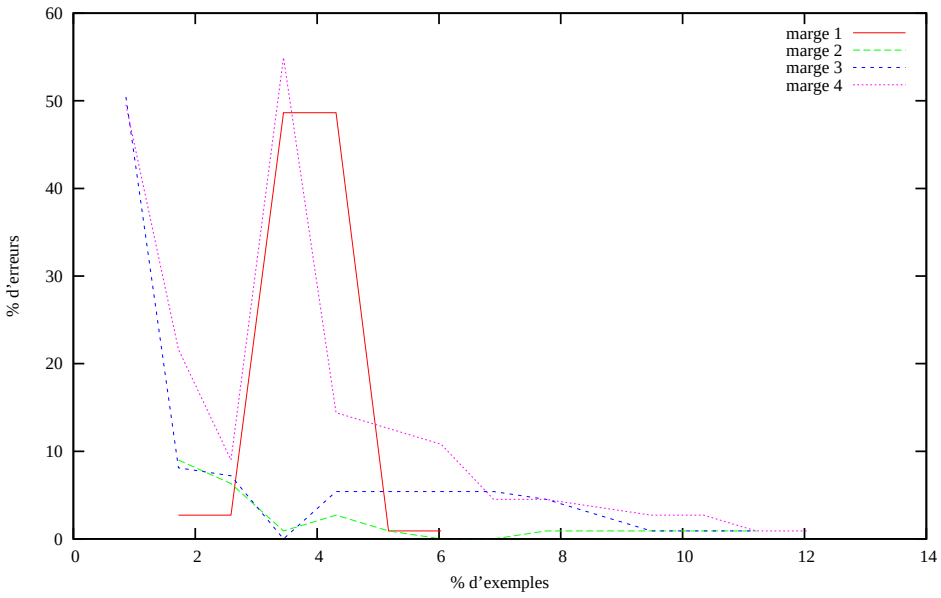


(b) validation

FIG. C.3 – *Taux d'erreur des 4 critères de marge (1 :* $[\gamma \nearrow]$*, 2 :* $[\gamma \searrow]$*, 3 :* $[\gamma_{\odot} \searrow]$ *et 4 :*$[\gamma_{\bar{\Gamma}} \searrow]$*) suivant le pourcentage d'exemples utilisé pour la base* **QIjpeg2M**.

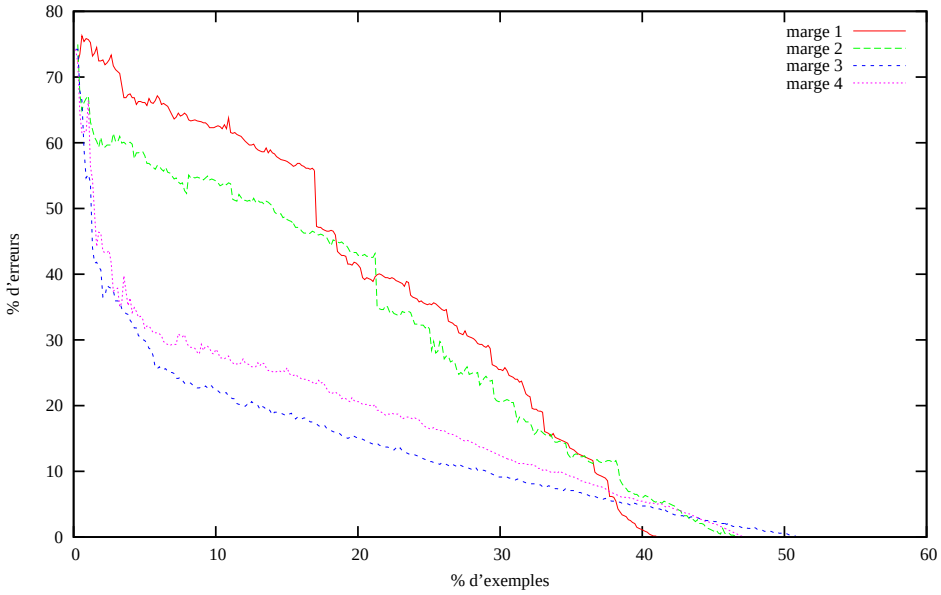


(a) apprentissage

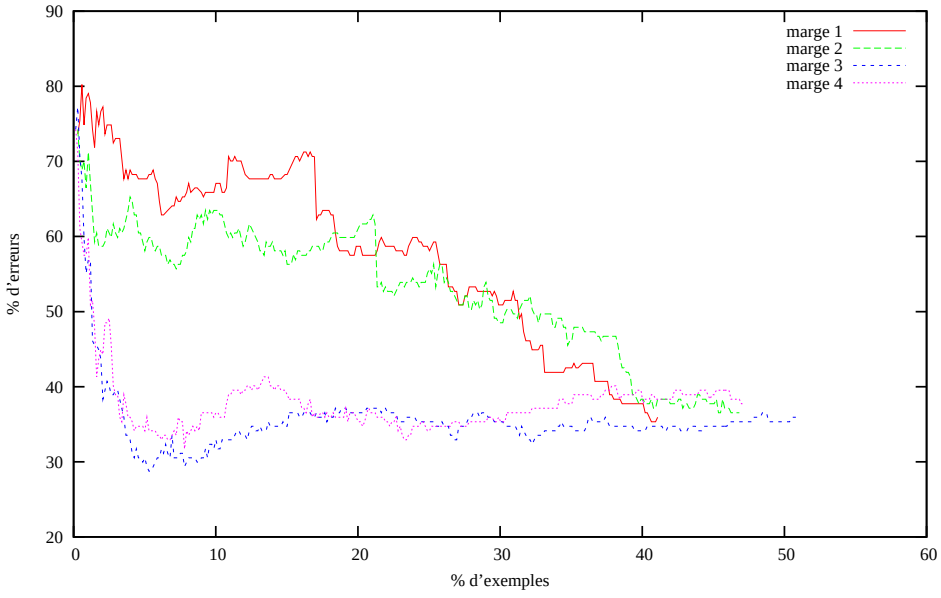


(b) validation

FIG. C.4 – *Taux d'erreur des 4 critères de marge (1 :* $[\gamma \nearrow]$*, 2 :* $[\gamma \searrow]$*, 3 :* $[\gamma_{\odot} \searrow]$ *et 4 :*$[\gamma_{\bar{\Gamma}} \searrow]$*) suivant le pourcentage d'exemples utilisé pour la base* **QIjpeg2M-2c**.

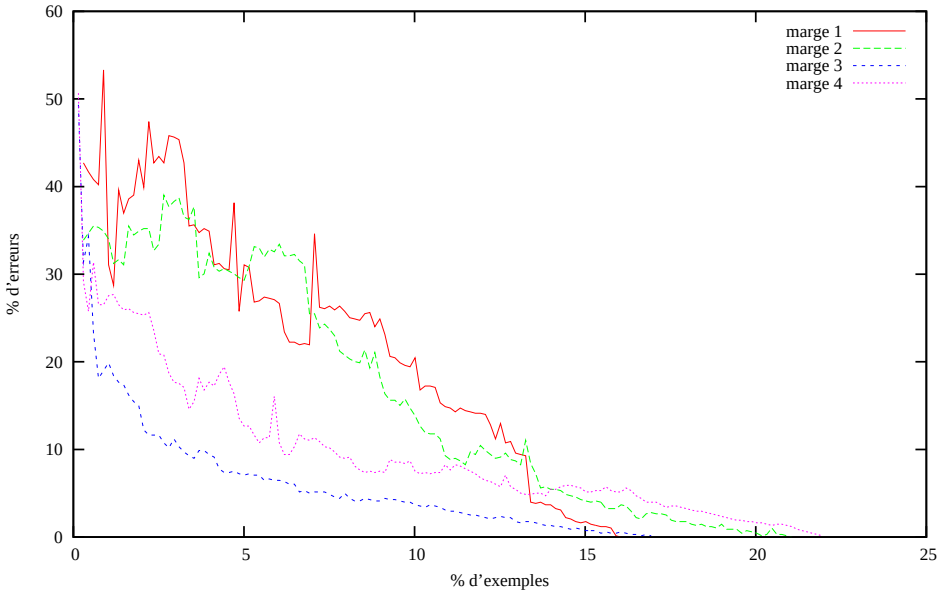


(a) apprentissage

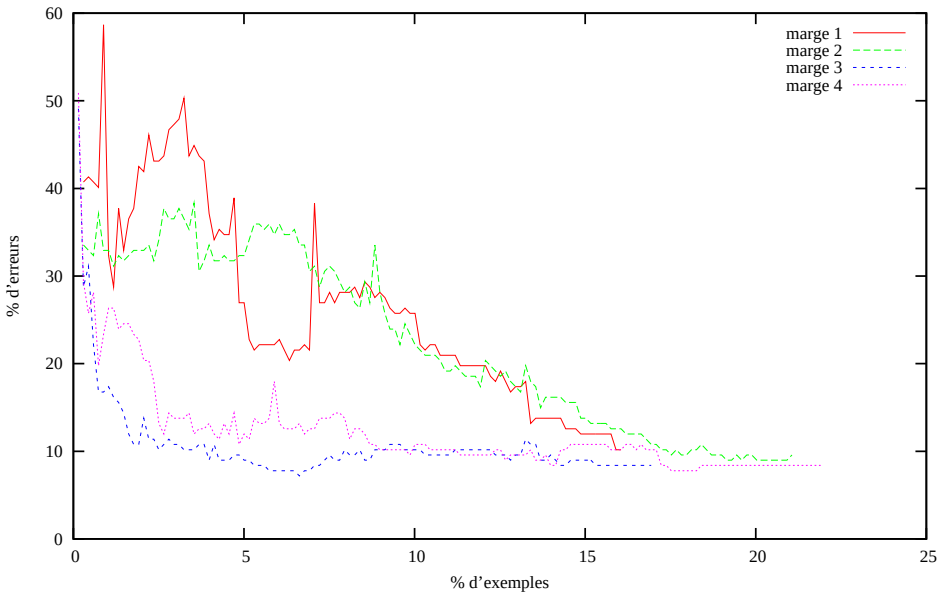


(b) validation

FIG. C.5 – *Taux d'erreur des 4 critères de marge (1 :* $[\gamma \nearrow]$*, 2 :* $[\gamma \searrow]$*, 3 :* $[\gamma_{\odot} \searrow]$ *et 4 :*$[\gamma_{\bar{\Gamma}} \searrow]$*) suivant le pourcentage d'exemples utilisé pour la base* **vehicle**.

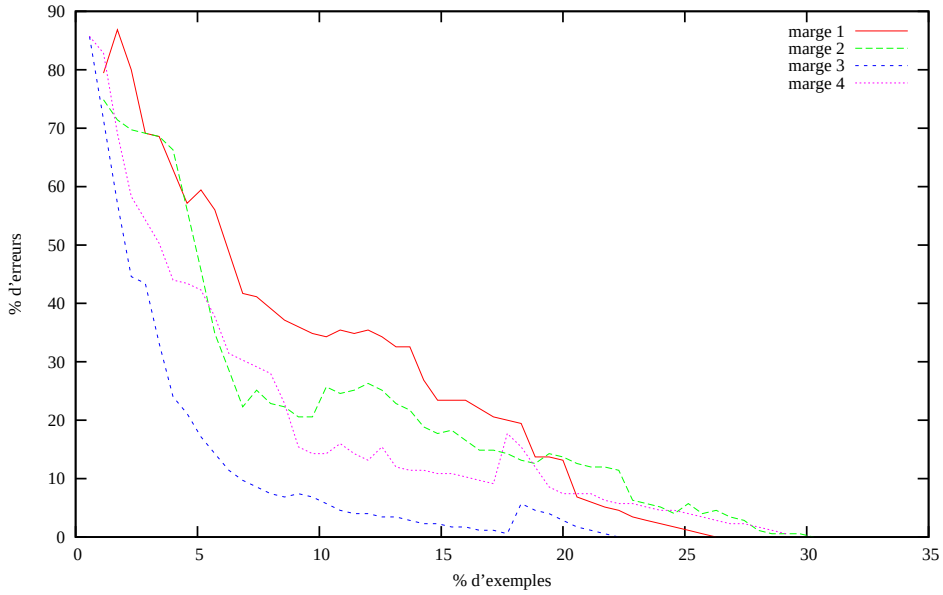


(a) apprentissage

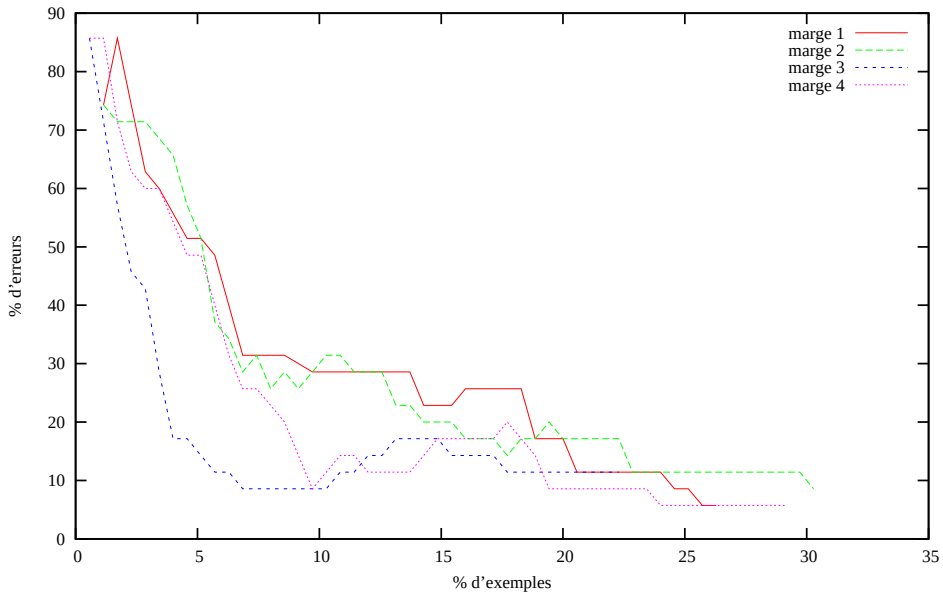


(b) validation

FIG. C.6 – *Taux d'erreur des 4 critères de marge (1 :* $[\gamma \nearrow]$*, 2 :* $[\gamma \searrow]$*, 3 :* $[\gamma_{\odot} \searrow]$ *et 4 :*$[\gamma_{\bar{\Gamma}} \searrow]$*) suivant le pourcentage d'exemples utilisé pour la base* **vehicle-2c**.

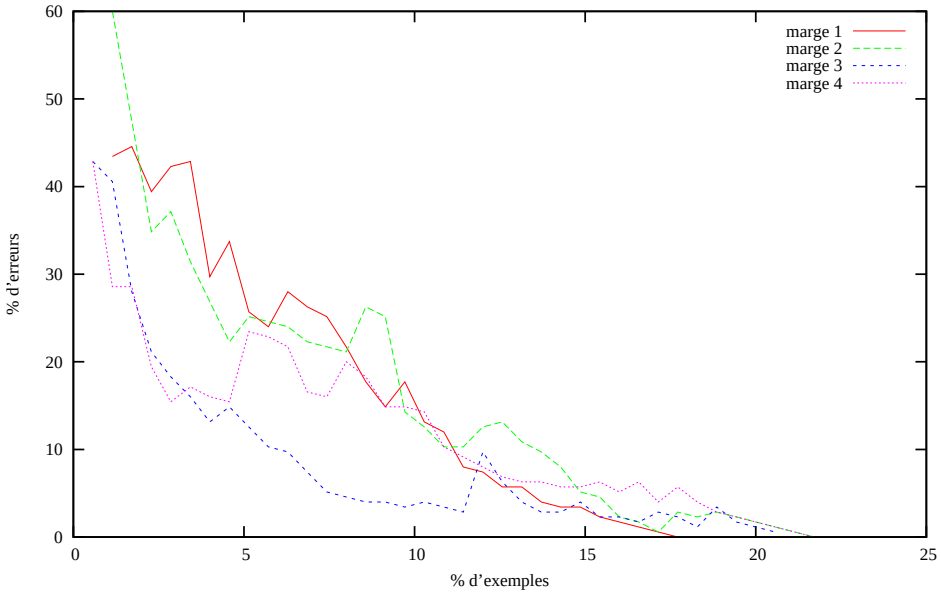


(a) apprentissage

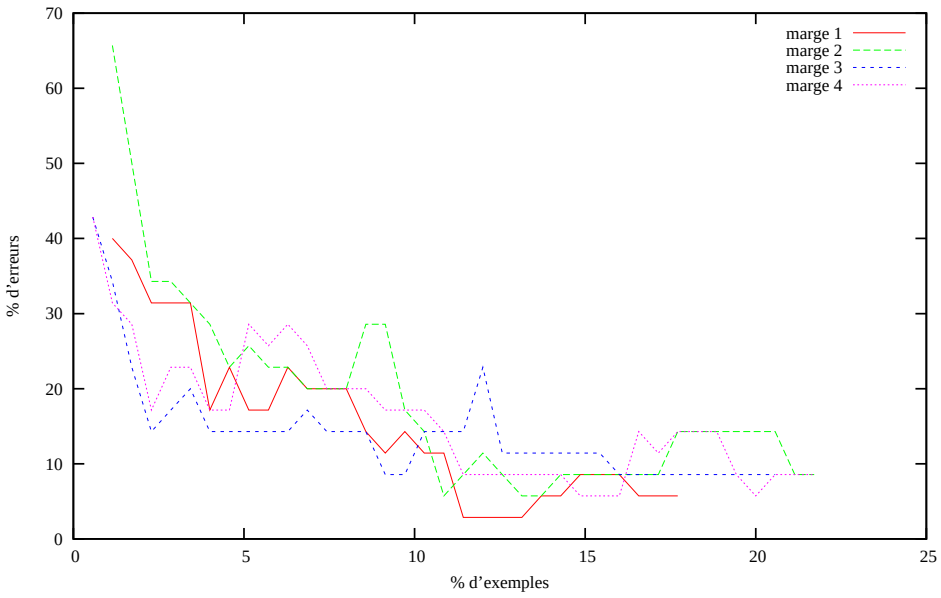


(b) validation

FIG. C.7 – *Taux d'erreur des 4 critères de marge (1 :* $[\gamma \nearrow]$*, 2 :* $[\gamma \searrow]$*, 3 :* $[\gamma_{\odot} \searrow]$ *et 4 :*$[\gamma_{\bar{\Gamma}} \searrow]$*) suivant le pourcentage d'exemples utilisé pour la base* **Segment**.

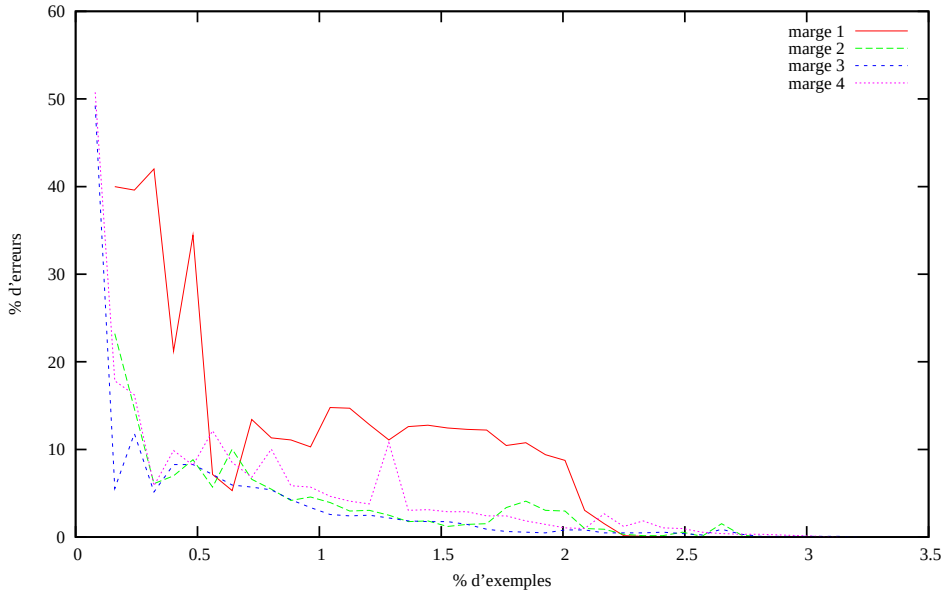


(a) apprentissage

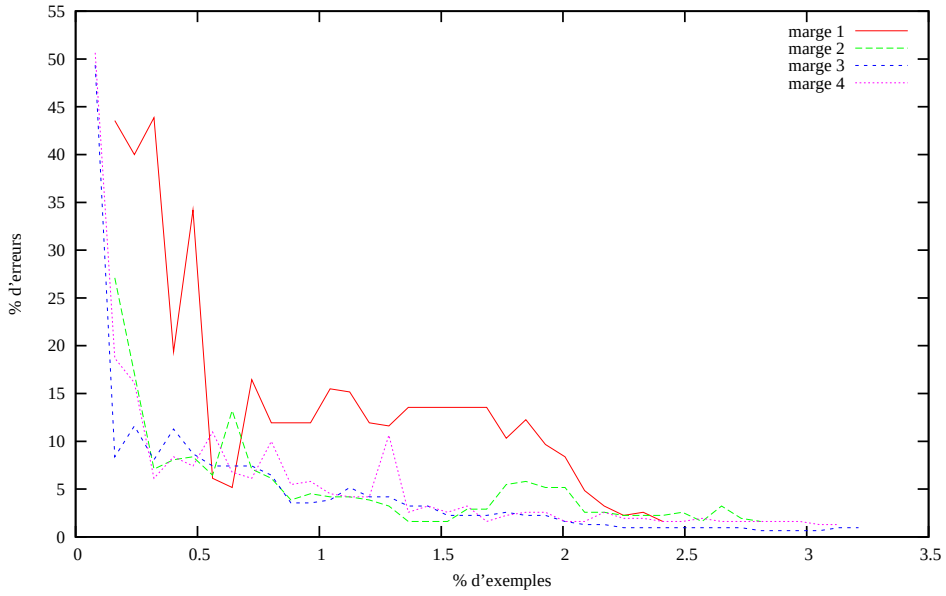


(b) validation

FIG. C.8 – *Taux d'erreur des 4 critères de marge (1 :* $[\gamma \nearrow]$*, 2 :* $[\gamma \searrow]$*, 3 :* $[\gamma_{\odot} \searrow]$ *et 4 :*$[\gamma_{\bar{\Gamma}} \searrow]$*) suivant le pourcentage d'exemples utilisé pour la base* **Segment-2c**.

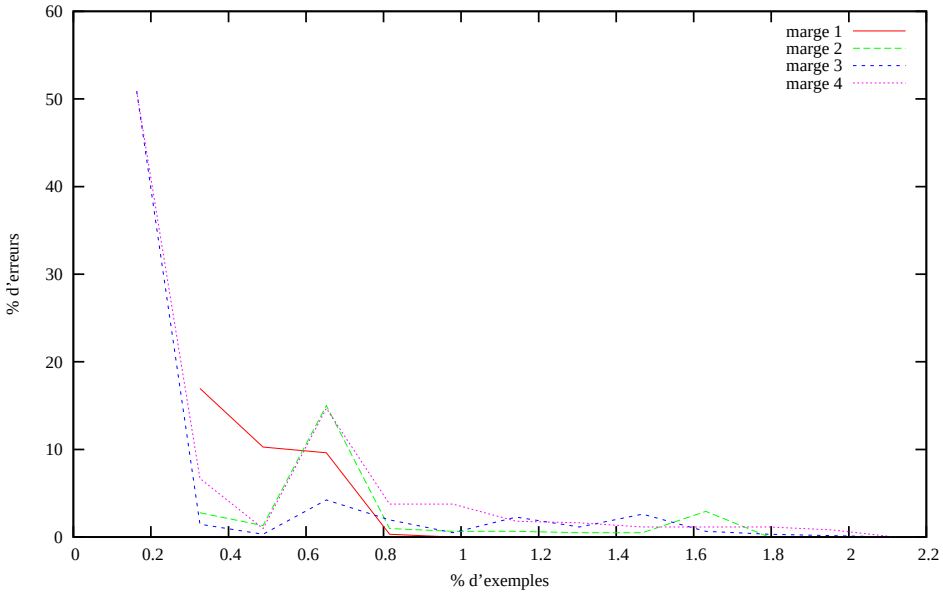


(a) apprentissage

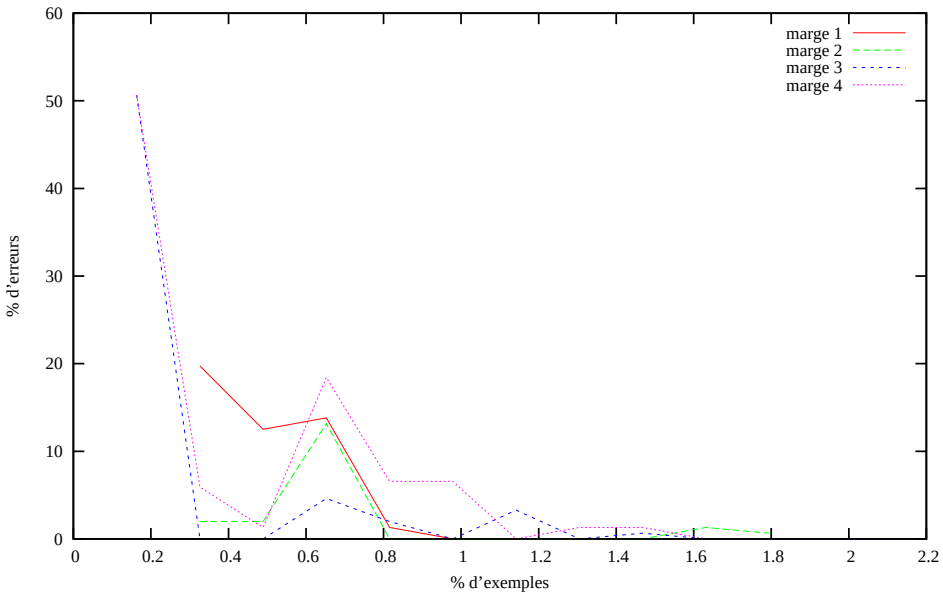


(b) validation

FIG. C.9 – *Taux d'erreur des 4 critères de marge (1 :* $[\gamma \nearrow]$*, 2 :* $[\gamma \searrow]$*, 3 :* $[\gamma_{\odot} \searrow]$ *et 4 :*$[\gamma_{\bar{\Gamma}} \searrow]$*) suivant le pourcentage d'exemples utilisé pour la base* **letter-2c**.



(a) apprentissage



(b) validation

FIG. C.10 – *Taux d'erreur des 4 critères de marge (1 :* $[\gamma \nearrow]$*, 2 :* $[\gamma \searrow]$*, 3 :* $[\gamma_{\odot} \searrow]$ *et 4 :*$[\gamma_{\bar{\Gamma}} \searrow]$*) suivant le pourcentage d'exemples utilisé pour la base* **OptDigits-2c**.

ANNEXE D

PUBLICATIONS DE L'AUTEUR

Revues internationnales

1. A comparison of supervised pixels-based color image segmentation methods. Application in cancerology.
   C. Meurie, *G. Lebrun*, O. Lezoray, A. Elmoataz
   WSEAS Transactions on Computers, ISSN 1109-2750, Issue 3, Vol.2, pp.739-744, July 2003.

Communications internationales avec actes et comité de lecture

1. A new model selection method for SVM.
   G. Lebrun, O.Lezoray, C. Charrier, H. Cardot
   7th International Conference on Intelligent Data Engineering and Automated Learning, Volume LNCS 4224, pp. 99-107, SPAIN, September 2006.

2. Speed-up LOO-CV with SVM classifier.
   G. Lebrun, O.Lezoray, C. Charrier, H. Cardot
   7th International Conference on Intelligent Data Engineering and Automated Learning, Volume LNCS 4224, pp. 108-115, SPAIN, September 2006.

3. A Machine Learning-Based Color Image Quality Metric.
   C. Charrier, *G. Lebrun*, O. Lezoray
   CGIV, pp. 251-256, Leeds, Royaume Uni, 2006.

4. Fast pixel classification by SVM using Vector Quantization, Tabu Search and hybrid Color Space.
   G. Lebrun, C. Charrier, O.Lezoray, C. Meurie, H. Cardot
   International Conference on Computer Analysis of Images and Patterns, Volume LNCS 3691, ISSN 0302-9743, pp. 685-692, France, Septembre 2005.

5. Fusion de segmentation d'images microscopiques par SVM dans différents espaces couleur selon la théorie de l'évidence.
   C. Charrier, *G. Lebrun*, O. Lezoray

Gretsi conference on signal and image processing, Belgique, Septembre 2005.

6. SVM Training Time Reduction using Vector Quantization
   G. Lebrun, C. Charrier, H. Cardot,
   17th International Conference on Pattern Recognition, pp. 160-163, Angleterre, Août 2004.

7. A supervised segmentation scheme for cancerology color images.
   C. Meurie, *G. Lebrun*, O. Lezoray, A. Elmoataz
   IEEE International Symposium on Signal Processing and Information Technology, Germany, December 2003.

8. A comparison of supervised pixels-based color image segmentation methods. Application in cancerology.
   C. Meurie, *G. Lebrun*, O. Lezoray, A. Elmoataz
   WSEAS International Conference On Signal, Speech and Image Processing, Greece, October 2003.

Communications nationales avec actes et comité de lecture

1. Classification rapide de pixels par SVM dans un espace couleur hybride.
   G. Lebrun, C. Charrier, O. Lezoray, C. Meurie, H. Cardot
   COmpression et REprésentation des Signaux Audiovisuels, pp. 231-236 , Renne, Novembre 2005.

2. Construction de fonctions de décision performantes et de complexités réduites avec des SVM.
   G. Lebrun, Christophe Charrier, Olivier Lezoray, Hubert Cardot
   RJCIA (Rencontres Jeunes Chercheurs en Intelligence Artificielle), pp. 85-98, Nice, Mai 2005.

3. Réduction du temps d'apprentissage des SVM par Quantification Vectorielle.
   G. Lebrun, C. Charrier, O. Lezoray, H. Cardot
   CORESA (COmpression et REprésentation des signaux Audiovisuels), pp. 223-226, Lille, Mai 2004.

BIBLIOGRAPHIE

[ABE05] S. ABE. « Support Vector Machines for Pattern Classification ». Springer, 2005.

[AL-ANI05] A. AL-ANI. « Feature Subset Selection Using Ant Colony Optimization ». *International Journal of Computational Intelligence*, 2(1), pages 53–58, 2005.

[ALLWEI00] E. L. ALLWEIN, R. E. SCHAPIRE & Y. SINGER. « Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers. ». *Journal of Machine Learning Research*, vol. 1, pages 113–141, 2000.

[ANGIUL05] F. ANGIULLI. « Fast condensed nearest neighbor rule ». Dans *the 22nd international conference on Machine learning*, pages 25–32, 2005.

[AYAT02] N.-E. AYAT, M. CHERIET & C. Y. SUEN. « KMOD - A Two-Parameter SVM Kernel for Pattern Recognition. ». Dans *ICPR (3)*, pages 331–334, 2002.

[AYAT05] N.-E. AYAT, M. CHERIET & C. Y. SUEN. « Automatic model selection for the optimization of SVM kernels. ». *Pattern Recognition*, 38(10), pages 1733–1745, 2005.

[AZZAG04] H. AZZAG, C. GUINOT & G. VENTURINI. « How to Use Ants for Hierarchical Clustering. ». Dans *ANTS Workshop*, pages 350–357, 2004.

[BABICH96] G. A. BABICH & O. I. CAMPS. « Weighted Parzen Windows for Pattern Classification. ». *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(5), pages 567–570, 1996.

[BANZHA98] W. BANZHAF, P. NORDIN, R. E. KELLER & F. D. FRANCONE. « Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and Its Applications ». Morgan Kaufmann, 1998.

[BELLMA61] R. E. BELLMAN. « Adaptive Control Processes ». Princeton University Press, 1961.

[BEN-HU01] A. BEN-HUR, D. HORN, H. T. SIEGELMANN & V. VAPNIK. « Support Vector Clustering ». *Journal of Machines Learning Research*, vol. 2, 2001.

[BENTLE75] J. L. BENTLEY. « Multidimensional Binary Search Trees Used for Associative Searching ». *Communication of the ACM*, 18(9), pages 509–517, 1975.

[BEZDEK98] J. C. BEZDEK, T. R. REICHHERZER, G. S. LIM & Y. ATTIKIOUZEL. « Multiple-prototype classifier design ». *IEEE Transactions on Systems, Man, and Cybernetics*, 28(1), pages 67–79, 1998.

[BISHOP92] C. BISHOP. « Neural networks for pattern recognition ». Oxford University Press, 1992.

[BLAKE98] C. BLAKE & C. MERZ. « UCI repository of machine learning databases. Advances in Kernel Methods, Support Vector Learning. ». Dans *University of California, Irvine, Dept. of Information and Computer Sciences*, 1998.

[BLUM97] A. L. BLUM & P. LANGLEY. « Selection of relevant features and examples in machine learning ». *Artificial Intelligence*, 97(1-2), pages 245–271, 1997.

[BONABE99] E. BONABEAU, M. DORIGO & G. THERAULAZ. « Swarm Intelligence, From Natural to Artificial Systems ». Oxford University Press, 1999.

[BORDES05] A. BORDES, S. ERTEKIN, J. WESTON & L. BOTTOU. « Fast Kernel Classifiers with Online and Active Learning ». *Journal of Machine Learning Research*, vol. 6, pages 1579–1619, 2005.

[BOSER92] B. E. BOSER, I. GUYON & V. VAPNIK. « A Training Algorithm for Optimal Margin Classifiers ». Dans *Computational Learing Theory*, pages 144–152, 1992.

[BREIMA93] L. BREIMAN, J. FREIDMAN, R. OLSHEN & C. STONE. « Classification And Regression Trees ». Wadsworth and Brooks, 1993.

[BURBID02] R. BURBIDGE. « Stopping Criteria for SVMs ». Available at http://stats.ma.ic.ac.uk /rdb/public_html/pubs/hermes.pdf, 2002.

[BURGES96] C. J. C. BURGES. « Simplified Support Vector Decision Rules. ». Dans *ICML*, pages 71–77, 1996.

[BURGES98] C. J. C. BURGES. « A Tutorial on Support Vector Machines for Pattern Recognition. ». *Data Min. Knowl. Discov.*, 2(2), pages 121–167, 1998.

[CALLUT05] J. CALLUT & P. DUPONT. « Séparateurs à Vaste Marge Optimisant la Fonction F_{beta}. ». Dans *CAP*, pages 79–92, 2005.

[CANO03] J. R. CANO, F. HERRERA & M. LOZANO. « Using evolutionary algorithms as instance selection for data reduction in KDD: an experimental study. ». *IEEE Trans. Evolutionary Computation*, 7(6), pages 561–575, 2003.

[CAUWEN00] G. CAUWENBERGHS & T. POGGIO. « Incremental and Decremental Support Vector Machine Learning. ». Dans *NIPS*, pages 409–415, 2000.

[CAWLEY01] G. C. CAWLEY. « Model Selection for Support Vector Machines via Adaptive Step-Size Tabu Search ». Dans *ICANNGA*, 2001.

[CERVER01] V. CERVERÓN & F. J. FERRI. « Another move toward the minimum consistent subset: a tabu search approach to the condensed nearest neighbor rule. ». *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 31(3), pages 408–413, 2001.

[CHANG01] C.-C. CHANG & C.-J. LIN. « LIBSVM: a library for support vector machines ». Sofware Available at http://www.csie.ntu.edu.tw/˜cjlin/libsvm, 2001.

[CHAPEL02] O. CHAPELLE, V. VAPNIK, O. BOUSQUET & S. MUKHERJEE. « Choosing Multiple Parameters for Support Vector Machines ». Dans *Machine Learning*, 46(1-3), pages 131–159, 2002.

[CHAPEL04] O. CHAPELLE. « Support Vector Machines : Principes d'induction, Réglage automatique et Connaissances à priori ». Thèse de doctorat, Université Pierre et Marie Curie , Paris VI, avril 2004.

[CHARRI98] C. CHARRIER. « Vers l'optimisation statistique et perceptuelle de la qualité pour la compression des images couleur par quantification vectorielle ». Thèse de doctorat, Université Jean Monnet, 1998.

[CHARRI06A] C. CHARRIER, G. LEBRUN & O. LEZORAY. « Fusion of svm-based microscopic color images through colorimetric transformation. ». Dans *ICASSP*, 2006.

[CHARRI06B] C. CHARRIER, G. LEBRUN & O. LEZORAY. « A Machine Learning-Based Color Image Quality Metric ». Dans *CGIV*, 2006.

[CHEN01] Y. CHEN, X. S. ZHOU & T. HUANG. « One-class SVM for learning in image retrieval. ». Dans *Image Processing*, vol. 1, pages 34–37, 2001.

[CHEN05A] Y.-W. CHEN & C.-J. LIN. « Combining SVMs with various feature selection strategies ». Dans I. GUYON, S. GUNN, M. NIKRAVESH & L. ZADEH, rédacteurs, *Feature extraction, foundations and applications*. Springer, 2005.

[CHEN05B] Y. CHEN & C. LIN. « Combining SVMs with various feature selection strategies ». Dans I. GUYON, S. GUNN, M. NIKRAVESH & L. ZADEH, rédacteurs, *Feature extraction, foundations and applications*, Springer, 2005.

[CHOU03] Y.-Y. CHOU & L. G. SHAPIRO. « A hierarchical multiple classifier learning algorithm. ». *Pattern Anal. Appl.*, 6(2), pages 150–168, 2003.

[COELLO02] C. A. C. COELLO & G. B. L. D. A. V. VELDHUIZEN. « Evolutionary Algorithms for Solving Multi-Objective Problems (volume 5) ». Kluwer Academic, 2002.

[COLLET05] Y. COLLETTE & P. SIARRY. « Three new metrics to measure the convergence of metaheuristics towards the Pareto frontier and the aesthetic of a set of solutions in biobjective optimization. ». *Computers & OR*, vol. 32, pages 773–792, 2005.

[COLLOB01] R. COLLOBERT & S. BENGIO. « SVMTorch: Support Vector Machines for Large-Scale Regression Problems ». Dans *Journal of Machine Learning Research*, vol. 1, pages 143–160, 2001.

[COLLOB02] R. COLLOBERT, S. BENGIO & Y. BENGIO. « A Parallel Mixture of SVMs for Very Large Scale Problems. ». *Neural Computation*, 14(5), pages 1105–1114, 2002.

[COLORN92] A. COLORNI, M. DORIGO & V. MANIEZZO. « An Investigation of some Properties of an Ant Algorithm. ». Dans *PPSN*, pages 515–526, 1992.

[CORNUÉ02] A. CORNUÉJOLS & L. MICLET. « Apprentissage artificiel ». Eyrolles, 2002.

[COVER67] T. M. COVER & P. E. HART. « Nearest Neighbor Pattern Classification ». *IEEE Transactions on Information Theory*, 13(1), pages 21–27, 1967.

[CRAMME01] K. CRAMMER & Y. SINGER. « On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. ». *Journal of Machine Learning Research*, vol. 2, pages 265–292, 2001.

[CRAMME02] K. CRAMMER, R. GILAD-BACHRACH, A. NAVOT & N. TISHBY. « Margin Analysis of the LVQ Algorithm. ». Dans *NIPS*, pages 462–469, 2002.

[CRISTI99] N. CRISTIANINI, C. CAMPBELL & J. SHAWE-TAYLOR. « Dynamically adapting kernels in support vector machines ». Dans *Proceedings of the 1998 conference on Advances in neural information processing systems II*, pages 204–210, MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-11245-0.

[CRISTI00] N. CRISTIANINI & J. SHAWE-TAYLOR. « An Introduction to Support Vector Machines and other kernel-bases learning methods ». Cambridge University Press, 2000.

[CRISTI01] N. CRISTIANINI, J. SHAWE-TAYLOR, A. ELISSEEFF & J. S. KANDOLA. « On Kernel-Target Alignment. ». Dans *NIPS*, pages 367–373, 2001.

[CUTZU03] F. CUTZU. « Polychotomous Classification with Pairwise Classifiers: A New Voting Principle. ». Dans *Multiple Classifier Systems*, pages 115–124, 2003.

[DALY93] S. DALY. « The Visible Differences Predictor: An Algorithm for the Assessment of Image Fidelity ». Dans *Digital Images and Human Vision*, pages 179–206. The MIT Press Cambridge, 1993.

[DARCY05] Y. DARCY & Y. GUERMEUR. « Radius-margin Bound on the Leave-one-out Error of Multi-class SVMs ». Rapport technique, INRIA, 2005.

[DARWIN59] C. DARWIN. « On the Origin of Species by Means of Natural Selection or the Preservation of Favored Races in the Struggle for Live ». Murrayn, 1859.

[DASARA94] B. V. DASARATHY. « Minimal consistent set (MCS) identification for optimal nearest neighbor decision systems design ». *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 24, pages 511–517, 1994.

[DEBUSE97] J. C. W. DEBUSE & V. J. RAYWARD-SMITH. « Feature Subset Selection within a Simulated Annealing Data Mining Algorithm. ». *J. Intell. Inf. Syst.*, 9(1), pages 57–81, 1997.

[DECOST00] D. DECOSTE & K. WAGSTAFF. « Alpha seeding for support vector machines ». Dans *Int. Conf. Knowledge Discovery Data Mining*, 2000.

[DEVI02] V. S. DEVI & M. N. MURTY. « An incremental prototype set building technique ». *Pattern Recognition*, 35(2), pages 505–513, 2002.

[DEVIJV80] P. A. DEVIJVER. « On the edited nearest neighbor rule ». Dans *International Conference on Pattern Recognition*, pages 72–80, 1980.

[DEVIJV82] P. A. DEVIJVER & J. KITTLER. « Pattern Recognition: A Statistical Approach ». Prentice-Hall, 1982.

[DIETTE95] T. G. DIETTERICH & G. BAKIRI. « Solving Multiclass Learning Problems via Error-Correcting Output Codes. ». *J. Artif. Intell. Res. (JAIR)*, vol. 2, pages 263–286, 1995.

[DONG03] J. X. DONG, C. Y. SUEN & A. KRZYZAK. « A Fast SVM Training Algorithm. ». *IJPRAI*, 17(3), pages 367–384, 2003.

[DONG05] J. X. DONG, A. KRZYZAK & C. Y. SUEN. « Fast SVM Training Algorithm with Decomposition on Very Large Data Sets. ». *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(4), pages 603–618, 2005.

[DREYFU04] G. DREYFUS, M. SAMUELIDES, J. MARTINEZ, M. GORDON, F. BADRAN, S. THIRIA & L. HÉRAULT. « Réseaux de neurones. Méthodologies et applications ». Eyrolles, 2004.

[DRÉO03] J. DRÉO, A. PÉTROWSKI, P. SIARRY & E. TAILLARD. « Métaheuristiques pour l'optimisation difficile ». Groupe Eyrolles, 2003.

[DUAN03] K. DUAN, S. S. KEERTHI & A. N. POO. « Evaluation of simple performance measures for tuning SVM hyperparameters ». *Neurocomputing*, vol. 51, pages 41–59, 2003.

[DUAN05] K.-B. DUAN & S. S. KEERTHI. « Which Is the Best Multiclass SVM Method? An Empirical Study. ». Dans *Multiple Classifier Systems*, pages 278–285, 2005.

[DUDANI76] S. A. DUDANI. « The distance-weighted k-nearest neighbor rule. ». *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 6, pages 325–327, 1976.

[EFRON82] B. EFRON. « The Jackknife, the bootstrap and other resampling methods ». SIAM, 1982.

[EFRON97] B. EFRON & R. TIBSHIRANI. « Improvements on cross-validation: The .632+ bootstrap method ». *Journal of the American Statistical*, vol. 92, pages 548–560, 1997.

[ESHELM97] L. J. ESHELMAN, K. E. MATHIAS & J. D. SCHAFFER. « Crossover Operator Biases: Exploiting the Population Distribution ». Dans *ICGA*, pages 354–361, 1997.

[ESTER96] M. ESTER, H. P. KRIEGEL, J. SANDER & X. XU. « A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise ». *int. Conf. on Knowledge Discovery and Data Mining (KDD'96)*, pages 226–231, 1996.

[FAIGLE37] U. FAIGLE & W. KERN. « Some convergence results for probabilistic tabu search ». *ORSA Journal on Computing*, vol. 4, page 1992, 32–37.

[FAN05] R. E. FAN, P. H. CHEN & C. J. LIN. « Working Set Selection Using the Second Order Information for Training SVM ». *JMLR*, vol. 6, pages 1889–1918, 2005.

[FOGEL66] L. J. FOGEL, A. J. OWENS & M. J. WALSH. « Artifical Intelligence through Simulated Evolution ». Wiley, 1966.

[FORGY65] E. W. FORGY. « Cluster Analysis of Multivariable Data: Efficiency versus Interpretability Models ». Dans *Biometrics*, 61(3), pages 768–769, 1965.

[FRANK04] E. FRANK & S. KRAMER. « Ensembles of nested dichotomies for multi-class problems. ». Dans *ICML*, pages 305–312, 2004.

[FRÖH04] H. FRÖHLICH, O. CHAPELLE & B. SCHÖLKOPF. « Feature Selection for Support Vector Machines Using Genetic Algorithms ». *IJAIT*, 13(4), pages 791–800, 2004.

[FUKUNA89] K. FUKUNAGA & R. R. HAYES. « The reduced Parzen classifier ». *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, pages 423–425, 1989.

[FÜRNK01] J. FÜRNKRANZ. « Round Robin Rule Learning. ». Dans *ICML*, pages 146–153, 2001.

[FÜRNK02] J. FÜRNKRANZ. « Pairwise Classification as an Ensemble Technique. ». Dans *ECML*, pages 97–110, 2002.

[GAGNÉ05A] C. GAGNÉ. « Algorithmes évolutionnaires appliqués à la reconnaissance des formes et à la conception optique ». Thèse de doctorat, Université de Laval, Québec, Mai 2005.

[GAGNÉ05B] C. GAGNÉ. « Algorithmes évolutionnaires appliqués à la reconnaissance des formes et à la conception optique ». Thèse de doctorat, Université de Laval, Québec, 2005.

[GERSHO92] A. GERSHO & R. M. GRAY. « Vector Quantization and Signal Compression ». Kluwer, 1992.

[GILAD-04] R. GILAD-BACHRACH, A. NAVOT & N. TISHBY. « Margin based feature selection - theory and algorithms. ». Dans *ICML*, 2004.

[GLOVER89A] F. GLOVER. « Tabu search: part I ». Dans *on Computing*, 1(3), pages 190–206, 1989.

[GLOVER89B] F. GLOVER. « Tabu search: part II ». Dans *on Computing*, 2(1), pages 4–32, 1989.

[GLOVER97] F. GLOVER & M. LAGUNA. « Tabu search ». Kluwer Academic Publishers, Boston MA, 1997.

[GLOVER02] F. GLOVER & S. HANAFI. « Tabu search and finite convergence. ». *Discrete Applied Mathematics*, 119(1-2), pages 3–36, 2002.

[GOLDBE94] D. GOLDBERG. « Algorithmes génétiques ». Addison-Wesley France, 1994.

[GRANDV97] Y. GRANDVALET, S. CANU & S. BOUCHERON. « Noise Injection: Theoretical Prospects. ». *Neural Computation*, 9(5), pages 1093–1108, 1997.

[GUÉ04] D. GUÉRIN, F. COINTAULT, C. GÉE & J. GUILLEMI. « Etude de faisabilité d'un système de comptage d'épis de blé par vision ». *Traitement du Signal*, 21(5), pages 549–560, 2004.

[GUERME05] Y. GUERMEUR, A. ELISSEEFF & D. ZELUS. « A comparative study of multi-class support vector machines in the unifying framework of large margin classifiers: Research Articles ». *Appl. Stoch. Model. Bus. Ind.*, 21(2), pages 199–214, 2005.

[GUHA98] S. GUHA, R. RASTOGI & K. SHIM. « CURE: an efficient clustering algorithm for large databases ». Dans *ACM SIGMOD International Conference on Management of Data*, pages 73–84, June 1998.

[GUIGUE05] V. GUIGUE, A. RAKOTOMAMONJY & S. CANU. « Estimation de signaux par noyaux d'ondelettes ». Dans *20e colloque GRETSI sur le traitement du signal et des images*, pages 1–4, 2005.

[GUYON03] I. GUYON & A. ELISSEEFF. « An Introduction to Variable and Feature Selection. ». *Journal of Machine Learning Research*, vol. 3, pages 1157–1182, 2003.

[HALKID01] M. HALKIDI, Y. BATISTAKIS & M. VAZIRGIANNIS. « On Clustering Validation Techniques ». *Journal of Intelligent Information Systems*, 17(2), pages 107–145, 2001.

[HAMAMO97] Y. HAMAMOTO, S. UCHIMURA & S. TOMITA. « A Bootstrap Technique for Nearest Neighbor Classifier Design. ». *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(1), pages 73–79, 1997.

[HAO99] J.-K. HAO, P. GALINIER & M. HABIB. « Métaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes ». Dans *Revue d'Intelligence Artificielle*, 13(2), pages 283–324, 1999.

[HART68] P. E. HART. « The condensed nearest neighbor rule ». *IEEE Transactions on Information Theory*, vol. 16, pages 515–516, 1968.

[HASTIE97] T. HASTIE & R. TIBSHIRANI. « Classification by Pairwise Coupling. ». Dans *NIPS*, pages 507–513, 1997.

[HASTIE04A] T. HASTIE, S. ROSSET, R. TIBSHIRANI & J. ZHU. « The Entire Regularization Path for the Support Vector Machine ». *Journal of Machine Learning Research*, vol. 5, pages 1391–1415, 2004. ISSN 1533-7928.

[HASTIE04B] T. HASTIE, S. ROSSET, R. TIBSHIRANI & J. ZHU. « The Entire Regularization Path for the Support Vector Machine. ». Dans *NIPS*, 2004.

[HAYKIN99] S. HAYKIN. « Neural Networks: a comprehensive foundation ». Tom Robbins, 1999.

[HERBRI02] R. HERBRICH. « Learning Kernel Classifiers: Theory and Algorithms ». MIT Press, 2002.

[HINNEB98] A. HINNEBURG & D. A. KEIM. « An Efficient Approach to Clustering in Large Multimedia Databases with Noise ». Dans *KDD*, pages 58–65, 1998.

[HINNEB02] A. HINNEBURG & D. A. KEIM. « Apprentissage Artificiel : Acquis, Limites, Enjeux ». Dans *J. Le Maître, ed, Assises 2002: Information - Interaction - Intelligence*, pages 207–222, 2002.

[HÖLLDO90] B. HÖLLDOBLER & E. WILSON. « The Ants ». Springer Verlag, 1990.

[HOLLAN75] J. H. HOLLAND. « Adaptation in Natural and Artificial Systems ». Thèse de doctorat, University of Michigan, 1975.

[HOLLAN92] J. H. HOLLAND. « Adaptation in Natural and Artificial Systems, 2nd edition ». MIT Press, 1992.

[HOLMST97] L. HOLMSTROM, P. KOISTINEN, J. LAAKSONEN & E. OJA. « Neural and statisitcal classifiers - taxonomy and two case studies ». *IEEE Transactions on Neural Networks*, 8(1), pages 5–17, 1997.

[HSU02] C.-W. HSU & C.-J. LIN. « A Comparison of Methods for Multiclass Support Vector Machines ». Dans *IEEE Transactions in Neural Networks*, 13(2), pages 415–425, 2002.

[HULL94] J. J. HULL. « A Database for Handwritten Text Recognition Research. ». *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(5), pages 550–554, 1994.

[HUSH03] D. R. HUSH & C. SCOVEL. « Polynomial-Time Decomposition Algorithms for Support Vector Machines. ». *Machine Learning*, 51(1), pages 51–71, 2003.

[JAIN88] A. K. JAIN & R. C. DUBES. « Algorithms for Clustering Data ». Prentice Hall, 1988.

[JAIN97] A. K. JAIN & D. E. ZONGKER. « Feature Selection: Evaluation, Application, and Small Sample Performance. ». *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(2), pages 153–158, 1997.

[JAIN99] A. K. JAIN, M. N. MURTY & P. J. FLYNN. « Data clustering: a review ». *ACM Computing Surveys*, 31(3), pages 264–323, 1999.

[JAIN00] A. K. JAIN, R. P. W. DUIN & J. MAO. « Statistical Pattern Recognition: A Review. ». *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(1), pages 4–37, 2000.

[JOACHI00] T. JOACHIMS. « Estimating the Generalization Performance of a SVM Efficiently ». Dans *ICML-00*, pages 431–438, 2000.

[JOZWIK83] A. JOZWIK. « A learning scheme for a fuzzy k-nn rule ». *IEEE Pattern Recognition Letters*, vol. 1, pages 287–289, 1983.

[KARACA03] B. KARACALI & H. KRIM. « Fast minimization of structural risk by nearest neighbor rule ». *IEEE Transactions on Neural Networks*, 14(1), pages 127–137, 2003.

[KARACA04] B. KARACALI, R. RAMANATH & W. E. SNYDER. « A comparative analysis of structural risk minimization by support vector machines and nearest neighbor rule. ». *Pattern Recognition Letters*, 25(1), pages 63–71, 2004.

[KARYPI99] G. KARYPIS, E.-H. S. HAN & V. K. NEWS. « Chameleon: Hierarchical Clustering Using Dynamic Modeling ». *Computer*, 32(8), pages 68–75, 1999.

[KASS80] G. V. KASS. « An Exploratory Technique for Investigating Large Quantities of Categorical Data ». *Applied Statistics*, vol. 29, pages 119–127, 1980.

[KATAGI06] S. KATAGIRI & S. ABE. « Incremental training of support vector machines using hyperspheres ». *Pattern Recognition Letters*, 27(13), pages 1495–1507, 2006.

[KEERTH01] S. S. KEERTHI, S. K. SHEVADE, C. BHATTACHARYYA & K. R. K. MURTHY. « Improvements to Platt's SMO Algorithm for SVM Classifier Design ». *Neural computation*, vol. 13, pages 637–649, 2001.

[KEERTH06] S. S. KEERTHI, O. CHAPELLE & D. DECOSTE. « Building Support Vector Machines with Reduced Classifier Complexity ». *JMLR*, vol. 7, pages 1493–1515, 2006.

[KELLER85] J. M. KELLER, M. R. GRAY & J. A. GIVENS. « A fuzzy k-nearest neighbor algorithm ». *IEEE Transactions on Systems, Man, and Cybernetics*, 15(4), pages 580–585, 1985.

[KOGGAL04] R. KOGGALAGE & S. HALGAMUGE. « Reducing the number of training samples for fast support vector machine classification. ». *Neural Information Processing. Letters and Reviews*, 2(3), pages 57–65, 2004.

[KOHAVI97A] R. KOHAVI & G. H. JOHN. « Wrappers for Feature Subset Selection ». Dans *Artificial Intelligence*, 97(1-2), pages 273–324, 1997.

[KOHAVI97B] R. KOHAVI & G. H. JOHN. « Wrappers for feature subset selection ». *Artificial Intelligence*, 97(1-2), pages 273–324, 1997.

[KOHONE01] T. KOHONEN. « Self-organization maps ». Springer Series in Information Sciences, Vol. 30, Third Extended Edition, 2001.

[KORYCI04] D. KORYCINSKI, M. M. CRAWFORD & J. W. BARNES. « Adaptive feature selection for hyperspectral data analysis ». Dans *SPIE*, vol. 5238, pages 213–225, 2004.

[KOWALI90] P. KOWALISKI. « Vision et mesure de la couleur ». Physique fondamentale et appliquée. Masson, 2ème édition, 1990.

[KOZA92] J. R. KOZA. « Genetic Programming: On the programming of computers by means of Natural Selection ». MIT Press, Cambridge, 1992.

[KRAUSK82] J. KRAUSKOPF, D. R. WILLIAMS & D. W. HEELEY. « Cardinal Directions of Color Space ». *Vision Research*, vol. 22, pages 1123–1131, 1982.

[KREBEL99] U. H.-G. KREBEL. « Pairwise classification and support vector machines ». *Advances in kernel methods: support vector learning*, pages 255–268, 1999.

[KULLBA51] S. KULLBACK & R. A. LEIBLER. « On information and sufficiency ». *Annals of Mathematical Statistics*, vol. 22, pages 79–86, 1951.

[KUNCHE97A] L. I. KUNCHEVA. « Editing for the k-nearest neigbors rule by genetic algorithm ». *Pattern Recognition Letters*, vol. 30, pages 1041–1049, 1997.

[KUNCHE97B] L. I. KUNCHEVA. « Fitness functions in editing k-NN reference set by genetic algorithms. ». *Pattern Recognition*, 30(6), pages 1041–1049, 1997.

[KUNCHE99] L. KUNCHEVA & L. C. JAIN. « Nearest neighbor classifier: Simultaneous editing and feature selection. ». *Pattern Recognition Letters*, 20(11-13), pages 1149–1156, 1999.

[KUNCHE04] L. I. KUNCHEVA. « Combining Pattern Classifiers: Methods and Algorithms ». Wiley, 2004.

[KWOK03] J. T. KWOK & I. W. TSANG. « Learning with Idealized Kernels. ». Dans *ICML*, pages 400–407, 2003.

[LABORA02] LABORATORY FOR IMAGE & VIDEO ENGINEERING, UNIVERSITY OF TEXAS (AUSTIN). « LIVE Image Quality Assessment Database ». *http://live.ece.utexas.edu/research/Quality*, 2002.

[LEBRUN04A] G. LEBRUN, C. CHARRIER & H. CARDOT. « SVM training time reduction using Vector Quantization ». Dans *ICPR*, vol. 1, pages 160–163, 2004.

[LEBRUN04B] G. LEBRUN, C. CHARRIER & O. LEZORAY. « Réduction du temps d'apprentissage des SVM par Quantification Vectorielle. ». Dans *CORESA*, pages 223–226, 2004.

[LEBRUN05A] G. LEBRUN, C. CHARRIER, O. LEZORAY & H. CARDOT. « Construction de fonctions de décision performantes et de complexités réduites avec des SVM ». Dans *RJCIA (Rencontres Jeunes Chercheurs en Intelligence Artificielle)*, 2005.

[LEBRUN05B] G. LEBRUN, C. CHARRIER, O. LEZORAY, C. MEURIE & H. CARDOT. « Fast Pixel Classification by SVM Using Vector Quantization, Tabu Search and Hybrid Color Space. ». Dans *CAIP*, pages 685–692, 2005.

[LEBRUN05C] G. LEBRUN, C. CHARRIER, O. LEZORAY, C. MEURIE & H. CARDOT. « Classification rapide de pixels par SVM dans un espace couleur hybride ». Dans *CORESA (COmpression et REprésentation des signaux Audiovisuels)*, pages 231–236, 2005.

[LEBRUN06A] G. LEBRUN, O. LEZORAY, C. CHARRIER & H. CARDOT. « A new model selection method for SVM ». Dans *IDEAL*, pages 99–107, 2006.

[LEBRUN06B] G. LEBRUN, O. LEZORAY, C. CHARRIER & H. CARDOT. « Speed-up LOO-CV with SVM classifier ». Dans *IDEAL*, pages 108–115, 2006.

[LEE00] J. LEE & C. LIN. « Automatic Model Selection for Support Vector Machines. Technical Report ». http://www.csie.ntu.edu.tw /˜cjlin/papers/modelselect.ps.gz, 2000.

[LEE01] Y.-J. LEE & O. MANGASARIAN. « RSVM: Reduced support vector machines ». Dans *SIAM International Conference on Data Mining*, 2001.

[LEE04] M. LEE, S. KEERTHI, C. ONG & D. DECOSTE. « An efficient method for computing leave-one-out error in support vector machines with Gaussian kernels ». *JAIR*, 15(3), pages 750–757, 2004.

[LEI05] H. LEI & V. GOVINDARAJU. « Half-Against-Half Multi-class Support Vector Machines. ». Dans *Multiple Classifier Systems*, pages 156–164, 2005.

[LEZORA00] O. LEZORAY. « Segmentation d'images par morphologie mathématique et classification de données par réseaux de neurones : Application à la classification de cellules en cytologie des séreuses ». Thèse de doctorat, Université de Caen, Janvier 2000.

[LEZORA02] O. LEZORAY & H. CARDOT. « Cooperation of color pixel classification schemes and color watershed: a study for microscopic images. ». *IEEE Transactions on Image Processing*, 11(7), pages 783–789, 2002.

[LEZORA03] O. LEZORAY, A. ELMOATAZ & H. CARDOT. « A Color object recognition scheme: application to cellular sorting ». *Machine Vision and Applications*, vol. 14, pages 166–171, 2003.

[LEZORA04] O. LEZORAY, D. FOURNIER & H. CARDOT. « Neural network induction graph for pattern recognition. ». *Neurocomputing*, vol. 57, pages 257–274, 2004.

[LEZORA05] O. LEZORAY & H. CARDOT. « Combining Multiple Pairwise Neural Networks Classifiers: A Comparative Study. ». Dans *ANNIIP*, pages 52–61, 2005.

[LIN03A] H. LIN, C. LIN & R. WENG. « A note on Platt's probabilistic outputs for support vector machines ». Rapport technique, Department of Computer Science and Information Engineering, 2003.

[LIN03B] H.-T. LIN & C.-J. LIN. « A Study on Sigmoid Kernels for SVM and the Training of non-PSD Kernels by SMO-type Methods ». Rapport technique, Department of Computer Science, National Taiwan University, 2003.

[LIN03C] K. LIN & C. LIN. « A study on reduced support vector machines ». *Networks, IEEE Transactions*, 14(6), pages 1449–1459, 2003.

[LINDE80] Y. LINDE, A. BUZO & R. M. GRAY. « An algorithm for vector quantizer design ». Dans *IEEE Trans. Comm.*, vol. COM-28, pages 84–95, 1980.

[LITTLE86] N. LITTLESTONE & M. WARMUTH. « Relating data compression and learnability ». Rapport technique, University of California, 1986.

[LIU98] H. LIU. « Feature Selection for Knowledge Discovery and Data Mining ». Springer, 1998.

[LOOSLI05] G. LOOSLI, S. CANU, S. VISHWANATHAN, A. J. SMOLA & M. CHATTOPADHYAY. « Boite à outils SVM simple et rapide ». *Revue d'Intelligence Artificielle*, vol. 19, pages 741–767, 2005.

[LOOSLI06A] G. LOOSLI. « Méthodes à noyaux pour la détection de contexte ». Thèse de doctorat, Institut National des Sciences Appliquées de Rouen, Novembre 2006.

[LOOSLI06B] G. LOOSLI, S. CANU & L. BOTTOU. « SVM et apprentissage des très grandes bases de données ». *CAp , Conférence d'apprentissage*, 2006.

[MACLEO87] J. E. S. MACLEOD, A. LUK & D. M. TITTERINGTON. « A re-examination of the distance-weighted k-nearest neighbor classification rule ». *IEEE Transactions on Systems, Man, and Cybernetics*, 17(4), pages 689–696, 1987.

[MACQUE67] J. MACQUEEN. « Some methods for classification and analysis of Multivariable Observations ». Dans *5th Berkeley Symposium on mathematical statistics and probability*, pages 281–297, 1967.

[MARTIN01] D. MARTIN, C. FOWLKES, D. TAL & J. MALIK. « A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics ». Dans *Proc. 8th Int'l Conf. Computer Vision*, vol. 2, pages 416–423, July 2001.

[MARTIN04] D. R. MARTIN, C. FOWLKES & J. MALIK. « Learning to Detect Natural Image Boundaries Using Local Brightness, Color, and Texture Cues. ». *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5), pages 530–549, 2004.

[MAYORA99] E. MAYORAZ & E. ALPAYDIN. « Support Vector Machines for Multiclass Classification. ». Dans *IWANN (2)*, pages 833–842, 1999.

[MEILĂ05] M. MEILĂ. « Comparing Clustering - An Axiomatic View ». *ICML*, 2005.

[MERCER09] J. MERCER. « Functions of positive and negative type and their connection with the theory of integral equations ». *Philos. Trans. Roy. Soc. London*, vol. 209, pages 415–446, 1909.

[MEURIE03A] C. MEURIE, G. LEBRUN, O. LEZORAY & A. ELMOATAZ. « A comparison of supervised pixels-based color image segmentation methods. Application in cancerology ». *WSEAS Transactions on Computers*, 2(3), pages 739–744, 2003.

[MEURIE03B] C. MEURIE, G. LEBRUN, O. LEZORAY & A. ELMOATAZ. « A supervised segmentation scheme for cancerology color images ». Dans *ISSPIT (IEEE International Symposium on Signal Processing and Information Technology)*, pages 664–667, 2003.

[MEURIE05A] C. MEURIE. « Segmentation d'images couleur par classification pixellaire et hiérarchie de partitions. ». Thèse de doctorat, Université de Caen, Octobre 2005.

[MEURIE05B] C. MEURIE, O. LEZORAY, C. CHARRIER & A. ELMOATAZ. « Combination of multiple pixel classifiers for microscopic image segmentation ». *IJRA (IASTED International Journal of Robotics and Automation)*, 20(2), pages 63–69, 2005. Special issue on Colour Image Processing and Analysis for Machine Vision, ISSN 0826-8185.

[MITCHE80] T. MITCHELL. « The need for biais in learning generalizations ». Rapport technique, CBM-TR 5-110, Rutgers University, 1980.

[MITCHE97] T. MITCHELL. « Machine Learning ». The McGraw-Hill Companies, 1997.

[MOREIR98] M. MOREIRA & E. MAYORAZ. « Improved Pairwise Coupling Classification with Correcting Classifiers. ». Dans *ECML*, pages 160–171, 1998.

[MORIN81] R. L. MORIN. « A reappraisal of distance-weighted k- nearest neighbor classification for pattern recognition with missing data ». *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 11, pages 241–243, 1981.

[NGUYEN05] D. NGUYEN & T. B. HO. « An efficient method for simplifying support vector machines. ». Dans *ICML*, pages 617–624, 2005.

[ODORIC97] R. ODORICO. « Learning Vector Quantization with Training Count (LVQTC). ». *Neural Networks*, 10(6), pages 1083–1088, 1997.

[OSUNA97] E. OSUNA, R. FREUND & F. GIROSI. « An improved training algorithm for support vector machines ». Dans *IEEE Workshop on NNSP*, pages 276–285, 1997.

[OU03] Y. Y. OU, C. Y. CHEN, S. C. HWANG & Y. J. OYANG. « Expediting Model Selection for Support Vector Machines Based on Data Reduction ». Dans *IEEE Proc. SMC*, pages 786–791, 2003.

[OU04] G. OU, Y. L. MURPHEY & L. A. FELDKAMP. « Multiclass Pattern Classification Using Neural Networks. ». Dans *ICPR (4)*, pages 585–588, 2004.

[PAPANI54] G. N. PAPANICOLAOU. « Atlas of Exfoliative Cytology ». Harvard University Press, 1954.

[PARZEN62] E. PARZEN. « On estimation of a probability density function and mode ». *Annals Mathematical Statistics*, vol. 33, pages 1065–1076, 1962.

[PEARL84] J. PEARL. « Heuristics: Intelligent Search Strategies for Computer Problem Solving ». Addison-Wesley, 1984.

[PELI90] E. PELI. « Contrast in complex images ». *Journal of the Optical Society of America*, 7(10), pages 2032–2040, octobre 1990.

[PÉREZ-02] F. PÉREZ-CRUZ & A. ARTÉS-RODRÍGUEZ. « Puncturing Multi-class Support Vector Machines. ». Dans *ICANN*, pages 751–756, 2002.

[PHETKA03] T. PHETKAEW, W. RIVEPIBOON & B. KIJSIRIKUL. « Reordering Adaptive Directed Acyclic Graphs for Multiclass Support Vector Machines. ». *JACIII*, 7(3), pages 315–321, 2003.

[PLATON00] K. N. PLATONIOTIS & A. N. VENETSANOPOULOS. « Color Image Processing and Applications ». Springer, 2000.

[PLATT99A] J. PLATT. « Fast Training of Support Vector Machines using Sequential Minimal Optimization, Advances in Kernel Methods-Support Vector Learning ». Dans *MIT Press*, pages 185–208, 1999.

[PLATT99B] J. PLATT. « Probabilistic outputs for support vector machines and comparison to regularized likelihood methods ». Dans A. J. SMOLA, P. BARTLETT, B. SCHOELKOPF & D. SCHUURMANS, rédacteurs, *Advances in Large Margin Classifiers*, pages 61–74, 1999.

[PLATT99C] J. C. PLATT, N. CRISTIANINI & J. SHAWE-TAYLOR. « Large Margin DAGs for Multiclass Classification. ». Dans *NIPS*, pages 547–553, 1999.

[PRICE94] D. PRICE, S. KNERR, L. PERSONNAZ & G. DREYFUS. « Pairwise Neural Network Classifiers with Probabilistic Outputs. ». Dans *NIPS*, pages 1109–1116, 1994.

[PUANGD04] D. PUANGDOWNREONG, S. SUJITJORN & T. KULWORAWANICH. « Convergence Analysis of Adaptive Tabu Search ». *ScienceAsia*, 30(2), pages 183–190, 2004.

[PUDIL94] P. PUDIL, J. NOVOVICOVÁ & J. KITTLER. « Floating search methods in feature selection. ». *Pattern Recognition Letters*, 15(10), pages 1119–1125, 1994.

[QUINLA86] J. R. QUINLAN. « Induction of decision trees. ». *Machine Learning*, 1(1), pages 81–106, 1986.

[QUINLA93] J. R. QUINLAN. « C4.5: Programs for Machine Learning ». Morgan Kaufmann, 1993.

[QUOST04] B. QUOST, T. DENOEUX & M. MASSON. « Combinaison de classifieurs binaires dans le cadre du Modèle des Croyances Transférables ». Dans *Rencontres Francophones sur la Logique Floue et ses Applications*, pages 123–130, 2004.

[QUOST05] B. QUOST, T. DENOEUX & M. MASSON. « Pairwise Classifier Combination in the Framework of Belief Functions ». Dans *Fusion*, 2005.

[QUOST06] B. QUOST, T. DENOEUX & M. MASSON. « One-against-all classifier combination in the framework of belief functions ». Dans *IPMU*, vol. 1, pages 356–363, 2006.

[RAJAN04] S. RAJAN & J. GHOSH. « An Empirical Comparison of Hierarchical vs. Two-Level Approaches to Multiclass Problems. ». Dans *Multiple Classifier Systems*, pages 283–292, 2004.

[RAKOTO05] A. RAKOTOMAMONJY & S. CANU. « Frames, Reproducing Kernels, Regularization and Learning ». *Journal of Machine Learning Research*, vol. 6, pages 1485–1515, 2005.

[RALAIV01] L. RALAIVOLA & F. D'ALCHÉ BUC. « Incremental Support Vector Machine Learning: A Local Approach. ». Dans *ICANN*, pages 322–330, 2001.

[RECHEN65] I. RECHENBERG. « Cybernetic Solution Path of an Experimental Problem ». Royal Aircraft Establishment Library Translation, 1965.

[RIFKIN04] R. M. RIFKIN & A. KLAUTAU. « In Defense of One-Vs-All Classification. ». *Journal of Machine Learning Research*, vol. 5, pages 101–141, 2004.

[RISSAN78] J. RISSANEN. « Modeling by the shortest data description ». *Automatica*, vol. 14, pages 465–471, 1978.

[RISSAN99] J. RISSANEN. « Hypothesis selection and testing by the MDL principle. ». *The Computer Journal*, 42(4), 1999.

[ROBERT88] T. ROBERTSON, F. WRIGHT & R. DYKSTRA. « Order Restricted Statistical Inference ». John Wiley and Sons, 1988.

[ROSENB58] F. ROSENBLATT. « The perceptron: a probabilistic model for information storage and organization in the brain. ». *Psychological Review*, vol. 65, pages 386–408, 1958.

[RÖ04] C. RÖVER & G. SZEPANNEK. « Application of a Genetic Algorithm to Variable Selection in Fuzzy Clustering ». Rapport technique, Technical Report 76/2004, SFB 475, Department of Statistics, 2004.

[RÜ04] S. RÜPING. « A Simple Method For Estimating Conditional Probabilities For SVMs. ». Dans *LWA*, pages 206–210, 2004.

[SAVICK03] P. SAVICKÝ & J. FÜRNKRANZ. « Combining Pairwise Classifiers with Stacking. ». Dans *IDA*, pages 219–229, 2003.

[SCHAPI97] R. E. SCHAPIRE, Y. FREUND, P. BARLETT & W. S. LEE. « Boosting the margin: A new explanation for the effectiveness of voting methods. ». Dans *ICML*, pages 322–330, 1997.

[SCHOHN00] G. SCHOHN & D. COHN. « Less is More: Active Learning with Support Vector Machines ». Dans *ICML*, pages 839–846, 2000.

[SCHÖL99] B. SCHÖLKOPF, R. C. WILLIAMSON, A. J. SMOLA, J. SHAWE-TAYLOR & J. C. PLATT. « Support Vector Method for Novelty Detection. ». Dans *NIPS*, pages 582–588, 1999.

[SCHUPP00] S. SCHUPP, A. ELMOATAZ, M.-J. FADILI, P. HERLIN & D. BLOYET. « Image Segmentation via Multiple Active Contour Models and Fuzzy Clustering with Biomedical Applications. ». Dans *ICPR*, pages 1622–1625, 2000.

[SCOTT02] C. SCOTT & R. NOWAK. « Dyadic Classification Trees via Structural Risk Minimization. ». Dans *NIPS*, pages 359–366, 2002.

[SHIN02] H. SHIN & S. CHO. « Pattern Selection for Support Vector Classifiers. ». Dans *IDEAL*, pages 469–474, 2002.

[SHIN03] H. SHIN & S. CHO. « Fast Pattern Selection Algorithm for Support Vector Classifiers: Time Complexity Analysis. ». Dans *IDEAL*, pages 1008–1015, 2003.

[SUN02] Z. SUN, X. YUAN, G. BEBIS & S. LOUIS. « Genetic feature subset selection for gender classication: A comparison study ». Dans *IEEE Workshop on Applications of Computer Vision*, 2002.

[TAHAHA03] F. TAHAHASHI & S. ABE. « Optimizing directed acyclic graph support vector machines ». Dans *Proc. Artificial Neural Networks*, pages 166–170, 2003.

[TAILLA95] . D. TAILLARD. « Comparison of Iterative Searches for the Quadratic Assignment Problem ». *Location Science 3*, 3(2), pages 87–105, 1995.

[TAILLA98] E. TAILLARD. « Programmation à mémoire adaptative et algorithmes pseudo-gloutons : nouvelles perspectives pour les métaheuristiques ». Thèse de doctorat, Thèse d'habilitation à diriger les à diriger les recherches, Université de Versailles Saint Quentin en Yvelines, France, 1998.

[TAN06] P. N. TAN, M. STEINBACH & V. KUMAR. « Introduction to Data Mining ». Addison-Wesley, 2006.

[TAX99] D. M. J. TAX & R. P. W. DUIN. « Support vector domain description ». *Pattern Recognition Letters*, 20(11-13), 1999.

[TIBSHI96] R. TIBSHIRANI. « Bias, variance and prediction error for classification rules ». Rapport technique, Department of Statistics, University of Toronto, 1996.

[TRÉM97] A. TRÉMEAU, C. CHARRIER & E. FAVIER. « Quantitative description of image distorsions linked to compression schemes ». Dans *Proceedings of The Int. Conf. on the Quantitative Description of Materials Microstructure*, Warsaw, avril 1997. QMAT'97.

[TREMEA03] A. TREMEAU. « L'IMAGE NUMERIQUE COULEUR ; DE L'ACQUISITION AU TRAITEMENT ». Dunod, 2003.

[TRUNK79] G. V. TRUNK. « A problem of dimensionality: A simple example. ». *IEEE Trans. Pattern Anal. Mach. Intell.*, 1(3), pages 153–158, 1979.

[TSAMAR03] I. TSAMARDINOS & C. F. ALIFERIS. « Towards Principled Feature Selection: Relevancy, Filters, and Wrappers ». Dans *Artificial Intelligence and Statistics*, 2003.

[TSANG05] I. W. TSANG, J. T. KWOK & P.-M. CHEUNG. « Core Vector Machines: Fast SVM Training on Very Large Data Sets ». *JMLR*, vol. 6, pages 363–392, 2005.

[UIT-RR00] UIT-R RECOMMENDATION BT.500-10. « Méthodologie d'évaluation subjective de la qualité des images de télévision ». Rapport technique, UIT, Geneva, Switzerland, 2000.

[VANDEN98] N. VANDENBROUCKE, L. MACAIRE & J.-G. POSTAIRE. « Color Pixels Classification in an Hybrid Color Space. ». Dans *ICIP (1)*, pages 176–180, 1998.

[VANDEN00A] N. VANDENBROUCKE. « Segmentation d'images couleur par classification de pixels dans des espaces d'attributs colorimétriques adaptés. Application à l'analyse d'images de football. ». Thèse de doctorat, Université des Sciences et Technologies de Lille 1, Janvier 2000.

[VANDEN00B] N. VANDENBROUCKE, L. MACAIRE & J.-G. POSTAIRE. « Color Image Segmentation by Supervised Pixel Classification in a Color Texture Feature Space: Application to Soccer Image Segmentation. ». Dans *ICPR*, pages 3625–3628, 2000.

[VANDEN03] N. VANDENBROUCKE, L. MACAIRE & J.-G. POSTAIRE. « Color image segmentation by pixel classification in an adapted hybrid color space. Application to soccer image analysis. ». *Computer Vision and Image Understanding*, 90(2), pages 190–216, 2003.

[VANDER99] R. J. VANDERBEI. « LOQO: An interior point code for quadratic programming ». *Optimization Methods and Software*, vol. 11, pages 451–484, 1999.

[VAPNIK95] V. N. VAPNIK. « The nature of statistical learning theory ». Springer-Verlag, 1995.

[VAPNIK98] V. N. VAPNIK. « Statistical Learning Theory ». New York, wiley édition, 1998.

[VEROPO99] K. VEROPOULOS, N. CRISTIANINI & C. CAMPBELL. « Controlling the Sensitivity of Support Vector Machines ». Dans *International Joint Conference on Artificial Intelligence*, pages 55–60, 1999.

[VINCEN01] P. VINCENT & Y. BENGIO. « K-Local Hyperplane and Convex Distance Nearest Neighbor Algorithms. ». Dans *NIPS*, pages 985–992, 2001.

[VINCEN03] P. VINCENT. « Modèles à noyaux à structure locale ». Thèse de doctorat, Univeristé de Montréal, 2003.

[VURAL04] V. VURAL & J. G. DY. « A hierarchical method for multi-class support vector machines. ». Dans *ICML*, pages 831–838, 2004.

[WANG97] W. WANG, J. YANG & R. R. MUNTZ. « STING: A Statistical Information Grid Approach to Spatial Data Mining ». Dans *Twenty-Third International Conference on Very Large Data Bases*, pages 186–195, Morgan Kaufmann, 1997.

[WANG02] Z. WANG & A. C. BOVIK. « A Universal Quality Index ». *IEEE Signal Processing Letters*, 9(3), pages 81–84, 2002.

[WANG04] H. WANG & D. BELL. « Extended k-Nearest Neighbours based on Evidence Theory ». *Computer Journal*, 47(6), pages 662–672, 2004.

[WANG05] Z. WANG, A. C. BOVIK & E. P. SIMONCELLI. « Structural Approaches to Image Quality Assessment ». Dans *Handbook of Image and Video Processing*. Academic Press, 2nd édition, 2005.

[WATSON87] A. B. WATSON. « The Cortex transform: Rapid computation of simulated neural images ». *Computer Vis. Graphics and Image Proces.*, vol. 39, pages 311–327, 1987.

[WILSON72] D. L. WILSON. « Asymtotic properties of nearest neighbor rules using edited data ». *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 2, pages 408–421, 1972.

[WOLPER95] D. H. WOLPERT & W. G. MACREADY. « No Free Lunch Theorems for Search ». Rapport technique, Technical Report SFI-TR-95-02-010, 1995.

[WOODRU93] D. L. WOODRUFF & E. ZEMEL. « Hashing vectors for tabu search ». *Ann. Oper. Res.*, 41(1-4), pages 123–137, 1993. ISSN 0254-5330.

[WOODRU94] D. L. WOODRUFF. « Simulated annealing and tabu search: Lessons from a line search. ». *Computers & OR*, 21(8), pages 823–839, 1994.

[WU04] T.-F. WU, C.-J. LIN & R. C. WENG. « Probability Estimates for Multi-class Classification by Pairwise Coupling. ». *Journal of Machine Learning Research*, vol. 5, pages 975–1005, 2004.

[XIE93] Q. XIE, C. A. LASZLO & R. K. WARD. « Vector Quantization Technique for Nonparametric Classifier Design. ». *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(12), pages 1326–1330, 1993.

[XU05] R. XU & D. I. I. WUNSCH. « Survey of Clustering Algorithms ». *IEEE Tansactions on neural networks*, 16(3), pages 645–678, 2005.

[XUCHUN05] E. S. XUCHUN LI. « A Study of AdaBoost with SVM Weak Learners ». Dans *IJCNN*, pages 196–201, 2005.

[YANG00] M.-H. YANG & N. AHUJA. « A Geometric Approach to Train Support Vector Machines. ». Dans *CVPR*, pages 1430–1437, 2000.

[YU03A] H. YU, J. YANG & J. HAN. « Classifying Large Data Sets Using SVM with Hierarchical Clusters ». Dans *SIGKDD*, pages 306–315, 2003.

[YU03B] H. YU, J. YANG & J. HAN. « Classifying large data sets using SVMs with hierarchical clusters. ». Dans *KDD*, pages 306–315, 2003.

[ZADROZ02] B. ZADROZNY & C. ELKAN. « Transforming classifier scores into accurate multiclass probability estimates. ». Dans *KDD*, pages 694–699, 2002.

[ZHANG96] T. ZHANG, R. RAMAKRISHNAN & M. LIVNY. « BIRCH: an efficient data clustering method for very large databases ». Dans *the International Conference Management of Data (ACM-SIGMOD)*, pages 103–114, 1996.

[ZHANG02] W. ZHANG & I. KING. « Locating support vectors via beta-skeleton technique ». *Neural Information Processing*, vol. 3, pages 1423–1427, 2002.