ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

THÈSE PRÉSENTÉE À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DU
DOCTORAT EN GÉNIE
Ph.D.

PAR
JONATHAN MILGRAM

CONTRIBUTION À L'INTÉGRATION DES MACHINES À VECTEURS DE SUPPORT AU SEIN DES SYSTÈMES DE RECONNAISSANCE DE FORMES: APPLICATION À LA LECTURE AUTOMATIQUE DE L'ÉCRITURE MANUSCRITE

MONTRÉAL, LE 29 JUIN 2007

CETTE THÈSE A ÉTÉ ÉVALUÉE
PAR UN JURY COMPOSÉ DE :

Mme Nadia Ghazzali, examinateur externe
Département de mathématiques et de statistique à l'Université Laval

M. Maarouf Saad, président du jury
Département de génie électrique à l'École de technologie supérieure

M. Éric Granger, membre du jury
Département de génie de la production automatisée à l'École de technologie supérieure

M. Robert Sabourin, directeur de thèse
Département de génie de la production automatisée à l'École de technologie supérieure

M. Mohamed Cheriet, codirecteur de thèse
Département de génie de la production automatisée à l'École de technologie supérieure

ELLE A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC
LE 31 MAI 2007
À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

CONTRIBUTION À L'INTÉGRATION DES MACHINES À VECTEURS DE SUPPORT AU SEIN DES SYSTÈMES DE RECONNAISSANCE DE FORMES: APPLICATION À LA LECTURE AUTOMATIQUE DE L'ÉCRITURE MANUSCRITE

Jonathan Milgram

SOMMAIRE

Durant ces dernières années, les machines à vecteurs de support (SVM) ont démontré à maintes reprises leur supériorité en termes de généralisation. L'objectif de cette thèse de doctorat a alors consisté à isoler les principaux problèmes liés à l'intégration des SVM au sein de systèmes de reconnaissance de formes et à y apporter des éléments de réponse. Tout d'abord, un SVM ne permettant de séparer que deux classes, il est nécessaire d'en combiner plusieurs pour résoudre des problèmes multi-classes. Nous avons alors comparé deux stratégies classiques à l'aide de deux problèmes de reconnaissance de caractères manuscrits. D'autre part, étant donné qu'il est fréquent que le classifieur ne contribue qu'en partie à la décision finale du système, nous avons comparé différentes méthodes permettant d'estimer des probabilités *a posteriori* à partir des sorties des SVM. Il en est ressorti que les SVM permettent d'estimer des probabilités significativement plus précises que celles estimées par un perceptron multi-couches (MLP). Cependant, ce gain en précision se fait aux dépens de la complexité. Nous avons alors cherché à combiner MLP et SVM de manière à accélérer la prise de décision. Les schémas de combinaison qui ont été proposés ont finalement permis de réduire considérablement la complexité sans perdre en précision. Enfin, de manière à traiter efficacement à la fois les données ambiguës et les données aberrantes, nous avons proposé de combiner au sein d'un système à deux niveaux de décision, des SVM qui sont des approches agissant par séparation avec une approche agissant par modélisation. Nous avons alors montré que cette combinaison permet de dissocier le rejet d'ambiguïté et le rejet d'ignorance et d'exploiter au mieux les capacités des deux types d'approche de classification. De plus, la grande modularité de ce type d'approche permetra de résoudre efficacement des problèmes comprenant un très grand nombre de classes.

CONTRIBUTION TO THE INTEGRATION OF SUPPORT VECTORS MACHINES INTO PATTERN RECOGNITION SYSTEMS: APPLICATION TO AUTOMATIC READING OF CURSIVE HANDWRITING

Jonathan Milgram

ABSTRACT

During the last years, support vector machines (SVMs) showed their superiority in terms of generalisation capability. Therefore, the objective of this thesis is to allow for the integration of SVMs in pattern recognition systems. Indeed, to use SVMs into complex systems, like automatic reading systems of cursive handwriting, it is necessary to deal with some particularities of these classifiers. Firstly, since a SVM is able to separate only two classes, it is necessary to combine several SVMs to resolve multi-class problems. Then, we compared the two most popular strategies on two handwritten character recognition problems. Moreover, since in a complex system, the classifier only contributes to a small part of the final decision, it is important to be able to estimate posterior probabilities. Then, we compared several methods to estimate this type of measure, and the results show that SVMs allow significantly better estimation of probabilities than MLP, but are also more complex. For this reason, we proposed to combine MLP and SVM to speed up the decision making. The proposed approach makes it possible to reduce considerably the complexity, without decreasing the precision. Finally, we propose to combine SVMs with a model-based approach. Indeed, this type of approach is better than discriminative approach to detect outliers, but less accurate to classify ambiguous data. Furthermore, the modularity of the proposed approach seems interesting to resolve classification problems, like Chinese or Japanese character recognition, for which the number of classes is very large.

REMERCIMENTS

En premier lieu, je tiens à remercier Robert Sabourin et Mohamed Cheriet pour m'avoir fait confiance tout au long de ce doctorat, pour m'avoir encadré dans mes travaux de recherche et avoir su faire preuve de patiente.

D'autre part, je remercie Nadia Ghazzali, Maarouf Saad et Eric Granger pour avoir pris le soin d'évaluer ce travail.

Mes remerciements vont aussi à Maurice Milgram et Lionel Prevost qui m'ont fait découvrir ce domaine de recherche et m'ont donné le goût de continuer.

Un grand merci va également à l'ensemble des membres passés et présents du LIVIA avec qui j'ai partagé bien plus qu'un lieu de travail. Parmi eux, je tiens spécialement à remercier Fred, Didine, Luiz, Marisa, Alessandro, Jean-Philippe, Youssef, Antoine, Luis, Paulo, Ali, Seb, Guillaume, Clem, Cyril, Cabelão, Vincent, Henri, Thomas, Mathias, Albert, Marcelo et moça Yoyo.

Par ailleurs, un merci tout particulier va à mon "colloc" Omar pour son amitié, son soutien et le plaisir que ce fut de vivre avec lui durant ces quelques années.

Enfin, je tiens à remercier du fond du cœur mes parents, ainsi que le reste de ma famille pour leur amour sans lequel je ne pourrais vivre ...

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS ET NOTATIONS

FLOP	FLoating-point OPeration
HMM	Hidden Markov Model
KFLOP	Kilo FLOP (10^3 FLOP)
k-NN	k-Nearest Neighbor
LQDF	Learning Quadratic Discriminant Function
MFLOP	Mega FLOP (10^6 FLOP)
MLP	Multi-Layer Perceptron
MQDF	Modified Quadratic Discriminant Function
NLL	Negative Log-Likelihood
OAA	One Against All
OAO	One Against One
RBF	Radial Basis Function
SSC	Sub-Space Classifier
SSE	Sum Square Error
SVM	Support Vector Machine
c	nombre de classes
d	nombre de caractéristiques
n	nombre d'exemples
x	donnée à classer
ω_i	$i^{ème}$ classe
$P(\omega_i \mid x)$	probabilité *a posteriori* que la donnée x appartienne à la classe ω_i
$P(\omega_i)$	probabilité *a priori* qu'une donnée appartienne à la classe ω_i
$p(x \mid \omega_i)$	densité de probabilités associée à la classe ω_i
$f_i(x)$	sortie du SVM entraîné à séparer la classe ω_i des autres classes
$f_{i,j}(x)$	sortie du SVM entraîné à séparer la classe ω_i de la classe ω_j

INTRODUCTION

Dans les années 90, un nouveau type d'algorithme d'apprentissage a été développé à partir des résultats de la théorie de l'apprentissage statistique : les machines à vecteurs de support, plus connues sous l'acronyme anglais SVM (Support Vector Machine). Depuis, les SVM ont prouvé à maintes reprises leur efficacité pour résoudre des problèmes de classification dans des espaces de grandes dimensions. Récemment encore, dans le cadre d'une étude comparative portant sur la reconnaissance d'images de chiffres manuscrits, Liu *et al.* [1] ont montré que les SVM s'avèrent plus discriminants que des classifieurs classiques, tel qu'un *k*-NN (*k*-Nearest Neighbor), un réseau RBF (Radial Basis Function) ou un MLP (Multi-Layer Perceptron). Quelques-uns de leurs résultats sont reportés au tableau I. Nous pouvons y constater que pour les trois bases de données utilisées, c'est effectivement avec les SVM que le taux d'erreur en généralisation est le plus faible.

Tableau I

Pourcentage d'erreur en généralisation (résultats extraits de [1])

	k-NN	RBF	MLP	**SVM**
CENPARMI Database	2,35	1,55	1,70	**0,95**
CEDAR Database	0,99	0,99	1,04	**0,68**
MNIST Database	0,97	0,69	0,60	**0,42**

À la vue de ces résultats, il peut donc sembler particulièrement tentant de vouloir utiliser ce type de classifieur au sein de systèmes de reconnaissance de formes et plus particulièrement au sein de systèmes de lecture automatique de l'écriture manuscrite.

Problématique de recherche

Tout d'abord, notons que ce projet de recherche fait suite à six thèses de doctorat [2-7], soutenues durant les cinq dernières années au sein du LIVIA (Laboratoire d'Imagerie, de Vision et d'Intelligence Artificielle). L'ensemble de ces travaux ont en commun une même application : la lecture automatique de l'écriture manuscrite. Nous avons donc choisi de profiter de cette importante expertise pour valider les méthodes qui seront proposées dans le cadre de ce doctorat. Cependant, notre objectif sera de proposer des méthodes qui ne sont pas spécifiques à cette application. Bien entendu, nous ne prétendons pas proposer des méthodes adaptées à l'ensemble des problèmes de reconnaissance de formes, mais à une famille d'applications comparables.

Prenons donc comme exemple le système de reconnaissance de noms de ville proposé par Koerich *et al.* [8]. Comme nous pouvons le constater à la figure 1, ce système est composé d'un module de reconnaissance qui utilise des HMM (Hidden Markov Models) pour attribuer un score à chacun des noms de ville faisant parti du lexique utilisé et d'un module de vérification qui utilise un SNN (Segmental Neural Network) pour raffiner les scores des hypothèses les plus probables. Ce SNN est en fait un MLP entraîné à distinguer des images de lettres isolées. Il pourrait donc être intéressant de remplacer ce classifieur par des SVM. Cependant, un certain nombre de questions se posent alors. L'objectif de cette thèse de doctorat consiste ainsi à isoler les principaux problèmes liés à l'intégration des SVM dans de tels systèmes et à y apporter des éléments de réponse.

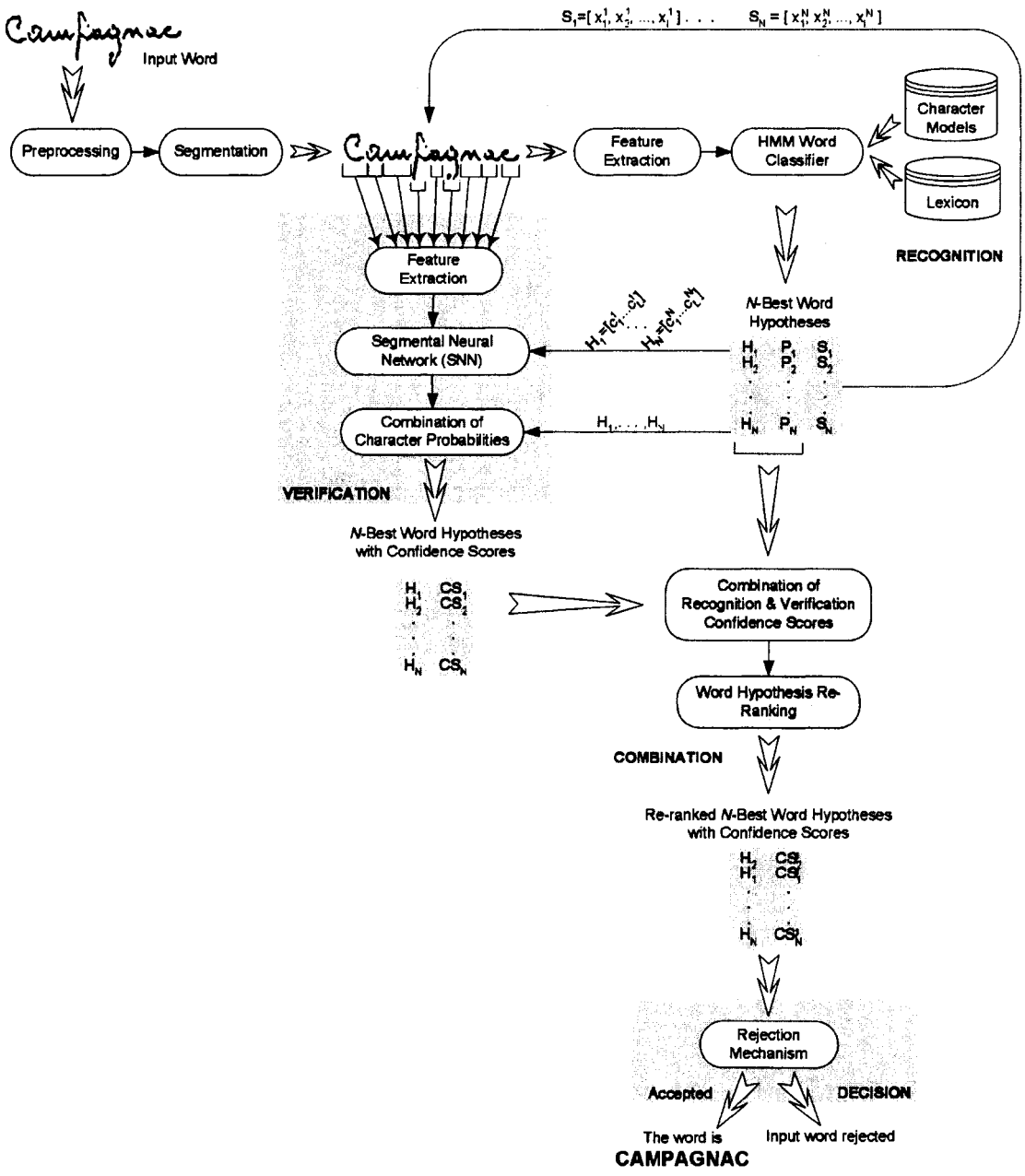


Figure 1 Vue d'ensemble du système proposé par Koerich *et al.* (extrait de [8])

Objectifs et contributions

Tout d'abord, dans sa formulation originale, un SVM ne permet de séparer que deux classes. Or, si plusieurs stratégies sont envisageables pour décomposer un problème multi-classes en un ensemble de sous-problèmes binaires, la première question est donc de savoir laquelle de ces stratégies est-il préférable d'utiliser ? Nous avons alors réalisé une étude comparative des deux stratégies les plus courantes, à savoir la stratégie « un contre un » et la stratégie « un contre tous ».

Ensuite, dans le cas de systèmes où le classifieur ne contribue qu'en partie à la décision finale du système, il est essentiel de pouvoir estimer des probabilités *a posteriori* d'appartenance aux différentes classes. La seconde question à laquelle nous nous sommes intéressés a été de savoir comment estimer de telles probabilités à l'aide de SVM. Nous avons alors proposé une nouvelle méthode adaptée à la stratégie « un contre tous » que nous avons comparé expérimentalement à l'approche classique et à trois méthodes adaptées à la stratégie « un contre un ».

D'autre part, si l'utilisation de SVM peut permettre de gagner en précision, ceci se fait malheureusement au détriment de la complexité. En effet, dans leur étude comparative, Liu *et al.* [1] montrent par exemple que sur la base MNIST, le temps de traitement nécessaire à la prise de décision est cinquante fois plus élevé dans le cas des SVM que dans le cas du MLP. Ainsi, dans le cas d'applications où le temps de traitement est important pour des raisons de rentabilité ou des contraintes de temps réel, comme dans le cas du trie automatique de documents manuscrits, cette propriété des SVM peut devenir un inconvénient majeur. La troisième question concerne donc l'accélération de la prise de décision et consiste donc à déterminer comment réduire la complexité de calcul sans perdre significativement en précision. En se basant sur une revue critique de la littérature, nous avons proposé une nouvelle approche qui consiste à combiner astucieusement MLP et SVM au sein d'un système à deux niveaux de décision.

Pour finir, les SVM, tout comme le reste des approches de classification agissant par séparation, s'avèrent généralement peu efficaces pour détecter d'éventuelles données aberrantes, c'est-à-dire des données qui n'appartiennent à aucune des classes du problème, comme par exemple les données qui résultent d'erreur lors de la segmentation d'un mot en lettres. Ainsi, la dernière question à laquelle nous nous sommes intéressés a été de savoir comment traiter efficacement ce type de données problématiques. Nous avons alors proposé un système original combinant des SVM avec une approche de classification agissant par modélisation qui s'avère bien mieux adaptée au rejet de données aberrantes.

Organisation du manuscrit

Notre problématique de recherche ne portant donc pas sur un problème unique, mais sur un ensemble de sous-problèmes, nous avons préféré répartir l'état de l'art, la méthodologie et les résultats expérimentaux dans chacun des chapitres correspondant à chacune des problématiques. Notre manuscrit est donc organisé de la façon suivante.

Le chapitre 1 présente les fondements théoriques sur lesquels reposent les machines à vecteurs de support, à savoir les notions d'hyperplan optimal et d'astuce du noyau. Un problème artificiel en deux dimentions est alors utilisé de manière à pouvoir visualiser l'effet des hyper-paramètres qui conditionnent l'apprentissage des SVM.

Le chapitre 2 est consacré à l'utilisation de SVM pour la résolution de problèmes multi-classes. Un état de l'art des différentes approches existantes y est présenté et les deux approches les plus courantes, à savoir les stratégies « un contre un » et « un contre tous », y sont comparées expérimentalement à l'aide de deux bases de données réelles, une base d'images de chiffres et une base d'images de lettres majuscules.

Le chapitre 3 porte sur l'estimation de probabilités *a posteriori* à l'aide de machines à vecteurs de support. La première partie du chapitre est consacrée au calibrage des sorties d'un SVM. Un état de l'art y est présenté et les principales méthodes y sont comparées expérimentalement à la fois à l'aide de données artificielles et des données réelles. La seconde partie du chapitre est quant à elle consacrée à l'extention aux problèmes multi-classes. Différentes méthodes adaptés aux stratégies « un contre un » et « un contre tous » y sont présentées en détails, puis comparées expérimentalement à l'aide de nos deux bases de données réelles.

Le chapitre 4 a pour thème l'accélération de la prise de décision. Un état de l'art des approches existantes y est présenté de manière à justifier l'approche que nous avons proposé qui est ensuite décrite en détails, puis validée expérimentalement. Les résultats obtenus sur nos deux bases de données montrent alors que notre approche permet bien d'accélérer fortement la prise de décision, sans perdre en précision.

Enfin, dans le chapitre 5 nous montrons qu'il peut aussi être avantageux de combiner les SVM avec une approche de classification agissant par modélisation, dont l'un des avantages est de pouvoir détecter efficacement les données aberrantes. Nous montrons alors expérimentalement que ce type de combinaison permet de traiter efficacement les deux types de données problématiques, à savoir les données ambigües et les données aberrrantes.

Bien entendu, en conclusion de ce manuscrit, nous dressons un bilan des contributions qu'apporte cette thèse de doctorat et des perspectives de travaux futurs qui en ressortent.

CHAPITRE 1

MACHINE À VECTEURS DE SUPPORT

Les machines à vecteurs de support peuvent être définies comme étant une famille d'algorithmes d'apprentissage permettant de résoudre efficacement des problèmes de classification, ou de régression, en exploitant les résultats de la théorie de l'apprentissage statistique développée en grande partie par Vladimir Vapnik [9]. Le principe sous-jacent aux SVM consiste à utiliser une transformation non linéaire pour redécrire les données d'apprentissage dans un espace de plus grande dimension, dans lequel elles pourront être traitées efficacement de manière linéaire. Dans le cas d'un problème de classification binaire, l'objectif est alors de déterminer dans le nouvel espace, que l'on nomme espace de redescription, un hyperplan qui permet de séparer les données d'apprentissage de manière optimale. Dans l'exemple illustré à la figure 2, les données initialement décrites par deux caractéristiques x_1 et x_2 sont redécrites par trois nouvelles caractéristiques $z_1 = x_1^2$, $z_2 = x_2^2$ et $z_3 = x_1 \cdot x_2$.

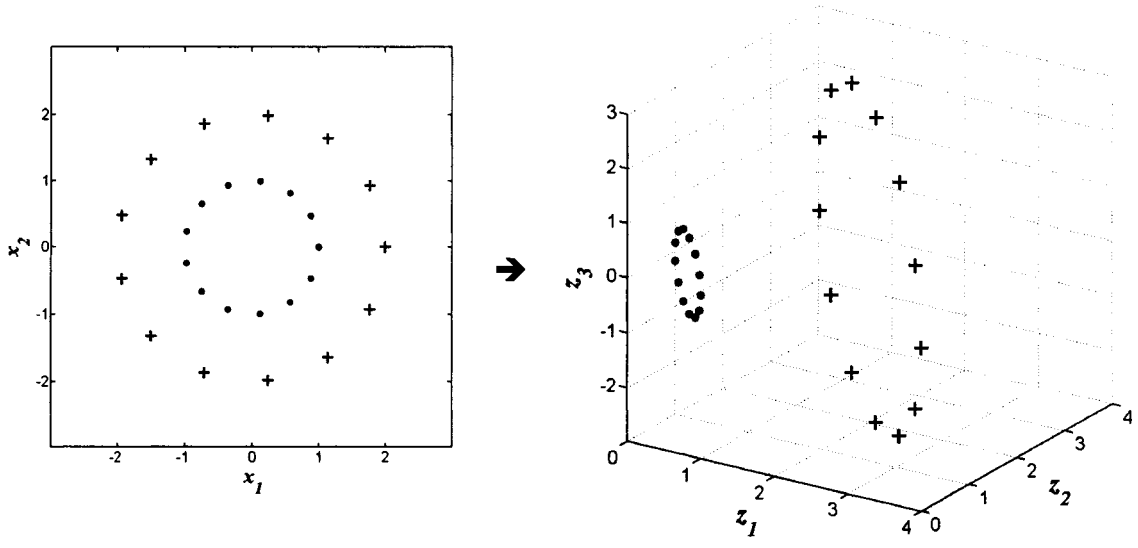


Figure 2 Redescription des données dans un espace de plus grande dimension

Ainsi, comme on peut le constater à la figure 3, les données qui n'étaient pas linéairement séparables dans l'espace de départ, le sont dans l'espace de redescription et la frontière de décision linéaire déterminée dans ce nouvel espace, correspond à une frontière de décision non-linéaire dans l'espace de départ.

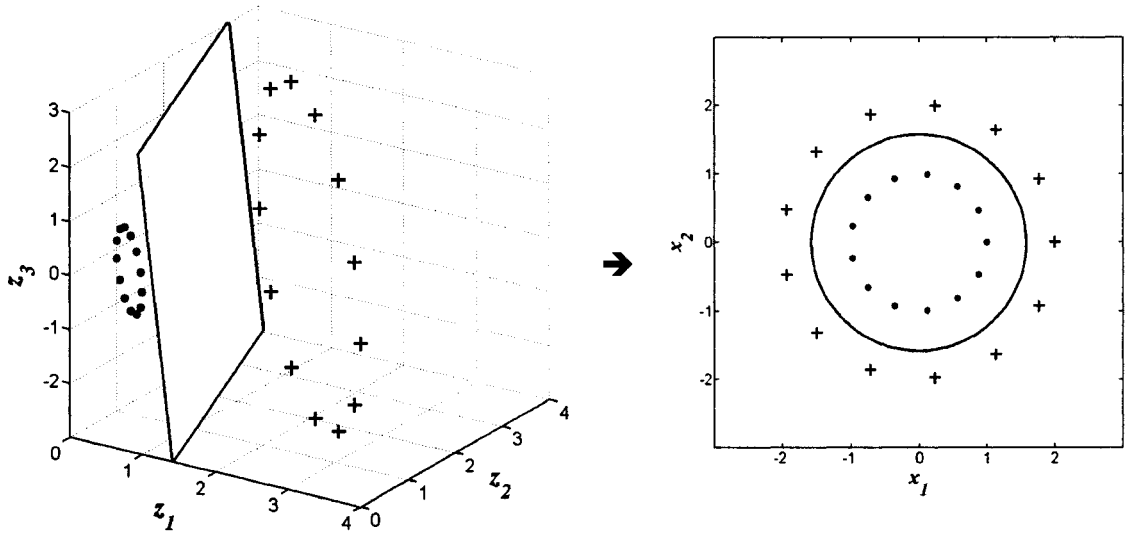


Figure 3 Discrimination linéaire dans l'espace de redescription

Cependant, deux problèmes se posent. Le premier, d'ordre conceptuel, est de choisir l'hyperplan qui permettra la meilleure généralisation. Le second, d'ordre pratique, est de déterminer cet hyperplan dans un espace de très grande dimension. En effet, pour obtenir une frontière de décision de type polynomiale de degré 5, dans un espace d'entrée caractérisé par plusieurs centaines de caractéristiques, il serait nécessaire de construire un hyperplan dans un espace de redescription qui compterait plusieurs milliards de caractéristiques ! Bien entendu, il est impossible de travailler directement dans un espace d'aussi grande dimension. La solution consiste alors à utiliser « l'astuce du noyau » pour déterminer l'hyperplan qui sépare de manière « optimale » les données dans cet espace de très grande dimension, sans avoir besoin de redécrire les données dans celui-ci.

1.1 L'hyperplan optimal

Commençons par préciser ce que l'on entend par « optimal ». En effet, la notion d'optimalité implique un critère d'évaluation. Dans le cas d'un problème de classification, ce critère pourrait donc être le nombre d'erreurs de classification parmi les données d'apprentissage. On parle alors de minimisation du risque empirique. Cependant, la minimisation de ce type de critère ne garantit pas la minimisation du risque réel qui correspond à l'espérance des coûts sur les données à venir. En effet, considérons le cas de données linéairement séparables : s'il existe généralement une infinité d'hyperplans qui permettent de séparer les données d'apprentissage, tous ne permettent pas de bien généraliser, c'est-à-dire de bien classer les données à venir. Un exemple simple est illustré à la figure 4. La droite permet effectivement de séparer les données de la première classe, représentées par des plus (+), de celles de la seconde classe, représentées par des points (•), mais cette frontière de décision conduirait à classer la donnée de test, représentée par une croix (×), comme appartenant à la seconde classe, alors que cette donnée appartient probablement à la première classe.

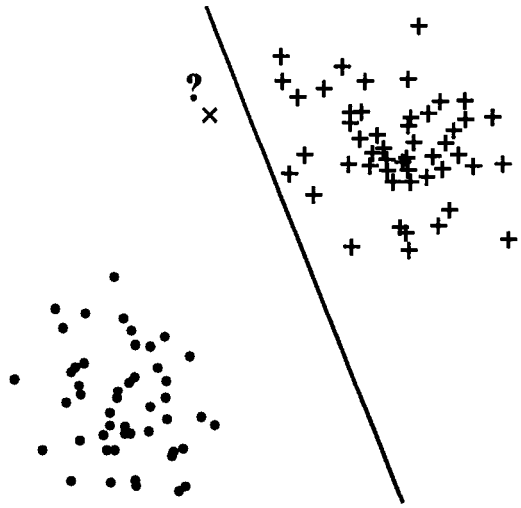


Figure 4 Exemple de frontière de décision sous-optimal

De ce fait, partant du principe que toutes les données à venir qui seront proches d'au moins une des données d'apprentissage devraient être classées dans la même classe que celle-ci, l'idée consiste alors à rechercher la frontière de décision qui permettrait non seulement de bien classer l'ensemble des données d'apprentissage, mais aussi toutes les données qui sont à une distance *r* de l'une des données d'apprentissage. Par conséquent, plus on augmente la valeur de *r*, plus l'on restreint le domaine des solutions, jusqu'à tendre vers une solution unique qui est représentée à la figure 5.

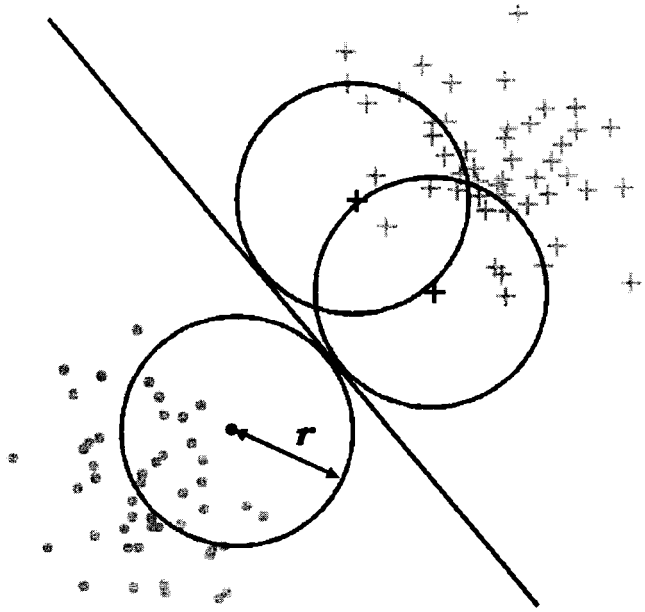


Figure 5 Exemple de frontière de décision optimal

Cet hyperplan peut être considéré comme optimal dans le sens où il maximise la marge de séparation, c'est-à-dire la distance entre lui-même et les données d'apprentissage les plus proches de lui qui sont appelées vecteurs de support (voir figure 6).

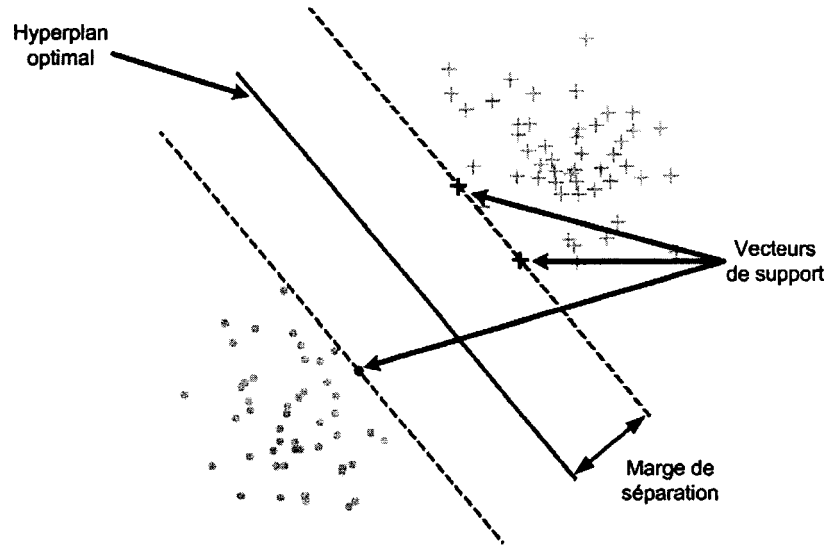


Figure 6 Hyperplan optimal, marge de séparation et vecteurs de support

En pratique, un hyperplan est constitué de l'ensemble des points $x \in \mathbb{R}^d$ vérifiant :

$$w \cdot x + b = 0 \tag{1.1}$$

et est donc caractérisé par un vecteur de poids $w \in \mathbb{R}^d$ et une valeur de biais $b \in \mathbb{R}$.

La fonction de décision associée à un hyperplan est donc :

$$f(x) = w \cdot x + b \tag{1.2}$$

La règle de décision consiste alors à classer la donnée x dans la première classe, caractérisée par le label $y = 1$, si $f(x) \geq 0$, ou dans la seconde classe caractérisée par le label $y = -1$, si $f(x) < 0$.

Ainsi, étant donné un ensemble de données d'apprentissage $\{x_i \in \mathbb{R}^d : i = 1,...,n\}$ et de labels $\{y_i \in \{1,-1\} : i = 1,...,n\}$, si les données des deux classes sont linéairement séparables, il existe au moins un hyperplan qui vérifie :

$$f(x_i)y_i > 0\,,\ (i = 1,...,n) \tag{1.3}$$

La recherche de l'hyperplan optimal, tel que défini précédemment, consiste donc à déterminer les valeurs de w et b qui vérifient les contraintes (1.3) et maximisent la marge de séparation qui correspond à la distance minimale entre les données d'apprentissage et leurs projections sur cet hyperplan et qui est égale à :

$$\min_{i=1,...,n}\left\{\frac{|w \cdot x_i + b|}{\|w\|}\right\} \tag{1.4}$$

Cependant, notons que si l'on multiplie w et b par n'importe quelle constante positive, l'hyperplan reste inchangé. Pour supprimer cette redondance inutile, on utilise par convention l'hyperplan canonique qui est défini par :

$$\min_{i=1,...,n}\{|w \cdot x_i + b|\} = 1 \tag{1.5}$$

Ce qui signifie que pour l'ensemble des vecteurs de support, la valeur de la fonction de décision $f(x)$ est égal soit à 1, soit à -1, selon la classe du vecteur de support. En outre, la marge de séparation est alors égale à $\frac{1}{\|w\|}$.

Déterminer l'hyperplan optimal revient donc à résoudre un problème d'optimisation qui dans sa formulation primale consiste à minimiser :

$$\frac{1}{2}\left\|w^2\right\| \tag{1.6}$$

sous les contraintes :

$$y_i(w\cdot x_i+b)\geq 1,\ (i=1,...,n) \tag{1.7}$$

Notons que le « > 0 » des contraintes (1.3) a été remplacé par « ≥ 1 » qui correspond au choix de l'hyperplan canonique et permet de garantir l'unicité de la solution. Une manière de résoudre ce type de problème d'optimisation sous contraintes est d'introduire les multiplicateurs de Lagrange :

$$\alpha_i \geq 0\ \ (i=1,...,n) \tag{1.8}$$

et le Lagrangien :

$$L(w,b,\alpha)=\frac{1}{2}\|w\|^2-\sum_{i=1}^{n}\alpha_i\left(y_i(w\cdot x_i+b)-1\right) \tag{1.9}$$

Le Lagrangien L doit alors être minimisé par rapport à w et b et maximisé par rapport aux α_i. Par conséquent, au point d'équilibre, nous devons obtenir :

$$\frac{\partial L(w,b,\alpha)}{\partial w}=0 \text{ et } \frac{\partial L(w,b,\alpha)}{\partial b}=0 \tag{1.10}$$

ce qui entraîne :

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{1.11}$$

et :

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i \tag{1.12}$$

La formulation duale du problème d'optimisation revient ainsi à maximiser :

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \tag{1.13}$$

sous les contraintes (1.8) et (1.11).

Notons par ailleurs que seuls les vecteurs de support ont des $\alpha_i \neq 0$ et que l'hyperplan optimal est donc caractérisé par ce sous-ensemble des données d'apprentissage, ainsi que par les multiplicateurs de Lagrange associés à chacune des données de ce sous-ensemble. Le vecteur de poids *w* peut ainsi être calculé à l'aide de l'équation (1.12) et le biais *b* en utilisant deux des vecteurs de support :

$$b = \frac{1}{2}\left(w \cdot x_1 + w \cdot x_2\right) \tag{1.14}$$

où x_1 est un vecteur de support de la classe positive et x_2 de la classe négative.

Cependant, depuis le début de ce chapitre, nous avons toujours considéré que les données d'apprentissage sont linéairement séparables. Si ce n'est pas le cas, la méthode décrite précédemment ne fonctionne pas. De plus, comme nous pouvons le constater à la figure 7, même si les données d'apprentissage sont linéairement séparables, en présence d'une donnée aberrante, la maximisation de la marge de séparation peut conduire à l'obtention d'un hyperplan qui n'est plus du tout optimal.

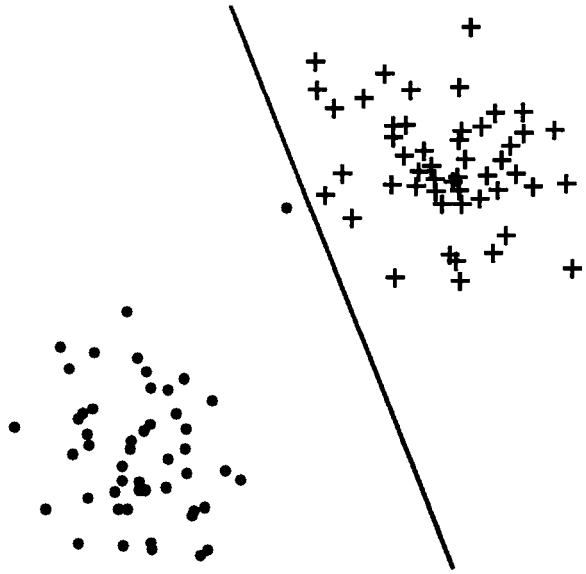


Figure 7 Hyperplan sous-optimal lié à la présence d'une donnée aberrante

Pour contourner ce problème, Cortes et Vapnik [10] introduisent la notion de classifieurs à marge molle (« soft margin » en anglais) qui consiste à tolérer que certaines données d'apprentissage violent les contraintes de séparation (1.7). Ils proposent ainsi d'introduire des variables d'écart (« slack variables » en anglais) :

$$\xi_i \geq 0 \ (i = 1,...,n) \tag{1.15}$$

et de nouvelles contraintes :

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i \quad (i = 1,...,n) \tag{1.16}$$

Bien entendu, si les valeurs des ξ_i sont assez grandes, les contraintes (1.16) seront vérifiées quelque soit l'hyperplan. Il est donc nécessaire de pénaliser les valeurs fortes de ξ_i en introduisant le terme $\sum_{i=1}^{n} \xi_i$ dans la fonction objective (1.6).

La formulation primale du problème d'optimisation revient alors à minimiser sous les contraintes (1.15) et (1.16) la fonction objective suivante :

$$\frac{1}{2}\left\|w^2\right\| + C\sum_{i=1}^{n} \xi_i \tag{1.17}$$

où C est une constante positive, dite de régularisation. Ainsi, plus la valeur de C est grande, moins l'algorithme tolère que les données violent les contraintes de séparation (1.7).

Si l'on reprend l'exemple précédent, on peut constater à la figure 8 que cette méthode permet de retrouver l'hyperplan optimal en associant à la donnée aberrante une variable d'écart différente de zéro.

Finalement, dans le cas de la formulation duale, on maximise toujours l'expression (1.13) sous les contraintes (1.8) et (1.11), aux quelles on ajoute les contraintes :

$$\alpha_i \leq C \quad (i = 1,...,n) \tag{1.18}$$

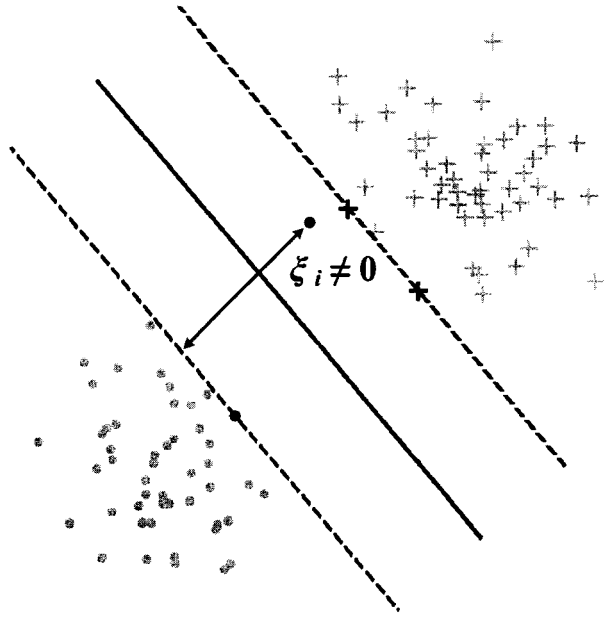


Figure 8 Classifieur à marge molle

Notons enfin que les données qui violent les contraintes de séparation (1.7) ont des multiplicateurs de Lagrange différents de zéro ($\alpha_i \neq 0$) et sont donc sélectionnées comme vecteurs de support.

1.2 L'astuce du noyau

Comme nous l'avons précisé en introduction de ce chapitre, le principe sous-jacent aux SVM consiste à utiliser une transformation non-linéaire pour redécrire le problème dans un espace de plus grande dimension, dans lequel elles pourront être séparées par un hyperplan. Nous venons donc de voir comment déterminer l'hyperplan qui permettra la meilleure généralisation, reste à savoir comment faire d'un point de vue pratique lorsque l'espace de redescription atteint des dimensions faramineuses.

La solution pour contourner ce problème a été proposée en 1992 par Boser *et al.* [11] et est aujourd'hui connue sous le nom « d'astuce du noyau » (« kernel trick » en anglais). Étant donné que la formulation duale du problème d'optimisation est uniquement basée sur le calcul de produits scalaires, l'idée consiste alors à trouver une fonction, que l'on nomme noyau, qui correspond à un produit scalaire dans l'espace de redescription, mais qui ne nécessite pas de redécrire les données dans cet espace pour en calculer sa valeur.

Si l'on reprend l'exemple de la figure 2, les données caractérisées par des vecteurs de dimension 2 étaient redécrites dans un espace de dimension 3 via la transformation non-linéaire suivante :

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \xrightarrow{\Phi} z = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}\, x_1 x_2 \end{bmatrix} \tag{1.19}$$

Le produit scalaire entre deux vecteurs de cet espace de redescription est alors égal à :

$$\begin{bmatrix} {x_1}^2 \\ {x_2}^2 \\ \sqrt{2}\, x_1 x_2 \end{bmatrix} \cdot \begin{bmatrix} {x_1'}^2 \\ {x_2'}^2 \\ \sqrt{2}\, x_1' x_2' \end{bmatrix} = {x_1}^2 {x_1'}^2 + {x_2}^2 {x_2'}^2 + 2 x_1 x_1' x_2 x_2' = (x_1 x_1' + x_2 x_2')^2 \tag{1.20}$$

Ainsi, en remplaçant dans la formulation duale du problème d'optimisation, le produit scalaire entre deux données $x \cdot x'$ par l'expression du noyau :

$$k(x,x') = \Phi(x) \cdot \Phi(x') = (x \cdot x')^2 \tag{1.21}$$

il est possible de déterminer directement la fonction de décision correspondant à la frontière de séparation non-linéaire de la figure 3, sans jamais passer par l'espace de redescription.

En définitif, l'apprentissage d'un SVM consiste donc à minimiser la fonction objective :

$$\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j k(x_i \cdot x_j) \tag{1.22}$$

sous les contraintes :

$$\sum_{i=1}^{n}\alpha_i y_i = 0 \text{ et } 0 \le \alpha_i \le C \quad (i = 1,...,n) \tag{1.23}$$

Pour ce faire, de nombreuses méthodes peuvent être utilisées. Une description des approches les plus courantes est présentée dans le chapitre 10 du livre de Schölkopf et Smola [12]. Nous ne rentrerons pas dans les détails de ces méthodes, mais nous remarquerons que quelque soit la méthode, la complexité de la procédure d'apprentissage augmente plus que linéairement avec le nombre d'exemples d'apprentissage.

Après l'apprentissage, la fonction de décision du SVM est égale à :

$$f(x) = \sum_{i=1}^{n} y_i \alpha_i k(x, x_i) + b \tag{1.24}$$

et la règle de décision consiste, comme dans le cas de l'hyperplan, à classer la donnée x dans la classe caractérisée par le label $y = 1$, si $f(x) \ge 0$, ou dans la classe caractérisée

par le label $y=-1$, si $f(x)<0$. En d'autres termes, le label d'une donnée inconnue x est égal à :

$$y = sign\big(f(x)\big) \tag{1.25}$$

Ainsi, le SVM est caractérisé par les données d'apprentissage pour lesquelles $\alpha_i \neq 0$, par les coefficients α_i associés à chacune de ces données que l'on nomme vecteurs de support et par un biais b. La valeur de ce biais peut être calculée en effectuant la moyenne, pour tous les vecteurs de support pour lesquels $\alpha_i < C$, des valeurs de :

$$b_i = y_i - \sum_{j=1}^{n} y_i \alpha_j k(x_i, x_j) \tag{1.26}$$

1.3 Les hyper-paramètres

Comme pour la plupart des algorithmes d'apprentissage, l'utilisation de SVM nécessite de fixer au préalable certains paramètres qui ne peuvent pas être modifiés lors de l'apprentissage. De manière à différentier ce type de paramètres, des paramètres qui sont optimisés durant l'apprentissage, on parle généralement d'hyper-paramètres. La constante de régularisation C en est un bon exemple. Un autre exemple est celui du choix du noyau qui sera utilisé par le SVM. En effet, non seulement il existe différents types de noyaux, mais chacun d'entre eux est généralement caractérisé par une ou plusieurs constantes. Si l'on reprend l'exemple précédent, le noyau (1.21) est en fait un cas particulier de noyau polynomial dont l'expression courante est :

$$k(x,x') = \big(A\, x \cdot x' + B\big)^p \tag{1.27}$$

Dans ce cas, le nombre d'hyper-paramètres lié au noyau est donc égal à trois. Cependant, les paramètres A et B de l'équation (1.27) sont généralement fixés arbitrairement à un et seul le degré p du polynôme, est optimisé.

Un autre noyau très populaire est le noyau Gaussien dont l'expression courante est :

$$k(x,x') = \exp\left(-\frac{\|x-x'\|^2}{\sigma^2}\right) \tag{1.28}$$

Ce noyau nécessite donc l'optimisation d'un seul hyper-paramètre : $\gamma = \frac{1}{\sigma^2}$. De plus, si ce paramètre est plus délicat à ajuster que le degré p du polynôme, une optimisation fine de celui-ci permet généralement d'obtenir de meilleurs résultats que lors de l'utilisation d'un noyau polynomial, dont les paramètres A et B ont été fixés à un.

Afin d'illustrer l'effet des hyper-paramètres, nous avons choisi d'utiliser des données artificielles en deux dimensions. Pour chacune des classes, 120 exemples ont alors été générés en utilisant les distributions suivantes :

$$p(x \mid y=1) = \frac{1}{2}N\left(\begin{pmatrix}+2\\+2\end{pmatrix},\begin{pmatrix}0.81 & 0\\0 & 0.81\end{pmatrix}\right) + \frac{1}{2}N\left(\begin{pmatrix}-2\\-2\end{pmatrix},\begin{pmatrix}0.81 & 0\\0 & 0.81\end{pmatrix}\right) \text{ et} \tag{1.29}$$

$$p(x \mid y=-1) = \frac{1}{2}N\left(\begin{pmatrix}-2\\+2\end{pmatrix},\begin{pmatrix}0.81 & 0\\0 & 0.81\end{pmatrix}\right) + \frac{1}{2}N\left(\begin{pmatrix}+2\\-2\end{pmatrix},\begin{pmatrix}0.81 & 0\\0 & 0.81\end{pmatrix}\right) \tag{1.30}$$

où $N(\mu,\Sigma)$ est une distribution normale de moyenne μ et de covariance Σ.

Les données obtenues sont présentées à la figure 9. Celles qui appartiennent à la classe correspondant au label $y=1$ sont représentées par des points (•), tandis que celles appartenant à la classe correspondant au label $y=-1$ sont représentées par des plus (+).

Ces données forment une base d'apprentissage qui est utilisée pour entraîner différents SVM correspondant à différentes valeurs des hyper-paramètres. Le noyau utilisé est le noyau Gaussien (1.28). Nous avons donc deux hyper-paramètres : C et γ.

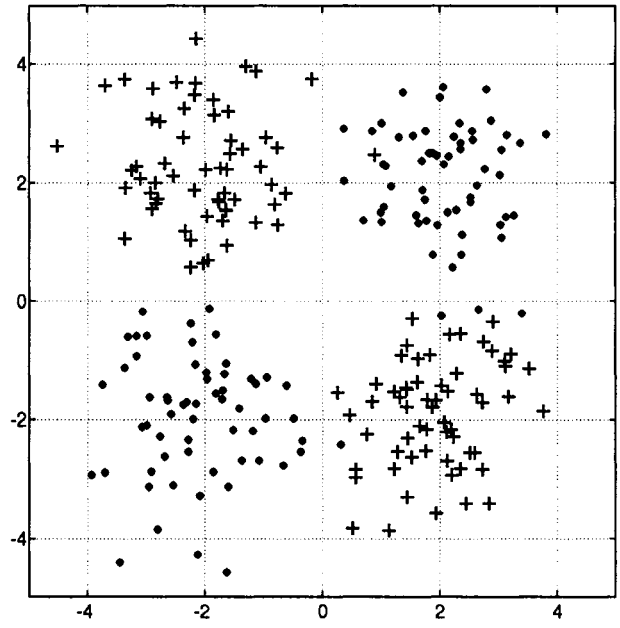


Figure 9 Données artificielles utilisées pour illustrer l'effet des hyper-paramètres

Dans un premier temps, le paramètre de régularisation a été fixé arbitrairement à $C=1$ et six apprentissages ont été réalisés en utilisant six valeurs différentes de γ. Les performances en généralisation ont ensuite été évaluées sur une base de test indépendante formée de 10 000 données supplémentaires qui ont été générées à partir des mêmes distributions (1.29) et (1.30).

Les résultats, en termes de taux d'erreur sur la base de test, mais aussi de nombre de vecteurs de support sont reportés dans le tableau II. Il est alors possible de constater que pour les valeurs extrêmes $\gamma = 10^{-3}$ et $\gamma = 10^{2}$, le taux d'erreur est vraiment très important et toutes les données d'apprentissage ou presque sont vecteurs de support. D'autre part, tandis que le meilleur résultat, à la fois en termes de taux d'erreur et de nombre de vecteurs de support, est obtenu pour $\gamma = 10^{-1}$, nous pouvons constater que pour des valeurs intermédiaires, $\gamma = 10^{-2}$ et $\gamma = 10^{0}$, le taux d'erreur n'est que légèrement plus élevé, mais le nombre de vecteurs de support augmente fortement.

Tableau II

Effet de l'hyper-paramètre du noyau Gaussien sur la généralisation et la complexité

Valeur du paramètre γ	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
Taux d'erreur en test (%)	44.37	3.30	2.92	3.14	5.05	17.85
Nombre vecteurs de support	240	187	47	72	207	239

De manière à essayer de mieux comprendre l'effet de ce paramètre, nous avons visualisé les frontières de décision correspondant à ces différents SVM. Les résultats sont présentés à la figure 10, où les vecteurs de support sont représentés en noir, tandis que les données non retenues sont représentées en gris.

Ainsi, nous pouvons constater que l'utilisation d'un γ trop grand, conduit à un important sur-apprentissage des données, tandis qu'une valeur trop faible entraîne un problème de sous-apprentissage. Il est donc important de bien choisir le noyau qui sera utilisé et de surtout bien optimiser le ou les paramètres du noyau choisi.

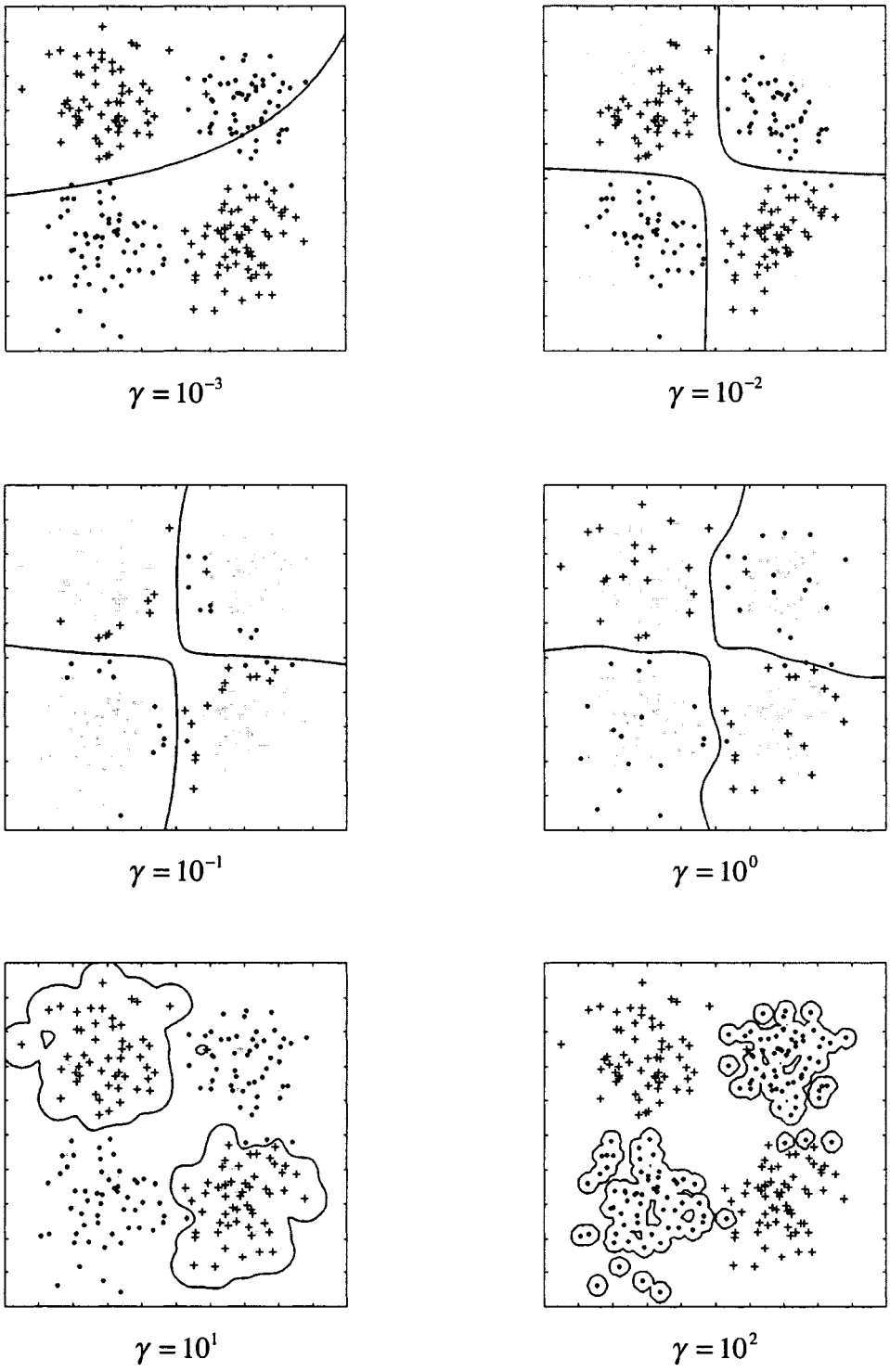


Figure 10 Effet l'hyper-paramètre du noyau Gaussien sur les frontières de décision

Dans un second temps, nous avons fixé $\gamma = 10^{-1}$ et nous avons effectué cinq apprentissages supplémentaires correspondant à cinq nouvelles valeurs de C. Les résultats en termes de taux d'erreur sur la base de test et de nombre de vecteurs de support sont reportés dans le tableau III. Il est alors possible de constater que plus C est petit, plus le nombre de vecteurs de support est grand. Cependant, l'utilisation d'une valeur de C trop grande conduit certes à l'obtention de peu de vecteurs de support, mais aussi à des taux d'erreur plus élevés. Par ailleurs, une valeur de C trop faible conduit non seulement à sélectionner toutes les données comme vecteurs de support, mais en plus à des taux d'erreur légèrement plus élevés.

Tableau III

Effet de l'hyper-paramètre de régularisation sur la généralisation et la complexité

Valeur du paramètre C	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}	10^{2}
Taux d'erreur en test (%)	2.94	2.94	2.89	2.92	3.23	3.79
Nombre vecteurs de support	240	240	135	47	23	19

Comme précédemment, nous avons visualisé les frontières de décision correspondant à ces différents SVM, de manière à essayer de mieux cerner l'effet de ce paramètre.

Les résultats sont présentés à la figure 11 où les vecteurs de support sont toujours représentés en noir et les données non retenues en gris. De plus, les lignes de niveau correspondant à $f(x) = -1$ et $f(x) = +1$ sont représentées en pointillé de manière à visualiser la marge de séparation et l'effet du paramètre de régularisation sur celle-ci.

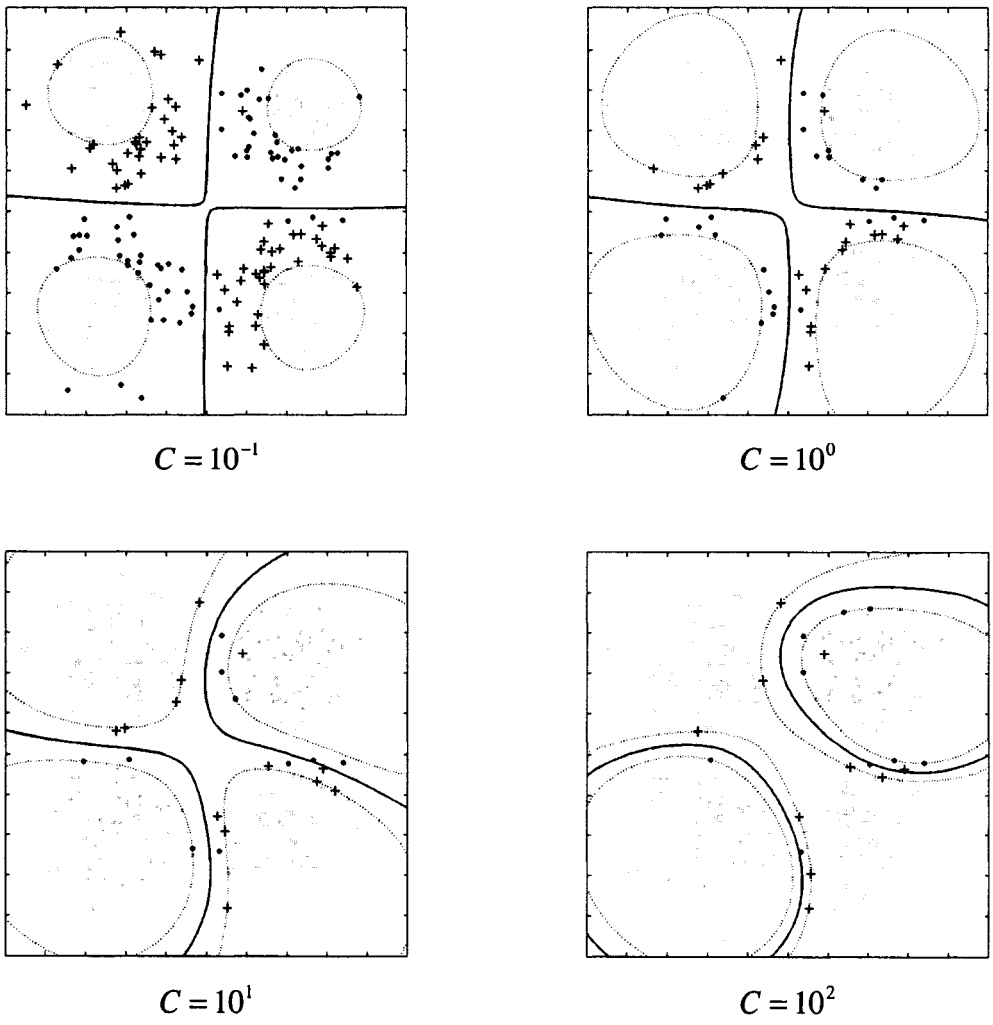


Figure 11 Effet du paramètre de régularisation sur les frontières de décision

Nous pouvons alors constater que l'augmentation du nombre de vecteurs de support, qui accompagne l'utilisation de valeurs de C petites, est directement lié à l'augmentation de la marge de séparation et donc du nombre de données situées à l'intérieur de celle-ci. Par ailleurs, nous pouvons constater un léger sur-apprentissage des données lors de l'utilisation d'une valeur de C trop grande.

Il est donc particulièrement important de trouver les « bons » hyper-paramètres. De plus ces hyper-paramètres ne sont pas indépendants et doivent donc être optimisés conjointement. La solution la plus simple est d'utiliser une grille de recherche comme cela est recommandé par Hsu *et al.* [13]. La méthode consiste alors à échantillonner les valeurs de C et de γ, à entraîner un SVM pour chaque combinaison et à évaluer la capacité à généraliser de chaque SVM sur une base de validation. Bien entendu, cette méthode peut s'avérer particulièrement fastidieuse pour des problèmes de grandes dimensions, où l'apprentissage d'un SVM peut être relativement long. Ainsi, différentes méthodes plus élaborées ont été proposées pour optimiser ces hyper-paramètres. Un état de l'art des différentes approches existantes est présentée dans la thèse de doctorat de Nedjem Eddine Ayat [2] qui propose une nouvelle approche basée sur la minimisation de l'erreur empirique.

1.4 Conclusion

Pour conclure cette brève présentation des machines à vecteurs de support, nous pouvons dire que cet algorithme d'apprentissage permet de résoudre efficacement des problèmes de classification à deux classes. Cependant, la lecture automatique de l'écriture manuscrite, tout comme la majorité des applications en reconnaissance de formes, nécessite de traiter des problèmes de classification où le nombre de classes est supérieur à deux. Ainsi, l'objectif du prochain chapitre est de montrer comment résoudre efficacement des problèmes multi-classes à l'aide de machines à vecteurs de support.

CHAPITRE 2

RÉSOLUTION DE PROBLÈMES MULTI-CLASSES

Comme nous l'avons vu dans le premier chapitre, dans sa formulation originale, un SVM ne permet de traiter que des problèmes à deux classes. Bien entendu, de nombreuses solutions ont été proposées pour étendre l'utilisation des SVM aux problèmes multi-classes. Cependant, aucune des approches existantes ne s'est réellement imposée. Ainsi, la résolution de problèmes multi-classes à l'aide de SVM reste encore aujourd'hui un sujet de recherche actif. Nous proposons donc d'effectuer dans ce chapitre un état de l'art des approches existantes et de comparer expérimentalement deux d'entre elles.

2.1 État de l'art

2.1.1 La stratégie « un contre tous »

La solution la plus simple pour résoudre un problème multi-classes à l'aide de SVM consiste à le décomposer en un ensemble de sous-problèmes binaires et à construire indépendamment un SVM pour chacun d'entre eux. Ainsi, Cortes et Vapnik [10] proposent d'utiliser une stratégie de décomposition très intuitive et facile à mettre en place. Cette stratégie, communément nommée « un contre tous », consiste à construire autant de SVM qu'il y a de classes. Chaque SVM est alors entraîné à séparer les données d'une classe qui seront étiquetées +1, de celles de toutes les autres classes qui seront étiquetées −1. Chaque SVM est ainsi associé à une classe et sa sortie avant seuillage peut être considérée comme une mesure d'appartenance à la classe. La règle de décision généralement utilisée consiste donc à attribuer la donnée inconnue à la classe correspondant au SVM ayant la plus grande valeur de sortie. Cependant, étant donné que les sorties des SVM ne sont pas calibrées, la décision peut parfois être faussée. À partir

de cette constatation, Mayoraz et Alpaydin [14] proposent soit de calibrer indépendamment les sorties de chaque SVM, soit d'utiliser un perceptron qui recevra les sorties brutes des différents SVM et retournera des valeurs calibrées. Dans le même esprit, il peut être intéressant d'utiliser les sorties des différents SVM pour estimer les probabilités *a posteriori* d'appartenance aux différentes classes et d'utiliser ensuite ces valeurs pour prendre la décision. Plusieurs méthodes permettant ceci sont décrites dans le prochain chapitre. Notons que nous utiliserons l'acronyme anglais OAA (One Against All) pour dénommer cette stratégie.

2.1.2 La stratégie « un contre un »

Une autre stratégie classique consiste à construire un SVM pour chaque paire de classes, soit $c(c\text{-}1)/2$ SVM pour un problème à c classes. Chaque classifieur est donc entraîné à séparer les données d'une classe de celles d'une autre classe. Nous parlerons alors de stratégie « un contre un ». Notons par ailleurs, que cette stratégie peut aussi être rencontrée dans la littérature sous les noms de « pairwise coupling » [15-21], « all pairs » [22], ou encore « round robin » [23]. La règle de décision généralement utilisée est le vote majoritaire. Chaque SVM vote alors pour une classe et la donnée inconnue est finalement associée à la classe ayant reçu le plus grand nombre de votes. Cependant, il est possible que plusieurs classes reçoivent le même nombre de votes. Alors, partant du principe que les cas d'égalité sont très peu fréquents, Hsu et Lin [24] proposent de prendre la décision de manière aléatoire. Kreßel [17] propose quant à lui d'associer la donnée inconnue à la classe la plus proche, qui est déterminée en considérant parmi les classes à départager la valeur de sortie négative de plus faible valeur absolue. Une autre solution pour contourner ce problème est proposée par Savicky et Fürnkranz [20] qui utilisent un classifieur naturellement multi-classes pour prendre la décision à partir de l'ensemble des sorties des classifieurs binaires. Par ailleurs, une approche originale est proposée par Platt *et al.* [25] qui proposent d'utiliser un arbre de décision dont chaque nœud est un SVM et chaque feuille une des classes du problème. Le principe de leur

méthode nommée DAG (pour Directed Acyclic Graph) est illustré à la figure 12. On peut alors constater que chaque SVM a pour rôle d'éliminer une hypothèse et qu'ainsi la prise de décision nécessite d'évaluer la sortie que de c-1 SVM.

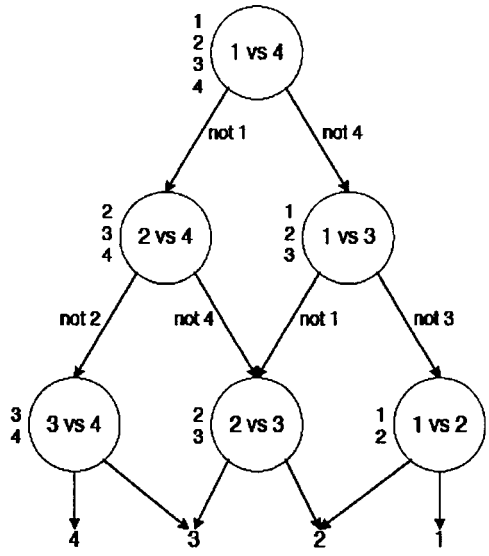


Figure 12 Illustration du principe du DAG (extrait de [25])

Pour finir, il est aussi possible d'estimer des probabilités « locales » au niveau de chaque SVM, puis de les combiner ensemble de manière à estimer les probabilités d'appartenance à chacune des classes du problème multi-classes. Pour ce faire, il existe différentes méthodes, dont un état de l'art est présenté dans le prochain chapitre. Notons que nous utiliserons l'acronyme anglais OAO (One Against One) pour dénommer cette stratégie.

2.1.3 Les autres stratégies

D'autres schémas de décomposition moins intuitifs que le « un contre tous » et le « un contre un » peuvent aussi être utilisés. Ainsi, Schwenker [26] propose d'utiliser le même principe d'arbres de décision que Platt *et al.* [25], mais ne se limite pas uniquement à l'utilisation de SVM issus de la stratégie « un contre un ». Un premier exemple d'arbre

de décision possible pour résoudre un problème à 6 classes est présenté à la figure 13-a. Le premier nœud est alors un SVM « trois contre trois », les deux nœuds suivants sont de type « deux contre un », tandis que les deux derniers nœuds sont de type « un contre un ». Un autre exemple d'arbre possible pour résoudre le même problème est présenté à la figure 13-b. Le premier nœud est alors un SVM « deux contre deux », tout comme les deux nœuds suivants, tandis que les trois derniers nœuds sont de type « un contre un ».

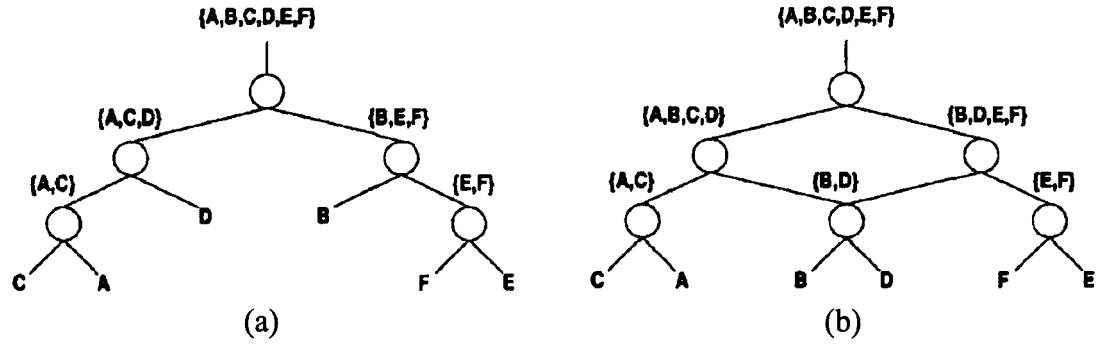


Figure 13 Exemples d'arbre de décision (extrait de [26])

Un autre type de stratégie basée sur la théorie des codes correcteurs d'erreurs (ECOC, pour Error-Correcting Output Codes) a été proposée par Dietterich et Bakiri [27]. Un exemple de code exhaustif est présenté au tableau IV. On peut alors constater que pour un problème à quatre classes, sept classifieurs binaires sont construits en utilisant toutes les données mais un étiquetage différent. Dans cet exemple, les quatre premières colonnes correspondent donc à la stratégie « un contre tous » tandis que les trois dernières colonnes correspondent à une décomposition « deux contre deux ». La décision sera enfin prise en calculant les distances de Hamming entre le vecteur contenant les sorties après seuillage des sept SVM et les quatre vecteurs lignes nommés « codewords » correspondant à chacune des classes. Bien entendu, l'utilisation de code exhaustif semble peu réaliste pour des problèmes où le nombre de classes est important. En effet, cette stratégie nécessite l'entraînement de $2^{c-1}-1$ SVM. Ainsi, 511 SVM seraient nécessaires

pour la reconnaissance de chiffres ($c = 10$) et plus de 33 millions pour la reconnaissance de lettres ($c = 26$). Les auteurs proposent donc, en fonction du nombre de classes du problème, différentes approches pour l'élaboration d'un « bon » code correcteur d'erreurs, c'est-à-dire un code qui permet de réduire le nombre d'erreurs tout en utilisant un nombre raisonnable de classifieurs binaires.

Tableau IV

Exemple de code correcteur d'erreurs exhaustif ($c = 4$)

	f_1	f_2	f_3	f_4	f_5	f_6	f_7
ω_1	+1	−1	−1	−1	+1	+1	+1
ω_2	−1	+1	−1	−1	+1	−1	−1
ω_3	−1	−1	+1	−1	−1	+1	−1
ω_4	−1	−1	−1	+1	−1	−1	+1

Par ailleurs, plus récemment, Allwein *et al.* [22] ont repris et étendu le concept de ECOC en introduisant d'une part l'étiquette « 0 » qui signifie que les données de la classe correspondante ne seront pas prises en compte lors de l'apprentissage et d'autre part une mesure plus fine qui utilise les sorties des SVM avant seuillage. Notons enfin que Crammer et Singer [28] ont proposé une approche originale pour l'élaboration du code correcteur d'erreurs, qui consiste à utiliser non plus un codage discret mais continu, dont les valeurs seront déterminées par apprentissage.

Enfin, une autre solution pour résoudre un problème multi-classes consiste à modifier la formulation du SVM de manière à construire directement un classifieur multi-classes en résolvant un unique problème d'optimisation. Une première approche a été proposée indépendamment par Vapnik [29] et par Weston et Watkins [30]. Par la suite, différentes

variantes ont été proposées par Bredensteiner et Bennett [31], par Lee *et al.* [32], par Crammer et Singer [33] ou encore par Guermeur [34]. Toutes ces méthodes sont basées sur le même schéma que la stratégie « un contre tous » sauf que les différentes frontières sont obtenues en résolvant de manière globale un unique problème d'optimisation, au lieu de résoudre séparément un ensemble de sous-problèmes locaux.

2.1.4 Études comparatives

Une première étude comparative a été réalisée par Hsu et Lin [24]. Cinq approches sont alors comparées sur différentes bases de données couramment utilisées dans le domaine du « machine learning ». Les résultats observés ne permettent pas de conclure que l'une des approches est supérieure aux autres en termes de précision. Finalement, les auteurs concluent que les approches basées sur la stratégie « un contre un » sont plus appropriées d'un point de vue pratique, car beaucoup plus rapide à entraîner que les approches basées sur la stratégie « un contre tous ».

Une seconde étude comparative a été réalisée par Rifkin et Klautau [35]. L'objectif est alors de combattre l'idée soutenue par différents auteurs, tels que Allwein *et al.* [22] ou Fürnkranz [23], selon laquelle la stratégie « un contre tous » serait moins efficace en terme de généralisation que des approches telles que les codes correcteurs d'erreurs ou la stratégie « un contre un ». Pour ce faire, Rifkin et Klautau [35] se basent sur une analyse critique de la littérature et remettent en cause, expériences à l'appui, un certain nombre de résultats. Ainsi, il semblerait que la principale raison qui favoriserait la stratégie « un contre un » par rapport à la stratégie « un contre tous » serait l'utilisation d'un classifieur dont la capacité à généraliser serait insuffisante. En effet, les résultats présentés par Hsu et Lin [24] et Kreßel [17] montrent clairement que lors de l'utilisation d'un noyau linéaire, la stratégie « un contre un » est significativement plus efficace que la stratégie « un contre tous », mais que ceci n'est plus nécessairement vrai lors de l'utilisation de noyaux autres que le noyau linéaire. Par ailleurs, d'un point de vue théorique, pour que

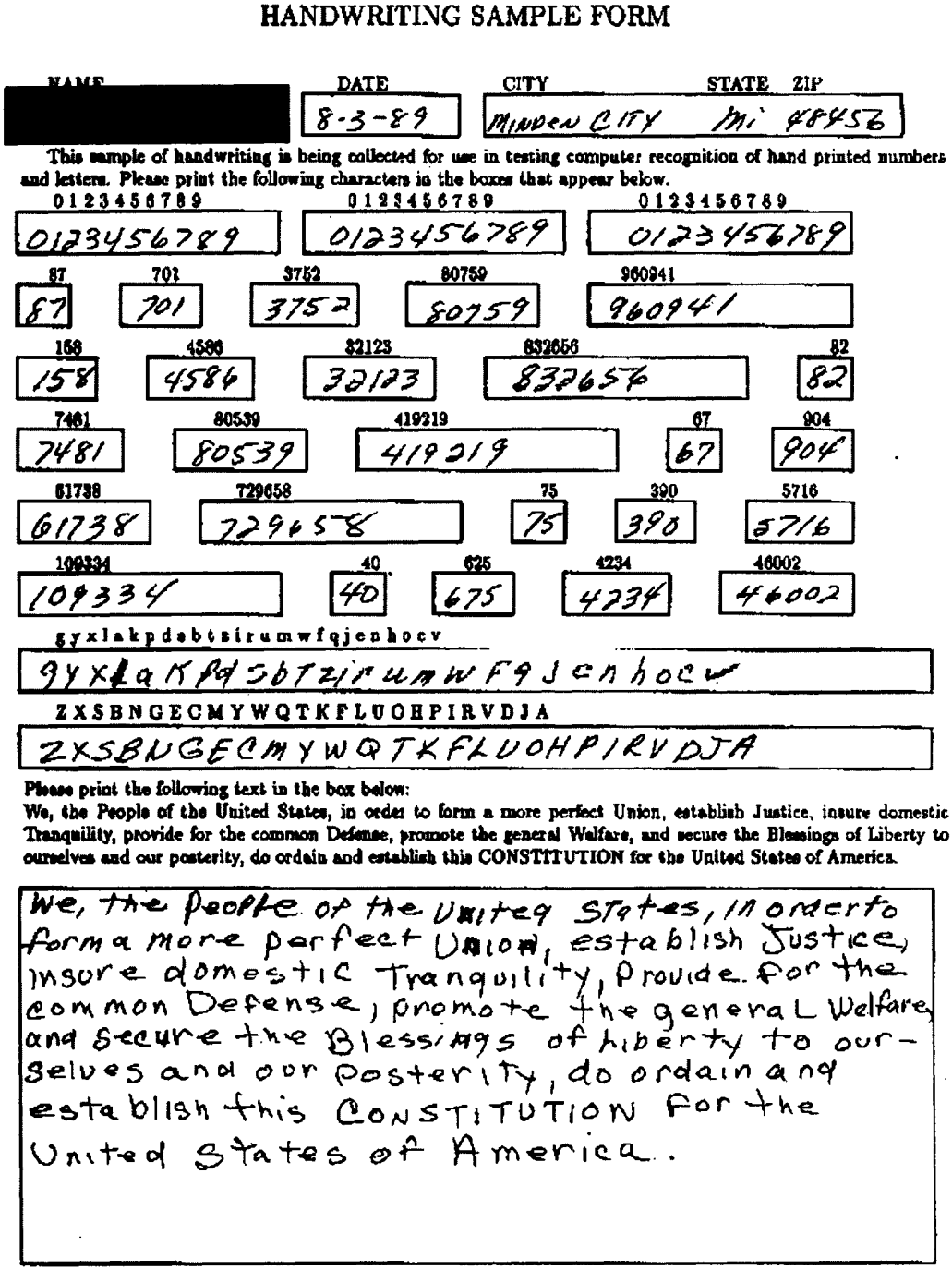
le principe des codes correcteurs d'erreurs fonctionne, il ne faut pas que les différents classifieurs binaires soient trop corrélés, ce qui semble être le cas lors de l'utilisation d'un classifieur très discriminant tel qu'un SVM. Finalement, Rifkin et Klautau [35] en concluent que si l'on utilise comme classifieur binaire un SVM dont les hyper-paramètres sont soigneusement optimisés, la stratégie « un contre tous » est alors aussi précise que n'importe quelle autre approche.

2.2 Résultats expérimentaux

Ainsi, la revue de la littérature ne permet pas de déterminer quelle approche est la mieux adaptée pour résoudre des problèmes multi-classes. Cependant, les conclusions relatives aux études comparatives présentées précédemment soulignent que dans le cas des SVM l'utilisation de stratégies complexes, telles que les stratégies basées sur les codes correcteurs d'erreurs, ne permet pas d'obtenir des résultats plus précis que l'utilisation d'approches aussi simples que la stratégie « un contre tous » ou la stratégie « un contre un ». Nous avons donc choisi de limiter notre étude à ces deux dernières stratégies que nous nous proposons de comparer à la fois en termes de précision et de complexité sur un problème réel de reconnaissance de caractère manuscrits isolés.

2.2.1 Bases de données

L'ensemble des expériences a été réalisé à l'aide des données de la base NIST-SD19 [36]. Cette base de données contient les images binaires de formulaires écrit par plus de 3 600 scripteurs différents. Un exemple de formulaire est présenté à la figure 14. Au total, plus de 800 000 images de caractères manuscrits segmentés ont été extraites de ces formulaires. Pour nos expériences, nous n'utiliserons que les images de chiffres manuscrits dont quelques exemples sont présentés à la figure 15 et les images de lettres majuscules. Les deux types de caractères seront traités séparément comme deux problèmes de classification distincts, un premier comprenant 10 classes et un second comprenant 26 classes.



HANDWRITING SAMPLE FORM

NAME [redacted] DATE 8-3-89 CITY Minden City STATE MI ZIP 48456

This sample of handwriting is being collected for use in testing computer recognition of hand printed numbers and letters. Please print the following characters in the boxes that appear below.

0123456789 0123456789 0123456789

0123456789 0123456789 0123456789

87 701 3752 80759 960941

87 701 3752 80759 960941

158 4586 32123 832656 82

158 4584 32123 832656 82

7481 80539 419219 67 904

7481 80539 419219 67 904

61738 729658 75 390 5716

61738 729658 75 390 5716

109334 40 625 4234 46002

109334 40 675 4234 46002

g y x l a k p d s b t s i r u m w f q j e n h o c v

9 y x l a k p d s b t z i r u m w f q j e n h o c v

Z X S B N G E C M Y W Q T K F L U O H P I R V D J A

Z X S B N G E C M Y W Q T K F L U O H P I R V D J A

Please print the following text in the box below:
We, the People of the United States, in order to form a more perfect Union, establish Justice, insure domestic Tranquility, provide for the common Defense, promote the general Welfare, and secure the Blessings of Liberty to ourselves and our posterity, do ordain and establish this CONSTITUTION for the United States of America.

We, the People of the United States, in order to form a more perfect Union, establish Justice, insure domestic Tranquility, Provide for the common Defense, promote the general Welfare, and secure the Blessings of liberty to ourselves and our Posterity, do ordain and establish this CONSTITUTION for the United States of America.

Figure 14 Exemple d'image de formulaire de la base NIST-SD19

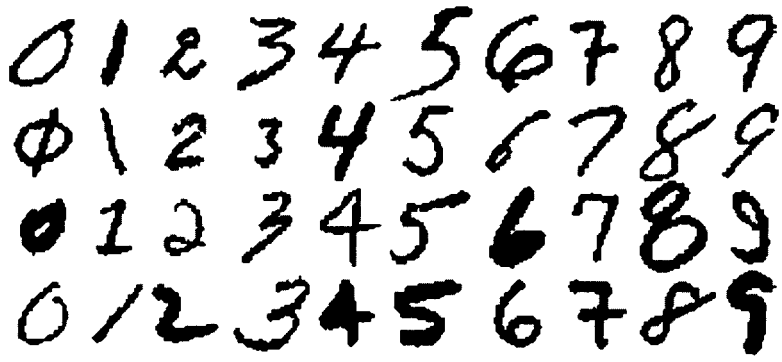


Figure 15 Exemples d'images de chiffres manuscrits isolés extrait des formulaires

Pour chacun des deux problèmes de classification, trois sous-ensembles de données ont été constitués. Une base d'apprentissage utilisée pour l'entraînement des classifieurs, une base de validation utilisée pour optimiser les différents paramètres des classifieurs tout en contrôlant le phénomène de sur-apprentissage et enfin une base de test utilisée pour évaluer la capacité à généraliser des différents classifieurs. Le nombre d'exemples contenus dans chacun de ces sous-ensembles est reporté dans le tableau V. Les bases d'apprentissage contiennent exactement le même nombre d'exemple dans chaque classe, correspondant aux premières images issues des corpus $hsf\_\{0,1,2,3\}$. Pour les chiffres, la base de validation est constituée du reste des images de ce corpus, tandis que pour les lettres elle est constituée de toute les images du corpus hsf_4. Enfin, les bases de test contiennent toutes les images du corpus hsf_7.

Tableau V

Nombre d'exemples dans chacun des sous-ensembles de données

	Chiffres	Lettres majuscules
Apprentissage	195 000	43 160
Validation	28 123	11 941
Test	60 089	12 092

Concernant la procédure d'extraction de caractéristiques, nous utiliserons l'espace de représentation proposé par Oliveira [6]. En effet, ces caractéristiques ont été utilisées avec succès sur la même base d'images de chiffres par Oliveira [6]. Chaque image est alors divisée en six zones : trois lignes et deux colonnes. Dans chaque zone, 22 caractéristiques sont extraites : treize relatives aux concavités, huit relatives aux directions du contour extérieur et une relative à la surface du caractère. Les caractéristiques ainsi extraites sont ensuite normalisées entre 0 et 1 pour finalement former un vecteur de dimension 132.

2.2.2 Résultats de références

De manière à obtenir des résultats de référence, nous avons testé dans un premier temps les classifieurs classiques que sont le *k*-NN et le MLP. Les résultats en termes de taux d'erreur obtenus sur les bases de test sont reportés dans le tableau VI. Notons que le *k*-NN utilise la distance euclidienne et que le nombre de voisins a été fixé en utilisant la base de validation ($k = 1$ pour les chiffres et $k = 3$ pour les lettres). Concernant le MLP, nous avons repris la même topologie de réseau que Oliveira *et al.* [37]. Les réseaux ne possèdent qu'une seule couche cachée dont le nombre de neurones est fixé en utilisant la base de validation (80 pour les chiffres et 100 pour les lettres). Les neurones de la couche d'entrée et de la couche de sortie sont totalement connectés à ceux de la couche cachée et la fonction de transfert utilisée est la sigmoïde. L'optimisation des poids du réseau est réalisée à l'aide d'un algorithme de type gradient stochastique avec terme d'inertie (momentum) et la fonction objective est l'erreur quadratique (SSE, de l'anglais Sum Square Error). D'autre part, nous pouvons remarquer que les résultats obtenus avec le MLP sont légèrement meilleurs que ceux reportés par Oliveira *et al.* [37] (0,83% d'erreur pour les chiffres) ou que ceux que nous avions reportés précédement [38] (0,80% d'erreur pour les chiffres et 3,81 % pour les lettres). En effet, nous utilisions alors la même technique de pas adaptatif proposée par Oliveira [6] qui consiste à débuter l'apprentissage avec un pas important $\eta = 0,5$ et à le diviser par 2 toutes les 25 époques.

Cette technique permet d'accélérer grandement l'apprentissage, mais nous avons constaté récemment qu'elle a pour effet d'arrêter l'apprentissage prématurément. Nous avons donc abandonné cette technique de pas adaptatif pour revenir à un pas constant fixé à $\eta = 0,01$. D'autre part, la constante de momentum est fixée à $\mu = 0,9$ et les poids sont initialisés aléatoirement entre 0 et 1 en utilisant une distribution uniforme. Notons pour finir que la technique d'« early stopping » est utilisée pour arrêter l'apprentissage. Cette technique classique pour éviter le sur-apprentissage consiste à utiliser une base de validation pour évaluer la capacité à généraliser du réseau à la fin de chaque itération et arrêter son apprentissage lorsque le taux d'erreur sur cette base augmente.

Tableau VI

Taux d'erreur obtenus sur les bases de test avec les classifieurs de référence

	Chiffres	Lettres
k-NN	1,35 %	7,60 %
MLP	**0,74 %**	**3,68 %**

Nous pouvons donc constater que le MLP permet d'obtenir sur les deux bases de données, des performances significativement meilleures que celles obtenues à l'aide d'un *k*-NN. En effet, le nombre d'exemples d'apprentissage est suffisamment important par rapport à la dimension de l'espace des caractéristiques. Dans ces conditions, le MLP souffre peu du problème de sur-apprentissage et permet donc d'obtenir de très bonnes performances en généralisation. Ainsi, nous disposons de résultats de référence particulièrement intéressants auxquels nous pourrons comparer les résultats obtenus avec les SVM.

2.2.3 Résultats obtenus avec les SVM

Le logiciel LIBSVM [39] a été utilisé pour réaliser l'apprentissage de l'ensemble des SVM correspondant aux deux stratégies comparées. Les hyper-paramètres ont été fixés empiriquement en cherchant à minimiser les taux d'erreur sur les bases de validation. Finalement, le noyau utilisé est le noyau Gaussien dont le paramètre vaut $\gamma = 0,38$ pour les chiffres et $\gamma = 3,50$ pour les lettres. Le paramètre de pénalisation a quant à lui été fixé à $C = 1000$ pour les chiffres et à $C = 10$ pour les lettres.

2.2.3.1 Résultats en termes de précision

Les taux d'erreur obtenus sur les bases de test sont reportés au tableau VII. La règle de décision utilisée pour la stratégie « un contre un » est le vote majoritaire, tandis que pour la stratégie « un contre tous » la décision est prise en considérant la plus grande valeur de sortie parmi l'ensemble des SVM.

Tableau VII

Taux d'erreur obtenus sur les bases de test avec les SVM

	Chiffres	Lettres
SVM-OAO	0,71 %	3,29 %
SVM-OAA	**0,63 %**	**3,24 %**

Ainsi, nous pouvons constater que la stratégie « un contre tous » est plus précise que la stratégie « un contre un ». De plus, si la différence n'est pas très importante sur la base de lettres, elle est significative sur la base de chiffres. En effet, la stratégie « un contre tous » commet 0,08% moins d'erreur que la stratégie « un contre un », ce qui représente tout de même plus de 10% des erreurs commise par cette stratégie.

2.2.3.2 Résultats en termes de complexité

Deux types de complexité peuvent être considérés : celle liée à la phase d'entraînement et celle liée à la phase de test.

Concernant la complexité de calcul nécessaire à l'entraînement de l'ensemble des SVM, en accord avec la littérature, nous pouvons constater au tableau VIII qu'il est bien plus rapide d'entraîner l'ensemble des SVM de la stratégie « un contre un » que l'ensemble des SVM de la stratégie « un contre tous ». Notons que les apprentissages ont été réalisés sur un ordinateur équipé d'un processeur AMD Athlon cadencé à 1,9 Ghz et muni de 1 Go de RAM.

Tableau VIII

Temps de calcul nécessaire à l'entraînement de tous les SVM

	Chiffres	Lettres
SVM-OAO	**36 min.**	**4 min.**
SVM-OAA	32 h 17 min.	51 min.

De première abord, ce résultat peut sembler surprenant car, comme nous pouvons le constater au tableau IX, il est nécessaire d'entraîner beaucoup plus de classifieurs dans le cas de la stratégie « un contre un » que dans le cas de la stratégie « un contre tous ».

Tableau IX

Nombre de SVM utilisés par chacune des stratégies

	Chiffres	Lettres
SVM-OAO	45	325
SVM-OAA	**10**	**26**

Cependant, comme nous pouvons le constater à la figure 16, le temps d'entraînement d'un SVM augmente bien plus que linéairement avec le nombre de données d'apprentissage. Ainsi, étant donné que chaque SVM de la stratégie « un contre un » est entraîné avec beaucoup moins de données d'apprentissage, il n'est alors pas surprenant que l'entraînement des $c(c\text{-}1)/2$ SVM de la stratégie « un contre un » s'avère nettement plus rapide que l'entraînement des c SVM de la stratégie « un contre tous ». Finalement, dans le cas des lettres, il est environ 12 fois plus rapide d'effectuer l'apprentissage des 325 SVM issus de la stratégie « un contre un » que d'effectuer l'apprentissage des 26 SVM issus de la stratégie « un contre tous ». De même, dans le cas des chiffres, l'apprentissage des 45 SVM issus de la stratégie « un contre un » est 50 fois plus rapide que l'apprentissage des 10 SVM issus de la stratégie « un contre tous ».

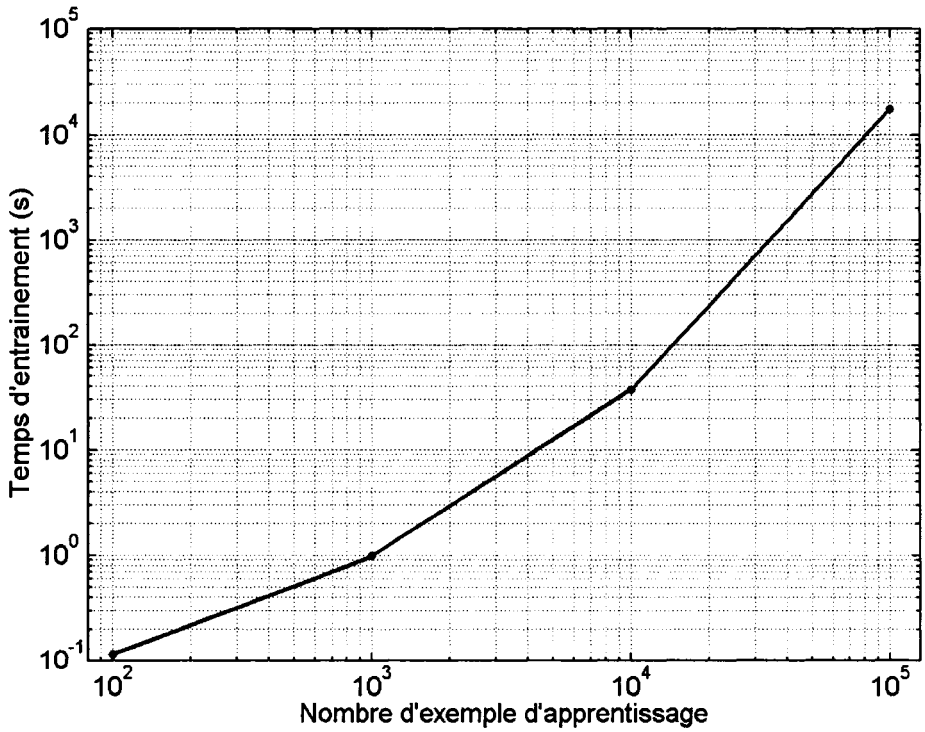


Figure 16 Temps nécessaire à l'entraînement des 10 SVM de la stratégie "un contre tous" sur la base de chiffres

Concernant la prise de décision, il pourrait sembler logique que la stratégie « un contre un » soit plus complexe que la stratégie « un contre tous » car il est nécessaire d'évaluer la sortie d'un plus grand nombre de classifieur. Cependant, dans le cas des SVM la complexité liée à la prise de décision n'est pas proportionnelle au nombre de classifieurs mais au nombre total de vecteurs de support. Notons que le nombre total de vecteurs de support n'est pas égal à la somme du nombre de vecteurs de support de chaque SVM, car une donnée d'apprentissage peut être sélectionnée par plusieurs SVM. Finalement, nous pouvons constater au tableau X que la stratégie « un contre tous » utilise plus de vecteurs que la stratégie « un contre un » (48 % de plus pour les chiffres et 21 % de plus pour les lettres).

Tableau X

Nombre de vecteurs de support

	Chiffres	Lettres
SVM-OAO	**5 753**	**9 152**
SVM-OAA	8 514	11 109

2.3 Conclusion

Plusieurs conclusions se dégagent de ces premiers résultats expérimentaux.

Tout d'abord, les SVM permettent bien d'obtenir des taux d'erreur en généralisation, légèrement plus faibles que ceux obtenus avec un MLP, même lorsque le nombre de données d'apprentissage est suffisamment important.

Ensuite, si la stratégie « un contre tous » semble plus précise que la stratégie « un contre un », elle apparaît aussi plus complexe à la fois au niveau de l'apprentissage et de la prise de décision. De plus, si la différence entre les deux stratégies est significative pour

les chiffres, elle est relativement faible pour les lettres et tout nous laisse à penser que la tendance pourrait très bien s'inverser lorsque le nombre de classes devient très grand comme dans le cas de la reconnaissance d'idéogrammes chinois ou japonais. En effet, dans le cas de la stratégie « un contre tous », le fort déséquilibre entre le nombre de données affectées à la classe positive et le nombre de données affectées à la classe négative risque alors de causer problème.

Cependant, notons que les règles de décision utilisées dans ce chapitre sont très simples et certainement pas optimales. Ainsi, pour comparer plus rigoureusement les deux stratégies, il serait préférable d'utiliser des règles de décision plus évoluées. C'est ce que nous chercherons donc à faire dans le prochain chapitre, en nous intéressant aux méthodes d'estimation de probabilités *a posteriori*.

CHAPITRE 3

ESTIMATION DE PROBABILITÉ A POSTERIORI

En reconnaissance de formes, il est fréquent que la décision prise par un classifieur ne contribue qu'en partie à la décision finale du système. C'est par exemple le cas en reconnaissance de l'écriture manuscrite, lorsqu'un classifieur entraîné à distinguer des lettres est ensuite utilisé au sein d'un système dédié à la reconnaissance de mots. Dans ce type de contexte, il est préférable de disposer d'une mesure de confiance dans la décision prise par le classifieur. Il peut alors être intéressant de chercher à estimer les probabilités *a posteriori* d'appartenance à chacune des classes. Or, comme nous l'avons vu dans le premier chapitre, dans sa formulation originale un SVM ne permet pas d'estimer de telles probabilités. Ainsi, dans ce chapitre, nous présenterons différentes méthodes de post-traitement permettant d'estimer des probabilités *a posteriori* à partir des sorties du ou des SVM. Nous nous intéresserons tout d'abord aux cas bi-classes, puis à l'extension aux cas multi-classes. Nous reprendrons alors notre comparaison entre la stratégie « un contre tous » et la stratégie « un contre un » dans ce contexte d'estimation de probabilités.

3.1 Calibrage des sorties du SVM

3.1.1 État de l'art

Un premier type d'approche pour obtenir des probabilités consiste à modifier la formulation originale du SVM. La méthode proposée par Wahba [40] consiste à rajouter en sortie du SVM une sigmoïde de la forme :

$$p(x) = \frac{1}{1+\exp\left(-f(x)\right)} \tag{3.1}$$

et à modifier la fonction minimisée lors de l'apprentissage qui prend alors la forme :

$$-\frac{1}{n}\sum_{i=1}^{n}\left(\frac{1+y_i}{2}\log\left(p(x_i)\right)+\frac{1-y_i}{2}\log\left(1-p(x_i)\right)\right)+\lambda\|w\|^2 \tag{3.2}$$

Cependant, le classifieur qui résulte de la minimisation de l'équation (3.2) n'est plus à proprement parlé un SVM dans le sens où tous les exemples d'apprentissage sont utilisés pour la prise de décision. Par ailleurs, Chu *et al.* [41] ont proposé une technique Bayesienne pour l'apprentissage des SVM qui permet aussi d'obtenir directement une estimation des probabilités *a posteriori* en sortie du SVM, tout en conservant la notion de vecteurs de support. Malheureusement, les résultats reportés par les auteurs sont dans l'ensemble inférieurs à ceux obtenus avec un SVM standard.

Un second type d'approche consiste à calibrer les sorties du SVM lors d'une étape de post-traitement. En effet, la valeur de sortie avant seuillage d'un SVM standard est liée à la distance entre la donnée à classer et la frontière de décision. La valeur absolue de la sortie du SVM peut donc être considérée comme une mesure de confiance dans la décision. Il semble ainsi raisonnable de penser que la probabilité que la donnée x appartienne à l'une ou l'autre des deux classes sont fortement liées à la valeur de sortie $f(x)$ du SVM et donc de poser l'hypothèse que :

$$P\left(y=1\,|\,x\right)\approx P\left(y=1\,|\,f(x)\right) \tag{3.3}$$

Une méthode simple pour estimer les probabilités *a posteriori* consiste alors à utiliser une approche paramétrique pour estimer les densités de probabilité des sorties du SVM :

$$\hat{p}\left(f(x)\,|\,y=1\right) \text{ et } \hat{p}\left(f(x)\,|\,y=-1\right) \tag{3.4}$$

La règle de Bayes peut ensuite être utilisée pour estimer les probabilités *a posteriori* :

$$\hat{P}\left(y=1 \mid f(x)\right)=\frac{\hat{p}\left(f(x) \mid y=1\right)\hat{P}\left(y=1\right)}{\hat{p}\left(f(x) \mid y=1\right)\hat{P}\left(y=1\right)+\hat{p}\left(f(x) \mid y=-1\right)\hat{P}\left(y=-1\right)} \tag{3.5}$$

Si l'on considère que pour chacune des deux classes les densités de probabilité des sorties du SVM suivent des distributions normales :

$$p\left(f(x) \mid y=1\right) \sim N\left(\mu_1,\sigma_1^{\,2}\right) \text{ et } p\left(f(x) \mid y=-1\right) \sim N\left(\mu_2,\sigma_2^{\,2}\right) \tag{3.6}$$

On obtient alors une fonction de calibrage de la forme :

$$\hat{P}\left(y=1 \mid f(x)\right)=\frac{1}{1+\exp\left(A f(x)^2+Bf(x)+C\right)} \tag{3.7}$$

dont les coefficients sont déterminés à partir des estimations des paramètres des distributions normales et des probabilités *a priori* :

$$A=\frac{1}{2\hat{\sigma}_1^{\,2}}-\frac{1}{2\hat{\sigma}_2^{\,2}} \tag{3.8}$$

$$B=\frac{\hat{\mu}_2}{\hat{\sigma}_2^{\,2}}-\frac{\hat{\mu}_1}{\hat{\sigma}_1^{\,2}} \text{ et} \tag{3.9}$$

$$C=\frac{\hat{\mu}_1^{\,2}}{2\hat{\sigma}_1^{\,2}}-\frac{\hat{\mu}_2^{\,2}}{2\hat{\sigma}_2^{\,2}}+\ln\left(\frac{\hat{\sigma}_1}{\hat{\sigma}_2}\right)+\ln\left(\frac{\hat{P}(y=-1)}{\hat{P}(y=1)}\right) \tag{3.10}$$

Cependant, la présence du terme quadratique dans l'exponentielle de l'équation (3.7) a pour effet de rendre cette fonction non monotone, ce qui s'avère problématique. En effet, lorsque les variances des deux distributions sont différentes, le coefficient A est différent de zéro et alors, soit la probabilité *a posteriori* d'appartenir à la classe positive chute à zéro lorsque la sortie du SVM atteint de fortes valeurs positives (figure 17-a), soit elle remonte à un pour de fortes valeurs négatives (figure 17-b).

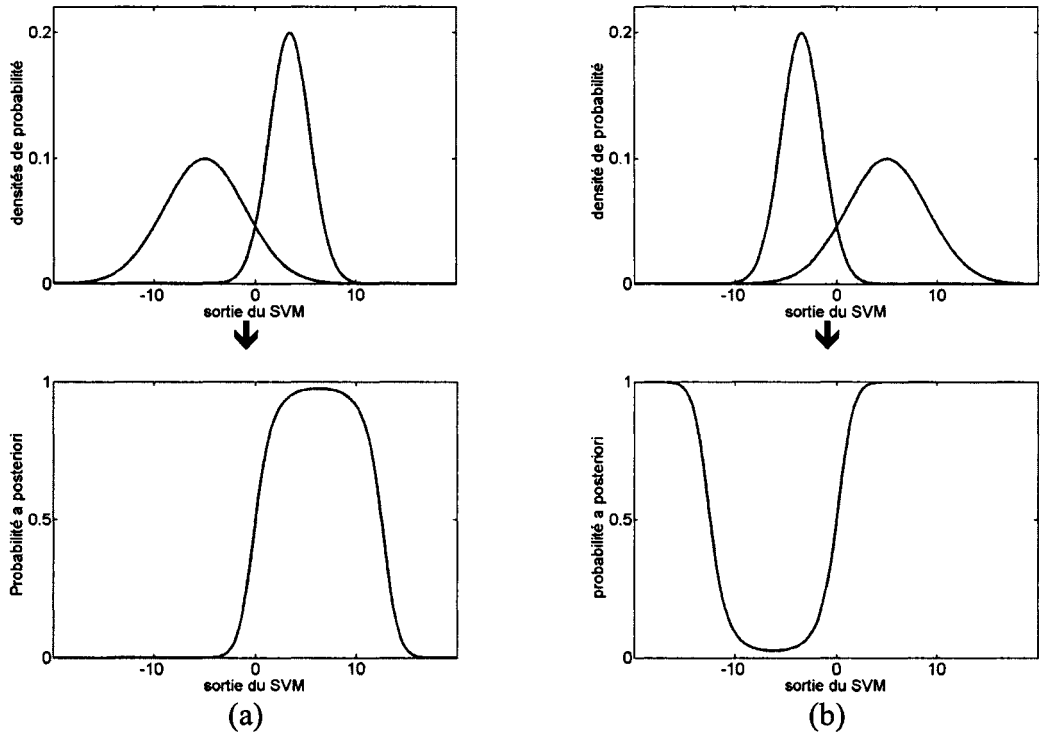


Figure 17 Problème lié à l'utilisation de densités de probabilité normales

Pour contourner ce problème, Hastie et Tibshirani [16] proposent d'utiliser la même valeur pour les deux variances, à savoir la moyenne arithmétique des variances. Ainsi, le coefficient A de l'équation (3.7) est égal à zéro et le terme quadratique disparaît, ce qui résout donc le problème de monotonie. De plus, de manière à ne pas déplacer les frontières de décision, ils s'arrangent pour annuler le coefficient C de l'équation (3.7) en

ne tenant pas compte des probabilités *a priori* et en utilisant des moyennes de même valeur absolue, à savoir la moyenne arithmétique des valeurs absolues des deux moyennes. La fonction alors obtenue est une sigmoïde de la forme :

$$\hat{P}\left(y=1 \mid f(x)\right)=\frac{1}{1+\exp\left(A f(x)\right)} \tag{3.11}$$

dont la pente est donc déterminée à partir des estimations des paramètres des distributions normales :

$$A=-2\frac{\hat{\mu}_1-\hat{\mu}_2}{\hat{\sigma}_1^{\,2}+\hat{\sigma}_2^{\,2}} \tag{3.12}$$

Malheureusement, si cette méthode permet de résoudre le problème de monotonie, elle introduit un biais dans l'estimation des probabilités qui risque d'être non négligeable lorsque les variances ont des valeurs très différentes.

Par ailleurs, Platt [42] propose lui aussi d'utiliser une sigmoïde en sortie du SVM. Bien qu'il considère que l'hypothèse de normalité des distributions soit souvent violée, il constate que pour les données situées entre les marges de séparation, les distributions semblent avoir un comportement de type exponentiel. Or, si l'on considère que :

$$p\left(f(x) \mid y=1\right)=\exp\left(A_1 f(x)+B_1\right) \text{ et } p\left(f(x) \mid y=-1\right)=\exp\left(A_2 f(x)+B_2\right) \tag{3.13}$$

l'utilisation de la règle de Bayes conduit effectivement à l'obtention d'une sigmoïde :

$$\hat{P}\left(y=1 \mid f(x)\right)=\frac{1}{1+\exp\left(A f(x)+B\right)} \tag{3.14}$$

Platt propose ainsi d'utiliser un algorithme d'apprentissage pour ajuster les paramètres de la sigmoïde de manière à minimiser l'opposé du logarithme de la vraisemblance :

$$-\sum_{i=1}^{n}\left(\frac{1+y_i}{2}\log\left(\hat{P}\left(y_i=1\mid x_i\right)\right)+\frac{1-y_i}{2}\log\left(1-\hat{P}\left(y_i=1\mid x_i\right)\right)\right) \tag{3.15}$$

Afin de résoudre ce problème d'optimisation, il propose d'utiliser un algorithme de descente de gradient de type Levenberg-Marquardt dont le pseudo-code est fourni en annexe de [42]. Cependant, Lin *et al.* [43] ont constaté des problèmes liés au choix de cet algorithme de minimisation et à son implémentation. Ils proposent donc d'utiliser un autre type d'algorithme de descente de gradient pour optimiser les paramètres de la sigmoïde.

Une dernière approche est proposée par Zadrozny et Elkan [44] qui contestent l'utilisation d'une fonction sigmoïde et proposent d'utiliser une approche non paramétrique nommée « pair-adjacent violators ». Cette méthode consiste à utiliser une fonction de calibrage en escalier. La plage des valeurs de sortie du SVM est alors découpée en intervalles réguliers et la valeur de la fonction de calibrage dans chaque intervalle est déterminée itérativement de manière à minimiser l'erreur quadratique moyenne :

$$\frac{1}{n}\sum_{i=1}^{n}\left(\frac{1+y_i}{2}-\hat{P}(y_i=1\mid x_i)\right)^2 \tag{3.16}$$

D'autre part, quelle que soit la méthode utilisée, il est nécessaire de disposer d'une base de données des sorties du SVM pour fixer les paramètres de la fonction de calibration. Pour obtenir cette base, le plus simple serait d'utiliser les données d'apprentissage. Cependant, Platt [42] nous met en garde contre cette façon de procéder, qui selon lui, peut introduire un biais important dans l'estimation des probabilités. Ainsi, il semble

préférable soit d'utiliser les données d'une base de validation, soit d'utiliser une technique de validation croisée pour obtenir une base non-biaisée.

3.1.2 Résultats expérimentaux

Dans un premier temps, nous avons choisi d'utiliser des données artificielles de manière à vérifier l'hypothèse de l'équation (3.3) qui sous entend que la sortie d'un SVM est une « bonne mesure » de confiance et donc qu'on peut approximer $P(y=1|x)$ en estimant $P(y=1|f(x))$. Ainsi, en utilisant des données artificielles, nous connaîtrons les distributions ayant permis de générer les données d'apprentissage et nous pourrons donc comparer les probabilités réelles et les valeurs estimées à partir du SVM. D'autre part, nous nous placerons dans le cas de problèmes à deux dimensions de manière à pouvoir facilement visualiser les résultats. Par la suite, nous visualiserons, dans le cas de données réelles, les distributions des sorties des SVM et nous vérifierons que les méthodes proposées permettent d'estimer correctement $P(y=1|f(x))$.

3.1.2.1 Données artificielles

Le premier problème utilise les distributions suivantes :

$$p(x|y=1) \sim N\left(\begin{pmatrix}3\\3\end{pmatrix},\begin{pmatrix}0.81 & 0\\0 & 0.81\end{pmatrix}\right) \text{ et} \tag{3.17}$$

$$p(x|y=-1) \sim N\left(\begin{pmatrix}6\\6\end{pmatrix},\begin{pmatrix}0.81 & 0\\0 & 0.81\end{pmatrix}\right) \tag{3.18}$$

où $N(\mu,\Sigma)$ est une distribution normale de moyenne μ et de covariance Σ.

Pour chacune des deux classes, 500 exemples ont été générés pour former une base d'apprentissage qui est utilisée pour entraîner un SVM. Le noyau linéaire est alors utilisé et le paramètre de pénalisation est arbitrairement fixé à 1. La frontière de décision obtenue ainsi que les données d'apprentissage sont représentées figure 18. Les données de la première classe ($y = 1$) sont représentées par des points (•), alors que les données de la seconde classe ($y = -1$) sont représentées par des plus (+).

D'autre part, pour chacune des classes, 50 000 exemples supplémentaires ont été générés de manière à former une base de validation qui est utilisée pour évaluer les distributions réelles des sorties du SVM.

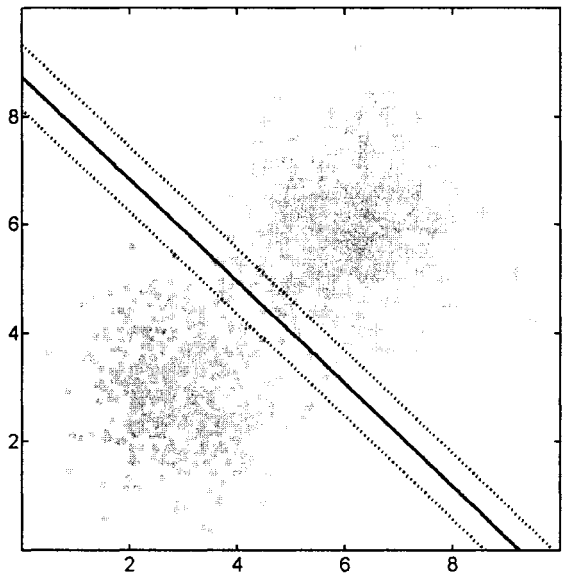
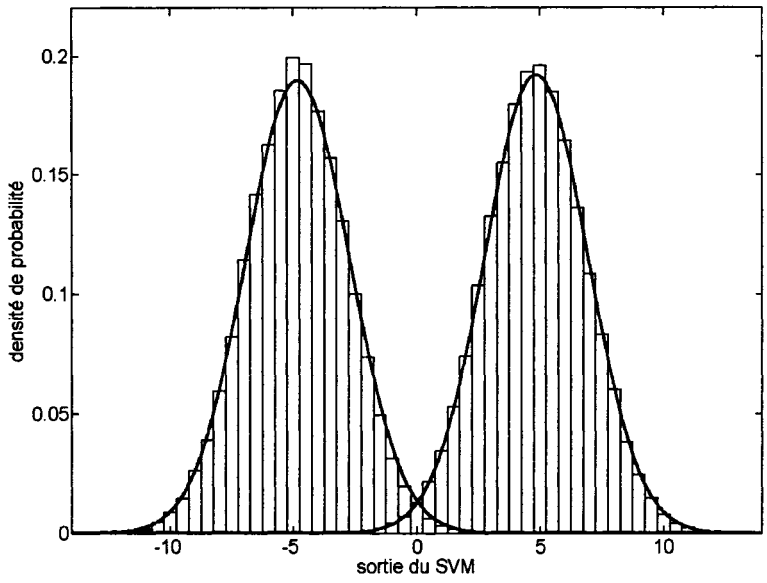
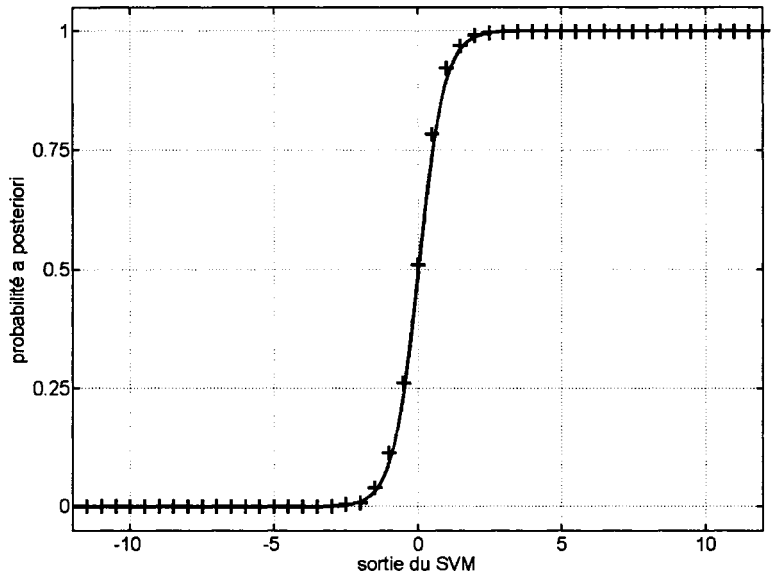


Figure 18 Illustration du premier problème artificiel

Dans un premier temps, nous avons vérifié que pour chacune des deux classes, les sorties du SVM suivent bien une loi normale. Ainsi, comme nous pouvons le voir à la figure 19-a, les histogrammes calculés à partir des données de validation coïncident avec les Gaussiennes dont les paramètres ont été estimés à partir des données d'apprentissage.



(a)

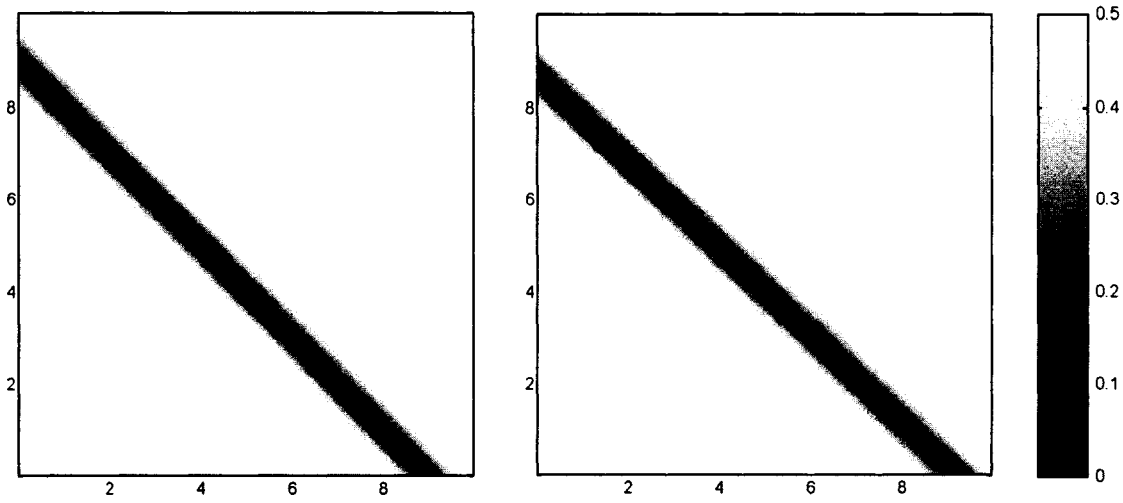


(b)

Figure 19 Estimation des probabilités dans le cas du premier problème artificiel

Ensuite, nous avons vérifié que la fonction de calibrage de l'équation (3.7) permet bien de transformer les sorties du SVM en probabilités *a posteriori*. Or, comme nous pouvons le voir figure 19-b, la fonction dont les coefficients ont été estimés à partir des données d'apprentissage coïncide avec les probabilités *a posteriori* représentées par des plus (+) qui ont été obtenues en appliquant la règle de Bayes aux valeurs des histogrammes représentés figure 19-a. Ainsi, dans le cas de ce premier problème très simple, il semble tout à fait approprié d'utiliser une méthode paramétrique pour définir une fonction de calibrage qui permettra de convertir les sorties du SVM en probabilités *a posteriori*.

Enfin, nous pouvons constater à la figure 20 que les probabilités estimées à partir des sorties du SVM, représentées à la figure 20-b, sont proches des probabilités réelles, représentées figure 20-a. L'hypothèse de l'équation (3.3) semble donc vérifiée dans ce cas de figure. Notons par ailleurs, que pour des raisons pratiques, plutôt que de représenter les valeurs des probabilités de l'une ou l'autre des deux classes, nous avons choisi de représenter les valeurs de $|0,5-p|$ où p représente soit la probabilité réelle d'appartenir à l'une des classes dans le cas de la figure 20-a, soit celle estimée à partir des sorties du SVM dans le cas de la figure 20-b. Ainsi, selon le code de couleur qui est représenté figure 20-c, plus les pixels sont clairs plus les valeurs des probabilités sont proches de zéro ou de un.



(a) (b) (c)

Figure 20 Comparaison des probabilités dans le cas du premier problème artificiel

Dans un second temps, nous avons repris le problème artificiel utilisé dans le premier chapitre pour étudier l'effet des hyper-paramètres. Les distributions utilisées sont donc celles définis par les équations (1.29) et (1.30). Pour chacune des deux classes, 5 000 exemples ont été générés pour former une base d'apprentissage qui est utilisée pour entraîner un SVM. Le noyau Gaussien est alors utilisé et les deux hyper-paramètres sont fixés à 10^{-1}. La frontière de décision obtenue ainsi que les données d'apprentissage sont représentées à la figure 21. D'autre part, pour chacune des classes, 50 000 exemples supplémentaires ont été générés de manière à former une base de validation qui est utilisée pour évaluer les distributions réelles des sorties du SVM.

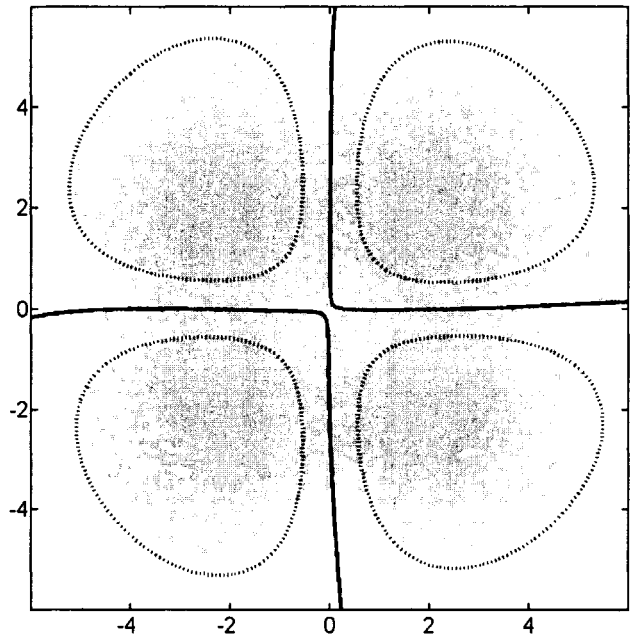
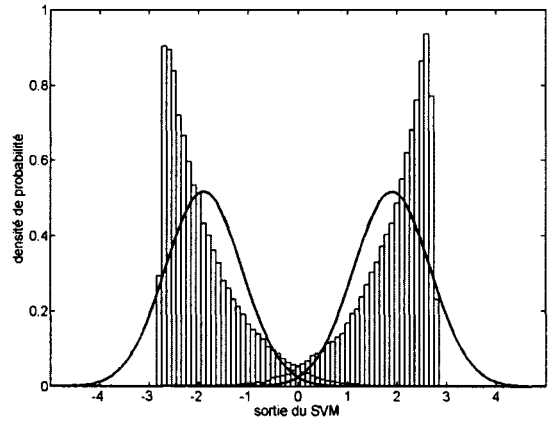
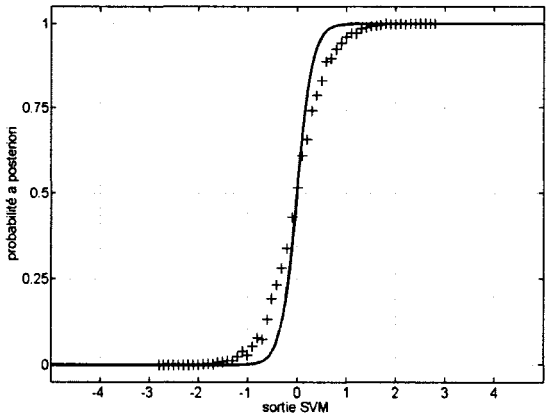


Figure 21 Second problème artificiel utilisé pour vérifier l'hypothèse de départ

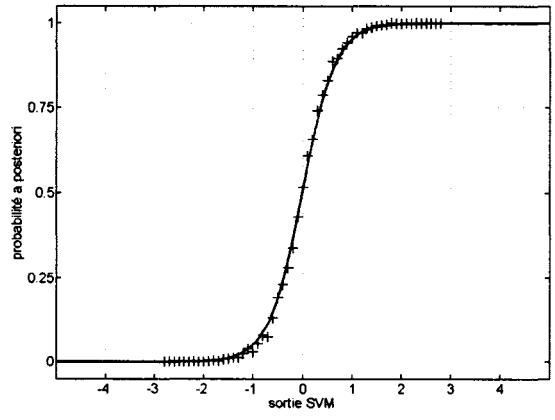
Nous pouvons constater à la figure 22-a que dans ce cas, les distributions des sorties du SVM ne suivent pas des lois normales. En effet, les histogrammes calculés à partir des données de validation ne coïncident pas avec les Gaussiennes dont les paramètres ont été estimés à partir des données d'apprentissage. Nous pouvons alors constater à la figure 22-b que la fonction de calibrage de l'équation (3.7) induit un biais important dans l'estimation des probabilités *a posteriori*. Par contre, nous pouvons constater à la figure 22-c que la sigmoïde, dont les paramètres ont été obtenus en appliquant l'algorithme de Lin *et al.* [43], permet bien de convertir les sorties du SVM en probabilités *a posteriori*.



(a)



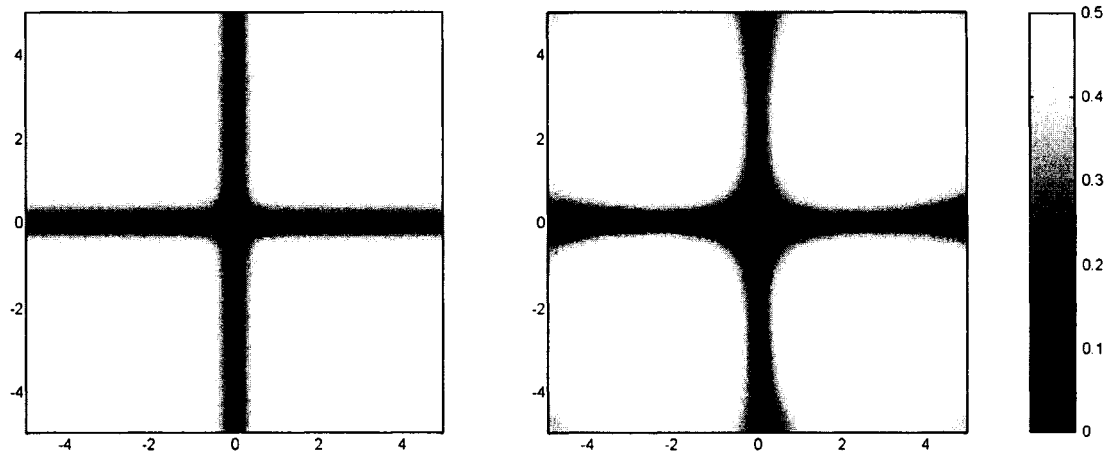
(b)



(c)

Figure 22 Estimation des probabilités dans le cas du second problème artificiel

Finalement, nous pouvons constater que les probabilités estimées à partir des sorties du SVM représentées à la figure 23-b ne correspondent pas parfaitement aux probabilités réelles représentées à la figure 23-a. Cependant, les probabilités estimées sont tout de même relativement proche des probabilités réelles. L'hypothèse de l'équation (3.3) semble donc acceptable.



(a) (b) (c)

Figure 23 Comparaison des probabilités dans le cas du second problème artificiel

3.1.2.2 Données réelles

Nous avons ensuite visualisé les distributions des valeurs de sorties des SVM entraînés sur les données réelles du chapitre 2. Première constatation, dans la majorité des cas, les distributions semblent suivre une loi normale. Ainsi, dans le cas du SVM entraîné à séparer les exemples du chiffre 1 des exemples du chiffre 7, nous pouvons constater à la figure 24-a que les histogrammes des sorties coïncident parfaitement avec les distributions normales estimées à partir de ces même valeurs. De plus, dans ce cas de figure, les deux distributions sont relativement symétriques, nous pouvons alors constater à la figure 24-b que de la fonction de calibrage (3.7) qui résulte de l'utilisation de la règle de Bayes, permet d'estimer correctement les probabilités *a posteriori*.

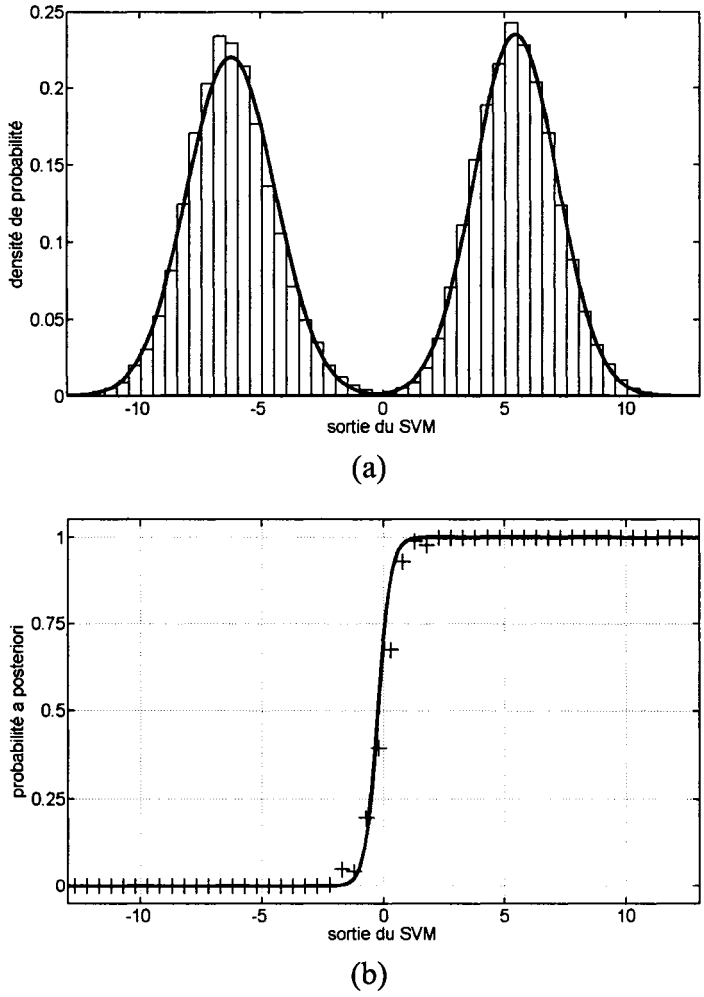
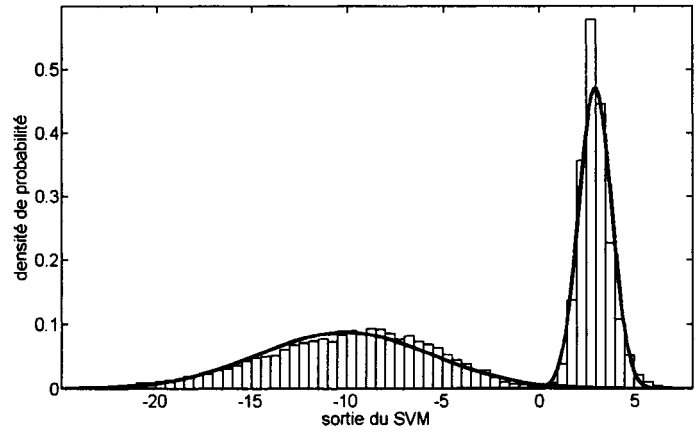
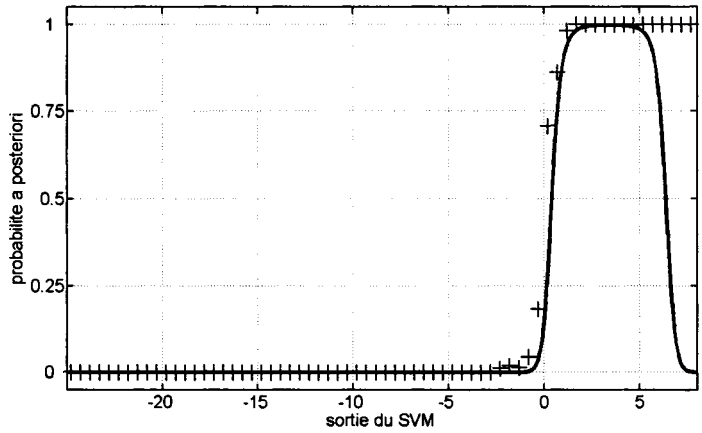


Figure 24 Calibrage des sorties du SVM « 1 contre 7 »

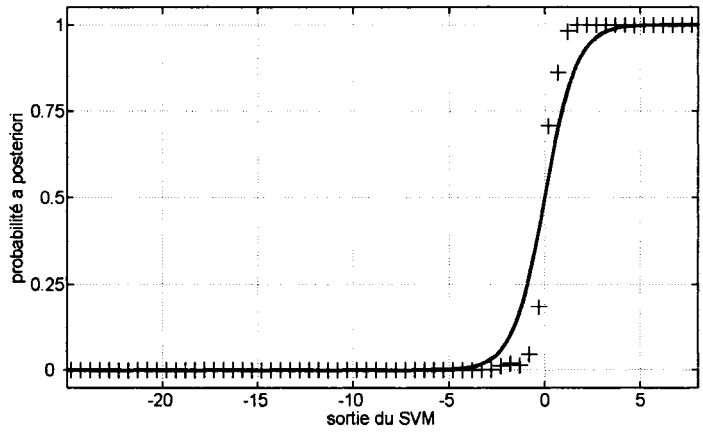
Deuxième constatation, les deux distributions peuvent avoir des écarts types très différents. C'est par exemple le cas pour les sorties du SVM entraîné à séparer les exemples du chiffre 1 des exemples du chiffre 8, dont les distributions sont présentées à la figure 25-a. Nous pouvons alors effectivement constater à la figure 25-b que la fonction de calibrage (3.7) qui résulte de l'utilisation de la règle de Bayes n'est pas monotone. Nous pouvons aussi constater à la figure 25-c que la solution proposée par Hastie et Tibshirani [16] pour obtenir une fonction de calibrage monotone introduit un biais non négligeable dans l'estimation des probabilités *a posteriori*.



(a)



(b)



(c)

Figure 25 Calibrage des sorties du SVM « 1 contre 8 »

Par ailleurs, la solution proposée par Platt [42] semble particulièrement bien adaptée. En effet, l'hypothèse du comportement exponentiel des distributions des sorties correspondant aux données proches de la frontière semble vraisemblable. L'utilisation d'une fonction sigmoïde pour calibrer les sorties du SVM apparaît donc être une solution simple et efficace. Cependant, tout comme par Lin *et al.* [43] nous avons constaté un problème avec l'algorithme d'optimisation proposé par Platt [42], mais l'algorithme proposé par Lin *et al.* [43] fonctionne parfaitement et permet de bien estimer les probabilités *a posteriori*. Ainsi, si l'on reprend l'exemple de la figure 25, nous pouvons constater que la sigmoïde obtenue avec l'algorithme de Platt [42] n'épouse pas du tout les probabilités *a posteriori* (voir figure 26), alors que celle obtenue avec l'algorithme de Lin *et al.* [43] est parfaitement adaptée (voir figure 27).

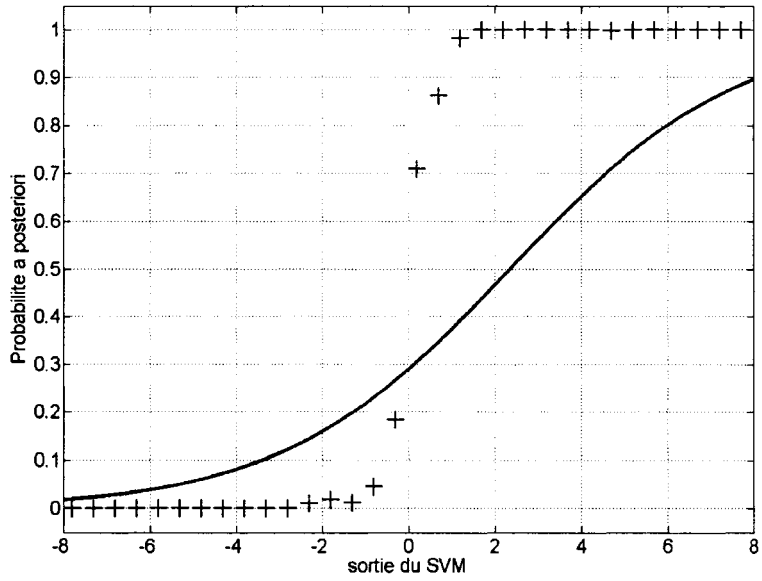


Figure 26 Résultat de l'algorithme de Platt [42] pour le SVM « 1 contre 8 »

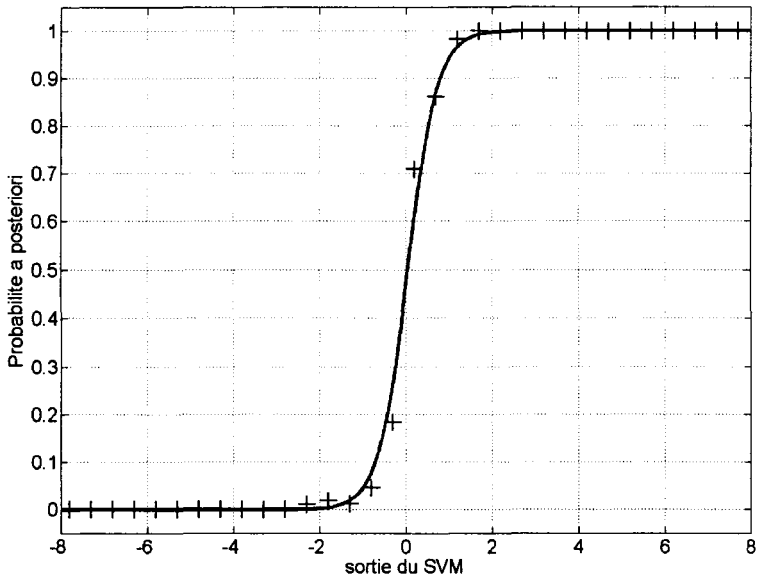


Figure 27 Résultat de l'algorithme de Lin *et al.* [43] pour le SVM « 1 contre 8 »

Notons pour finir, que dans notre cas, les probabilités *a priori* d'appartenance aux différentes classes sont égales. Or, les bases de validation ne contiennent pas le même nombre d'exemples pour chaque classe. Nous avons donc choisi d'utiliser la technique de validation croisée pour constituer les bases de données de sorties des SVM qui seront utilisées pour estimer les paramètres des distributions ou pour optimiser les paramètres des fonctions sigmoïdes. Chaque base d'apprentissage a alors été coupée en quatre ensembles et quatre SVM ont été entraînés en faisant permuter les trois ensembles utilisés pour l'entraînement et l'ensemble utilisé pour évaluer les sorties. Finalement, les quatre ensembles sont réunis pour former la base de données de sorties de SVM. D'autre part, notons que les quatre SVM utilisés pour constituer la base de données de sorties ne sont ensuite plus utilisés, mais c'est bien entendu le SVM entraîné avec toutes les données d'apprentissage qui sera utilisé.

3.1.3 Conclusion

Après avoir visualisé les distributions des sorties de tous les SVM issues des stratégies « un contre un » et « un contre tous », nous pouvons conclure que l'ajout d'une sigmoïde dont les paramètres sont soigneusement ajustés par apprentissage, semble être une solution efficace pour estimer des probabilités *a posteriori* à partir des sorties d'un SVM. Cependant, cette solution ne permet que de traiter des problèmes à deux classes. Dans la suite de ce chapitre, nous nous intéresserons donc à l'extension de cette méthode aux problèmes multi-classes.

3.2 Extension aux problèmes multi-classes

3.2.1 Stratégie « un contre tous »

Chaque SVM de la stratégie « un contre tous » étant entraîné à séparer les données d'une classe de celles de toutes les autres classes, chaque sortie est directement liée à la probabilité d'appartenance à l'une des classes. Le problème consiste donc à calibrer les sorties de l'ensemble des SVM.

3.2.1.1 Description des méthodes

La solution la plus intuitive pour estimer les probabilités *a posteriori* d'appartenance à chacune des classes ω_i consiste donc à calibrer séparément les sorties $f_i(x)$ de chaque SVM. Ainsi, étant donnée les résultats de la section précédente, la solution la plus adaptée serait d'ajouter une sigmoïde :

$$\hat{P}\left(\omega_i \mid f_i(x)\right) = \frac{1}{1+\exp\left(A_i f_i(x) + B_i\right)} \tag{3.19}$$

dont les paramètres serait optimisés par apprentissage en utilisant l'algorithme proposé par Lin *et al.* [43]. Cependant, rien ne garantit alors que :

$$\sum_{i=1}^{c} \hat{P}\left(\omega_i \mid f_i(x)\right) = 1 \tag{3.20}$$

En pratique, nous avons constaté que pour une très grande majorité des exemples, la somme des probabilités est très proche de un. Cependant, il arrive parfois que celle-ci puisse être largement supérieure ou inférieure à un. Une solution très simple pour éviter ce problème consiste alors à diviser chaque probabilité par la somme de toutes les probabilités :

$$\hat{P}\left(\omega_i \mid x\right) = \frac{\hat{P}\left(\omega_i \mid f_i(x)\right)}{\sum_{j=1}^{c} \hat{P}\left(\omega_j \mid f_j(x)\right)} \tag{3.21}$$

Toutefois, si nous reprenons l'hypothèse qui a conduit Platt [42] à l'utilisation d'une sigmoïde, c'est à dire que les distributions des sorties correspondant aux données proches de la frontière ont un comportement exponentiel, l'utilisation de la règle de Bayes nous conduit à l'obtention d'une fonction paramétrique de la forme :

$$\hat{P}(\omega_i \mid x) = \frac{\exp\left(A_i f_i\left(x\right) + B_i\right)}{\sum_{j=1}^{c} \exp\left(A_j f_j\left(x\right) + B_j\right)} \tag{3.22}$$

Cette fonction, qui peut donc être vue comme une généralisation de la fonction sigmoïde pour des problèmes multi-classes, se nomme softmax. Elle a été introduite sous une forme simplifiée par Bridle [45]. L'objectif était alors d'estimer des probabilités en sortie d'un réseau de neurones. Les paramètres de la fonction étaient donc fixes ($A_i = 1$

et $B_i = 0$) et ce sont les poids du réseau qui étaient ajustés. Dans notre cas, ce sont, bien entendu, les paramètres de la fonction softmax qui seront ajustés. Nous procéderons alors de la même manière que dans le cas de la sigmoïde, c'est-à-dire en minimisant la NLL, qui dans le cas multi-classes prend la forme suivante :

$$-\sum_{i=1}^{n} \log\left(\hat{P}\left(\omega_{\ell(i)} \mid x_i\right)\right) \tag{3.23}$$

où $\ell(i)$ représente le label associé à l'exemple x_i.

Par ailleurs, notons que l'ajout de n'importe quelle constante à l'ensemble des B_i ne modifie pas les valeurs de sortie de la fonction (3.22). Nous n'avons donc pas un seul minimum global, mais une infinité. Ainsi, de manière à faciliter l'optimisation de ces paramètres, nous proposons de fixer l'un des B_i arbitrairement, par exemple : $B_1 = 0$. D'autre part, notons que l'utilisation de ce type de fonction pour estimer des probabilités *a posteriori* à partir des sorties des SVM de la stratégie « un contre tous » a aussi été proposée indépendamment par Duan *et al.* [46].

3.2.1.2 Résultats expérimentaux

De manière à comparer les différentes approches en termes de précision, la solution la plus simple consiste bien évidemment à évaluer le taux d'erreur sur la base de test. Cependant, il est possible que cette mesure ne s'avère pas assez précise pour différentier des techniques très proches. Ainsi, pour comparer la qualité de différentes estimations de probabilité, il est commun d'utiliser une fonction d'erreur. Nous proposons alors d'utiliser comme second critère de comparaison, la NLL évaluée sur la base de test.

Bien que ce type de mesure soit plus précise qu'un taux d'erreur, elle présente le défaut d'être peu concrète. Nous proposons donc un nouveau critère basé sur le principe démontré par Chow [47] qui stipule que lorsque les densités de probabilité sont parfaitement connues, la règle optimale de rejet consiste alors à rejeter un exemple x si :

$$\max_{i=1,\ldots,c}\left(P(\omega_i \mid x)\right) < T \tag{3.24}$$

Le taux d'exemples rejetés et par conséquent le taux d'erreur sont donc directement liés au seuil T de l'équation (3.24). Ainsi, en faisant varier ce seuil, on obtient une description complète du compromis erreur-rejet (voir l'exemple de la figure 28).

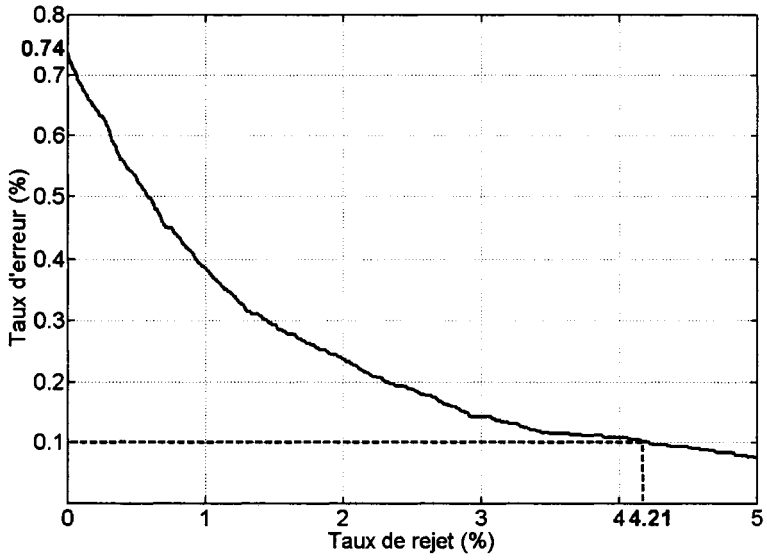


Figure 28 Compromis erreur-rejet du MLP sur la base de chiffres

Toutefois, dans le cas d'applications réelles, les densités de probabilité ne sont, bien entendu, pas connues. Il n'est alors possible que d'estimer avec plus ou moins de précision les probabilités *a posteriori*. Nous pouvons donc en conclure que meilleur sera l'estimation des probabilités, meilleur sera le compromis erreur-rejet. De ce fait, nous

proposons d'utiliser comme mesure de la qualité des probabilités estimées, le taux de rejet nécessaire pour faire décroître le taux d'erreur jusqu'à une valeur donnée. Nous choisirons de fixer cette valeur à 0,1% qui est une valeur de rejet couramment utilisée en reconnaissance de caractères manuscrits.

Dans la section précédente, nous avons pu constater en visualisant les distributions des sorties des SVM, que l'algorithme proposé par Platt [42] pour optimiser les paramètres de la sigmoïde ne fonctionne pas toujours correctement, tandis que celui proposé par Lin *et al.* [43] semble opérationnel. Nous avons donc voulu vérifier l'impact sur les probabilités estimées. Les résultats obtenus sur la base de chiffres sont reportés au tableau XI. Nous pouvons alors constater que l'algorithme proposé par Platt [42] introduit effectivement un biais important dans l'estimation des probabilités.

Tableau XI

Impact de l'algorithme utilisé pour l'optimisation des paramètres des sigmoïdes sur les résultats obtenus avec la stratégie « un contre tous » sur la base de chiffres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
Platt [42]	0,72	5,08	3 694
Lin *et al.* [43]	**0,64**	**3,86**	**1 893**

Dans un second temps, nous avons évalué l'impact de la fonction utilisée pour calibrer les sorties des SVM. Nous avons alors comparé les trois fonctions de calibrage présentées dans la section précédente. Les deux premières fonctions sont basées sur l'hypothèse que les distributions des sorties des SVM suivent des lois normales. La première (Équation (3.7)) résulte de l'utilisation de la règle de Bayes, tandis que la seconde (Équation (3.11)) est une sigmoïde dont la pente est calculée à partir de la

moyenne et de l'écart type de chacune des deux distributions. La troisième fonction (Équation (3.14)) est la sigmoïde dont les paramètres sont optimisés à l'aide de l'algorithme proposé par Lin *et al.* [43]. Les résultats obtenus sur la base de chiffres sont reportés au tableau XII. Comme le laissait envisager la section précédente, nous pouvons alors constater que les deux premières fonctions de calibrage introduisent un biais significatif dans l'estimation des probabilités. L'utilisation de l'algorithme proposé par Lin *et al.* [43] s'avère donc bien être la meilleure façon de calibrer les sorties d'un SVM.

Tableau XII

Impact de la fonction utilisée pour calibrer les sorties des SVM sur les résultats obtenus avec la stratégie « un contre tous » sur la base de chiffres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
Équation (3.7)	0,64	5,00	3 011
Équation (3.11)	0,66	11,37	2 272
Équation (3.14)	**0,64**	**3,86**	**1 893**

Nous avons ensuite évalué l'impact de la normalisation des probabilités de manière à garantir que la somme soit égale à un. Les résultats obtenus sur la base de chiffres sont reportés au tableau XIII. Nous pouvons alors constater que l'utilisation de l'équation (3.21) permet non seulement de garantir que la somme des probabilités soit égale à un, mais aussi d'améliorer légèrement la qualité des probabilités estimées.

Tableau XIII

Impact de la normalisation des sortie des sigmoïdes sur les résultats obtenus avec la stratégie « un contre tous » sur la base de chiffres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
Sans normalisation	0,64	3,86	1 893
Avec normalisation	**0,64**	**3,73**	**1 517**

Pour finir, nous avons comparé les résultats obtenus avec les sigmoïdes après normalisation, à ceux obtenus en utilisant la fonction softmax. Les résultats sont reportés aux tableaux XIV et XV.

Tableau XIV

Résultats obtenus avec la stratégie « un contre tous » sur la base de chiffres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
Sigmoïde	0,64	3,73	1 517
Softmax	**0,63**	**2,30**	**1 310**

Tableau XV

Résultats obtenus avec la stratégie « un contre tous » sur la base de lettres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
Sigmoïde	3,17	36,26	1 570
Softmax	**3,18**	**25,52**	**1 375**

3.2.1.3 Conclusion

Nous pouvons donc en conclure que l'utilisation de la fonction softmax semble être la solution la mieux adaptée pour estimer les probabilités *a posteriori* à partir des sorties des SVM de la stratégie « un contre tous ».

3.2.2 Stratégie « un contre un »

Dans le cas de la stratégie « un contre un », le problème est quelque peu différent. En effet, si l'on utilise des sigmoïdes pour calibrer les sorties des différents SVM, les probabilités qui sont alors estimées de manière locale supposent que la donnée appartient à l'une des deux classes correspondant au SVM. Bien entendu, dans la majorité des cas, ce n'est pas le cas et les probabilités ainsi estimées ne sont donc pas significatives. La difficulté est alors de combiner ces probabilités « locales », que nous noterons $\hat{P}\left(\omega_i \mid f_{i,j}(x)\right)$, de manière à estimer les probabilités « globales » $\hat{P}(\omega_i \mid x)$.

3.2.2.1 Description des méthodes

De nombreuses méthodes permettant de réaliser cette tâche sont proposées dans la littérature [15, 16, 18, 19, 21]. Nous avons choisi de comparer trois d'entre elles.

La première méthode a été proposée par Price *et al.* [19], qui partent de l'hypothèse que pour toutes les classes ω_i :

$$\sum_{j=1, j\neq i}^{c} P(\omega_{i,j} \mid x) - (c-2)P(\omega_i \mid x) = 1 \tag{3.25}$$

où $\omega_{i,j}$ représente l'union des classes ω_i et ω_j.

Alors, en considérant que :

$$\hat{P}(\omega_i \mid f_{i,j}(x)) \approx \frac{P(\omega_i \mid x)}{P(\omega_{i,j} \mid x)} \tag{3.26}$$

ils obtiennent l'expression suivante :

$$\hat{P}(\omega_i \mid x) = \frac{1}{\sum_{j=1, j\neq i}^{c} \frac{1}{\hat{P}\left(\omega_i \mid f_{i,j}(x)\right)} - \left(c-2\right)} \tag{3.27}$$

Cependant, rien ne garantit que la somme des probabilités soit égale à un. Il semble donc préférable de diviser chaque probabilité par la somme de toutes les probabilités.

La seconde méthode a été proposée par Hastie et Tibshirani [16]. Leur approche est très différente. Elle consiste à utiliser un algorithme itératif pour déterminer les probabilités $p_i = \hat{P}(\omega_i \mid x)$, qui minimise la distance de Kullback-Leibler entre $r_{ij} = \hat{P}\left(\omega_i \mid f_{i,j}(x)\right)$ et $\mu_{ij} = \frac{p_i}{p_i + p_j}$:

$$\sum_{i<j} n_{ij} \left(r_{ij} \log \frac{r_{ij}}{\mu_{ij}} + (1 - r_{ij}) \log \frac{1 - r_{ij}}{1 - \mu_{ij}} \right) \tag{3.28}$$

où n_{ij} représente le nombre d'exemples d'apprentissage appartenant soit à la classe ω_i soit à la classe ω_j.

Pour ce faire, les auteurs proposent d'initialiser les probabilités de la façon suivante :

$$p_i = \frac{2}{c(c-1)} \sum_{j=1, j \neq i}^{c} r_{ij} \tag{3.29}$$

puis répètent $(i = 1,2,\ldots,c,1,\ldots)$ jusqu'à convergence :

1. $\mu_{ij} \leftarrow \frac{p_i}{p_i + p_j}, \ \forall j \neq i$
2. $p_i \leftarrow p_i \cdot \frac{\sum_{j \neq i} n_{ij} r_{ij}}{\sum_{j \neq i} n_{ij} \mu_{ij}}$
3. $p_i \leftarrow \frac{p_i}{\sum_{j=1}^{c} p_j}$

Par ailleurs, ils ne précisent pas quel critère de convergence utiliser. Nous avons donc proposé notre propre critère :

$$\sum_{i=1}^{c}\left(p_i(t)-p_i(t-1)\right)^2 \leq \varepsilon \tag{3.30}$$

où $p_i(t)$ représente les valeurs actuelles des probabilités p_i et $p_i(t-1)$ les valeurs précédentes de ces probabilités.

En pratique, il sera donc nécessaire de fixer minutieusement la valeur du seuil ε. En effet, si la valeur utilisée est trop grande, l'algorithme ne convergera pas et les probabilités seront alors fortement biaisées. Par contre, si la valeur utilisée est trop petite, le nombre d'itérations va augmenter très fortement, sans que la qualité des probabilités ne soit améliorée de manière significative. Cependant, dans le cadre de nos expériences, nous nous sommes uniquement intéressé à la qualité des probabilités, nous avons donc volontairement choisi d'utiliser une valeur vraiment très faible ($\varepsilon = 10^{-12}$), de manière à garantir la convergence de l'algorithme et ce au prix d'un nombre très important d'itérations.

La troisième méthode a été proposée par Hamamura *et al.* [15]. Partant de l'hypothèse que les sorties des différents classifieurs sont indépendantes les unes des autres, ils obtiennent l'expression suivante :

$$\hat{P}(\omega_i \mid x) = \left(\prod_{j=1, j\neq i}^{c} \hat{P}\left(\omega_i \mid f_{i,j}(x)\right) \cdot \frac{\hat{P}(\omega_i)+\hat{P}(\omega_j)}{\hat{P}(\omega_i)} \right) \cdot \hat{P}(\omega_i) \cdot C \tag{3.31}$$

où C est calculé pour satisfaire $\sum_{i=1}^{c} \hat{P}\left(\omega_i \mid x\right) = 1$.

3.2.2.2 Résultats expérimentaux

Les résultats obtenus avec les trois méthodes sont reportés aux tableaux XVI et XVII.

Tableau XVI

Résultats obtenus avec la stratégie « un contre un » sur la base de chiffres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
Price *et al.* [19]	0,70	3,49	1 483
Hastie et Tibshirani [16]	0,69	3,37	1 604
Hamamura *et al.* [15]	0,70	4,31	1 825

Tableau XVII

Résultats obtenus avec la stratégie « un contre un » sur la base de lettres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
Price *et al.* [19]	3,22	25,97	1 421
Hastie et Tibshirani [16]	3,37	25,99	1 548
Hamamura *et al.* [15]	3,29	35,80	2 197

Première constatation, les résultats en termes de taux d'erreur ne sont pas vraiment meilleurs que ceux obtenus au chapitre 2 avec une simple règle de vote majoritaire (0,71% pour les chiffres et 3,29% pour les lettres).

Deuxième constatation, la troisième méthode s'avère moins précise que les deux autres méthodes. En effet, si la différence en termes de taux d'erreur n'apparaît pas significative, la différence en termes de taux de rejet et de NLL est relativement importante.

Troisième constatation, les résultats obtenus avec les deux premières méthodes sont relativement proches. Cependant la première méthode semble légèrement plus précise. Nous pouvons notamment remarquer que la NLL est légèrement plus faible avec la première méthode et le taux d'erreur sur la base de lettres est significativement plus élevé avec la seconde méthode.

3.2.2.3 Conclusion

Finalement, la méthode proposée par Price *et al.* [19] semble être, dans notre cas, la plus adaptée pour combiner les probabilités « locales » issues de la stratégie « un contre un ». En effet, les résultats obtenus avec cette méthode sont nettement meilleurs que ceux obtenus avec la méthode proposée par Hamamura *et al.* [15] et très légèrement meilleurs que ceux obtenus avec la méthode proposée par Hastie et Tibshirani [16], qui par ailleurs, a pour inconvénient d'être itérative.

3.2.3 Comparaison des deux stratégies

Dans cette section, nous proposons de reprendre la comparaison des deux stratégies qui a été commencée dans le chapitre précédent. La stratégie « un contre tous » semblait alors plus précise que la stratégie « un contre un ». Cependant, les règles de décision utilisées étaient très simples et il était donc difficile de conclure à la supériorité d'une des deux stratégies. Nous avions alors conclu qu'il était donc préférable de comparer les deux stratégies en utilisant des méthodes d'estimation de probabilités plus élaborées. Nous avons donc reportés aux tableaux XVIII et XIX les meilleurs résultats, en termes d'estimation de probabilités, obtenus avec chacune des deux stratégies.

Tableau XVIII

Comparaison des deux stratégies sur la base de chiffres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
SVM-OAO	0,70	3,49	1 483
SVM-OAA	**0,63**	**2,30**	**1 310**

Tableau XIX

Comparaison des deux stratégies sur la base de lettres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
SVM-OAO	3,22	25,97	1 421
SVM-OAA	**3,18**	**25,52**	**1 375**

Finalement, si la différence entre les deux stratégies s'avère très faible sur la base de lettres, elle est significative sur la base de chiffres. Nous pouvons notamment remarquer que la stratégie « un contre tous » permet de rejeter 34% de chiffres en moins que la stratégie « un contre un » pour atteindre le même taux d'erreur de 0,1%.

3.2.4 Combinaison des deux stratégies

Le principal problème lors de l'utilisation de la stratégie « un contre un » est que les probabilités $\hat{P}\left(\omega_i \mid f_{i,j}(x)\right)$ qui sont estimées à partir des sorties du SVM entraîné à séparer les données de la classe ω_i de celles de la classe ω_j, ne sont pas pertinentes

lorsque l'exemple x n'appartient ni à la classe ω_i ni à la classe ω_j. Une solution envisageable serait alors d'utiliser les probabilités issues de la stratégie « un contre tous » de manière à atténuer l'effet de ces probabilités problématiques.

3.2.4.1 Description de la méthode

Ainsi, afin d'atténuer l'effet de ces fameuses probabilités $\hat{P}\left(\omega_i \mid f_{i,j}(x)\right)$, Moreira et Mayoraz [18] proposent de les multiplier par les probabilités $\hat{P}(\omega_{i,j} \mid x)$. Ils modifient alors la formulation de l'équation (3.29) qui devient :

$$\hat{P}(\omega_i \mid x) = \frac{2}{c(c-1)} \sum_{j=1, j\neq i}^{c} \hat{P}\left(\omega_i \mid f_{i,j}(x)\right) \cdot \hat{P}\left(\omega_{i,j} \mid x\right) \tag{3.32}$$

Cependant, si la constante $\frac{2}{c(c-1)}$ permet dans l'équation (3.29) de garantir que la somme des probabilités soit égale à un, car :

$$\sum_{i=1}^{c} \sum_{j=1, j\neq i}^{c} \hat{P}\left(\omega_i \mid f_{i,j}(x)\right) = \frac{c(c-1)}{2} \tag{3.33}$$

ce n'est plus du tout le cas dans l'équation (3.32), car :

$$\sum_{j=1, j\neq i}^{c} \hat{P}\left(\omega_i \mid f_{i,j}(x)\right) \cdot \hat{P}\left(\omega_{i,j} \mid x\right) \neq \frac{c(c-1)}{2} \tag{3.34}$$

Nous avons donc reformulé l'équation proposée par Moreira et Mayoraz [18] de la manière suivante :

$$\hat{P}(\omega_i \mid x) = \frac{\sum_{j=1, j\neq i}^{c} \hat{P}\left(\omega_i \mid f_{i,j}(x)\right) \cdot \hat{P}\left(\omega_{i,j} \mid x\right)}{\sum_{k=1}^{c} \sum_{j=1, j\neq k}^{c} \hat{P}\left(\omega_k \mid f_{k,j}(x)\right) \cdot \hat{P}\left(\omega_{k,j} \mid x\right)} \tag{3.35}$$

Finalement, les probabilités $\hat{P}(\omega_{i,j} \mid x)$ seront obtenues en additionnant les probabilités estimées avec la stratégie « un contre tous » :

$$\hat{P}(\omega_{i,j} \mid x) = \hat{P}(\omega_i \mid x) + \hat{P}(\omega_j \mid x) \tag{3.36}$$

La formulation de l'équation (3.35) est donc bien une combinaison des stratégies « un contre un » et « un contre tous ». En effet, les probabilités $\hat{P}\left(\omega_i \mid f_{i,j}(x)\right)$ sont alors estimées en calibrant les sorties des SVM issus de la stratégie « un contre un » à l'aide de sigmoïdes, tandis que les probabilités $\hat{P}(\omega_{i,j} \mid x)$ sont estimées à partir des sorties des SVM issus de la stratégie « un contre tous » à l'aide de la fonction softmax.

3.2.4.2 Résultats expérimentaux

Les résultats obtenus en combinant les deux stratégies sont reportés aux tableaux XX et XXI. On peut alors constater que sur la base de chiffres, la combinaison des deux stratégies ne s'avère pas significativement plus précise que la stratégie « un contre tous ». En effet, les résultats en termes de taux d'erreur et de taux de rejet sont équivalents et seule la NLL est légèrement plus faible. Cependant, la combinaison des deux stratégies s'avère très légèrement plus précise que la stratégie « un contre tous » sur la base de lettres.

Tableau XX

Résultats obtenus avec la combinaison des deux stratégies sur la base de chiffres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
SVM-OAO	0,70	3,49	1 483
SVM-OAA	0,63	2,30	1 310
Combinaison	**0,64**	**2,34**	**1 241**

Tableau XXI

Résultats obtenus avec la combinaison des deux stratégies sur la base de lettres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
SVM-OAO	3,22	25,97	1 421
SVM-OAA	3,18	25,52	1 375
Combinaison	**3,15**	**23,57**	**1 319**

Par contre, ce léger gain en précision se fait au détriment d'une augmentation significative de la complexité. En effet, comme nous pouvons le constater au tableau XXII, le nombre de vecteurs de support utilisés par la combinaison des deux stratégies est plus élevé que le nombre de vecteurs de support de la stratégie « un contre tous », ce qui signifie qu'il existe un certain nombre de données d'apprentissage qui sont sélectionnées comme vecteurs de support par la stratégie « un contre un » et ne le sont pas avec la stratégie « un contre tous ».

Tableau XXII

Comparaison du nombre de vecteurs de support

	Chiffres	Lettres
SVM-OAO	5,753	9,152
SVM-OAA	8,514	11,109
Combinaison	**9,830**	**12,418**

3.2.4.3 Conclusion

En définitive, les probabilités estimées en combinant les deux stratégies ne s'avèrent que très légèrement plus précises que celles qui sont obtenues à l'aide de la stratégie « un contre tous ». Ce résultat, bien qu'un peu décevant, est tout de même intéressant. Nous pouvons effectivement en conclure que l'idée d'atténuer les probabilités problématiques de la stratégie « un contre un » en utilisant celles estimées à l'aide de la stratégie « un contre tous » semble fonctionner. D'autre part, si les probabilités obtenues en combinant les deux stratégies sont très proches de celles estimées par la stratégie « un contre tous », c'est que vraisemblablement les probabilités estimées par la stratégie « un contre un » n'apportent que très peu d'informations supplémentaires. Une perspective intéressante serait alors de revenir au niveau du processus d'extraction de caractéristiques et de chercher à trouver des caractéristiques adaptées à chaque sous-problème. Les probabilités « locales » seraient alors estimées à partir des sorties des SVM de la stratégie « un contre un » qui seraient donc entraînés sur des espaces de représentation différents, tandis que les probabilités « globales » seraient estimées à partir des sorties des SVM de la stratégie « un contre tous » qui eux seraient entraînés sur un même espace de représentation et permettraient de combiner efficacement les probabilités « locales ».

3.3 Comparaison SVM *vs.* MLP

Maintenant que nous avons comparé différentes méthodes pour estimer des probabilités, nous proposons de comparer les résultats obtenues avec les SVM avec ceux obtenus avec notre classifieur de référence, à savoir le MLP. Cependant, la fonction d'erreur qui était utilisée dans le chapitre précédent, pour effectuer l'apprentissage des réseaux MLP était la SSE, tandis que la fonction d'erreur utilisée lors du post-traitement des SVM est la NLL. Il nous a donc semblé plus rigoureux d'effectuer à nouveau l'apprentissage des MLP, en utilisant la NLL. Les résultats en termes de précision se sont avérés similaires à ceux obtenus précédemment. Par contre, la vitesse de convergence de l'algorithme d'apprentissage a été accélérée d'environ un facteur 10. Par ailleurs, nous avons remarqué dans le cas de la stratégie « un contre tous », que l'utilisation de la fonction softmax était préférable à l'utilisation de sigmoïdes. Ainsi, nous avons effectué de nouveaux apprentissage en remplaçant les sigmoïdes de la couche de sortie du MLP, par des fonctions softmax. Les résultats obtenus sont reportés aux tableaux XXIII et XXIV. Nous pouvons alors constater que l'utilisation de la fonction softmax permet là encore d'obtenir des résultats légèrement plus précis.

Tableau XXIII

Résultats obtenus avec un MLP sur la base de chiffres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
Sigmoïde	0,74	4,21	2 257
Softmax	**0,73**	**3,61**	**1 710**

Tableau XXIV

Résultats obtenus avec un MLP sur la base de lettres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
Sigmoïde	3,68	33,01	2 603
Softmax	**3,47**	**33,58**	**1 611**

Finalement, nous pouvons comparer aux tableaux XXV et XXVI, les résultats obtenus avec les nouveaux MLP et ceux obtenus avec les SVM de la stratégie « un contre tous ». De plus, les compromis erreur-rejet obtenus avec les deux types de classifieur sont présentés aux figures 29 et 30. Nous pouvons alors constater que les probabilités estimées par les SVM sont significativement plus précises que celles estimés par le MLP.

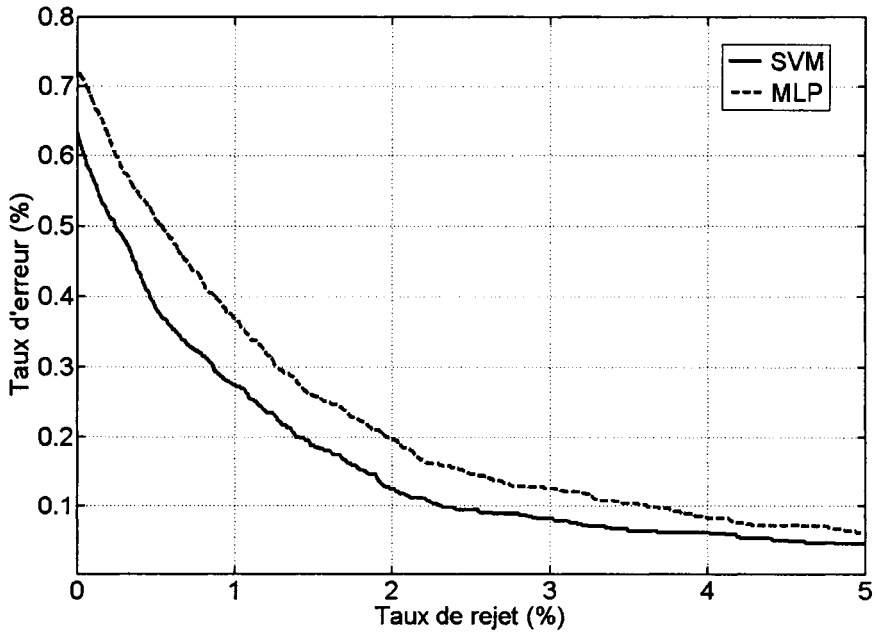


Figure 29 Compromis erreur-rejet obtenus sur la base de chiffres

Tableau XXV

Comparaison MLP *vs.* SVM sur la base de chiffres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
MLP	0,73	3,61	1 710
SVM	**0,63**	**2,30**	**1 310**

Sur la base de chiffres, la différence de 0,1% du taux d'erreur peut ne pas sembler très importante, notons cependant que ceci représente tout de même près de 14% des erreurs commises par le MLP. De plus, pour atteindre un taux d'erreur de 0,1%, il est nécessaire de rejeter 38% de chiffres en moins avec les SVM qu'avec le MLP.

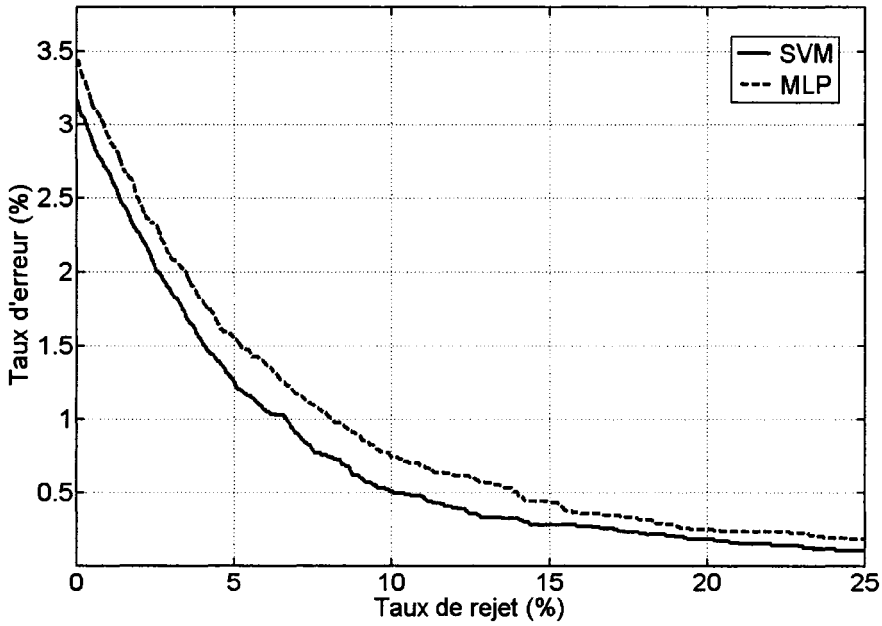


Figure 30 Compromis erreur-rejet obtenus sur la base de lettres

Tableau XXVI

Comparaison MLP *vs.* SVM sur la base de lettres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL
MLP	3,47	33,58	1 611
SVM	**3,18**	**25,52**	**1 375**

De même, pour atteindre un taux d'erreur de 0,1%, il est nécessaire de rejeter 30% de lettres en moins avec les SVM qu'avec le MLP et en l'absence de rejet, les SVM commettent 9% d'erreur en moins que le MLP.

D'autre part, de manière à comparer les deux classifieurs en termes de complexité, nous avons utilisé la méthode proposée par de Ridder *et al.* [48], qui consiste a évaluer le nombre d'opérations élémentaires (FLOP, pour FLoating-point OPeration) nécessaire à la prise de décision. Ils considèrent alors qu'une addition, une soustraction, une multiplication ou une division ont la même complexité, tandis que l'évaluation d'une fonction exponentielle sera considérée comme 2,5 fois plus complexe. Les résultats obtenus sont reportés au tableau XXVII.

Tableau XXVII

Complexité nécessaire à la prise de décision (KFLOP[1])

	Chiffres	Lettres
MLP	23	32
SVM	2 300	3 007

[1] 1 KFLOP = 1000 FLOP

Nous pouvons alors constater que la quantité de calcul nécessaire à la prise de décision est environ 100 fois plus importante dans le cas des SVM que dans le cas du MLP.

3.4 Bilan

Pour conclure, nous proposons de récapituler les quelques conclusions qui se sont dégagées de ce chapitre :

- La solution la mieux adaptée pour calibrer la sortie d'un SVM consiste à utiliser une sigmoïde dont les paramètres seront optimisés en minimisant la NLL.

- Dans le cas de la stratégie « un contre tous », il est préférable d'utiliser la fonction softmax dont les paramètres seront optimisés en minimisant la NLL.

- Dans le cas de la stratégie « un contre un », parmi les trois méthodes testées, dans notre cas, c'est celle proposée par Price *et al.* [19] qui s'est révélée être la plus adaptée pour combiner efficacement les probabilités « locales ».

- Sur la base de chiffres, la stratégie « un contre tous » s'avère significativement plus précise que la stratégie « un contre un », tandis que sur la base de lettres la différence est beaucoup moins importante.

- La combinaison des deux stratégies ne s'avère que très légèrement plus précise que la stratégie « un contre tous », mais ouvre la porte à des perspectives intéressantes, concernant l'extraction de caractéristiques spécialisées dans chaque sous-problème.

- Les probabilités estimées à l'aide des SVM s'avèrent significativement plus précises que celles obtenues avec un MLP, mais la complexité nécessaire à la prise de décision est beaucoup plus importante avec les SVM.

Finalement, dans bon nombre d'applications, le temps nécessaire à la prise de décision est un critère important. Nous pouvons alors nous demander si le gain en précision apporté par l'utilisation de SVM justifie une telle augmentation de la complexité. Nous nous intéresserons donc dans le prochain chapitre à l'accélération de la prise de décision.

CHAPITRE 4

ACCÉLÉRATION DE LA PRISE DE DÉCISION

Comme nous venons de le voir dans le chapitre précédent, les SVM permettent d'estimer des probabilités plus précises que celles estimées par un MLP. Cependant, ce gain en précision se fait au détriment de la complexité de calcul nécessaire à la prise de décision. Ainsi, nous proposerons dans ce chapitre, une nouvelle approche dont l'objectif sera d'accélérer la prise de décision des SVM, sans dégrader les performances en généralisation.

4.1 État de l'art

Bien entendu, le problème de l'accélération de la prise de décision des SVM a déjà été étudié et plusieurs méthodes sont proposées dans la littérature. Nous pouvons alors distinguer trois types d'approches différentes et parfois complémentaires.

4.1.1 Réduction du nombre de vecteurs de support

Étant donné que la complexité de calcul nécessaire à l'évaluation de la valeur de sortie d'un SVM est directement proportionnelle au nombre de vecteurs de support, une solution évidente pour accélérer la prise de décision, consiste à réduire le nombre de vecteurs de support.

Une première méthode dénommée RS-SVM (de l'anglais, Reduced Set-SVM) a été proposée par Burges [49] et utilisée avec succès pour la reconnaissance de chiffres manuscrits par Burges et Schölkopf [50]. Il s'agit d'une technique de post-traitement qui consiste à remplacer l'ensemble des N_S vecteurs de support par un ensemble réduit de N_X nouveaux vecteurs.

Pour déterminer les valeurs de ces nouveaux vecteurs z_i et des coefficients β_i qui leurs sont associés, Burges [49] propose d'utiliser une méthode de gradient conjugué dont l'objectif sera de minimiser :

$$d = \left\| \Psi - \Psi' \right\|^2 \tag{4.1}$$

où Ψ représente le vecteur normal à l'hyperplan de l'espace de redescription :

$$\Psi = \sum_{i=1}^{N_S} \alpha_i \Phi(x_i) \tag{4.2}$$

et Ψ' le vecteur normal à l'hyperplan défini par les nouveaux vecteurs :

$$\Psi' = \sum_{i=1}^{N_X} \beta_i \Phi(z_i) \tag{4.3}$$

Notons par ailleurs, que l'astuce du noyau permet de calculer d sans avoir besoin de calculer Ψ et Ψ'.

Si cette méthode s'avère efficace, son principal inconvénient reste la complexité de calcul nécessaire à l'optimisation des $(N_C + 1)N_X$ paramètres, N_C représentant le nombre de caractéristiques et N_X le nombre de vecteurs z_i utilisés. De plus, l'existence de minimums locaux nécessite d'effectuer plusieurs optimisations en utilisant des initialisations différentes des paramètres. D'autre part, il semble difficile de fixer *a priori*, le nombre de vecteurs z_i à utiliser. Il sera donc nécessaire de tester plusieurs valeurs de N_X, ce qui augmente encore considérablement la complexité de calcul de cette étape de post-traitement.

Ainsi, pour accélérer cette méthode, Schölkopf *et al.* [51] proposent d'utiliser un algorithme incrémental qui débute avec un seul vecteur z_i et qui ajoute les nouveaux vecteurs un à un, de manière à ce qu'à chaque itération, seules les valeurs du vecteur ajouté et la valeur du coefficient β_i associé, soit $\left(N_C + 1\right)$ paramètres, sont optimisés. Cependant, le problème des minimums locaux est toujours présent et la nouvelle méthode reste donc relativement complexe.

Pour éviter ce problème de minimums locaux, Nguyen et Ho [52] proposent une autre méthode qui consiste à éliminer les vecteurs de support un à un, en les combinant deux à deux, de manière à créer de nouveaux vecteurs :

$$z = k\,x_i + (1-k)x_j \tag{4.4}$$

Il est alors nécessaire d'optimiser à chaque itération que deux paramètres, le paramètre k qui possède une solution optimale unique et le coefficient β_i associé qui est calculé à partir de la formule proposée par Schölkopf *et al.* [51]. Cette procédure permet donc de réduire très facilement le nombre de vecteurs de support, mais les résultats s'avèrent moins précis que ceux obtenus avec la méthode de Burges [49]. Nguyen et Ho [52] proposent donc d'utiliser dans un second temps la méthode de Burges [49] pour raffiner les résultats. La présence de minimum locaux ne pose alors plus de problème car le point de départ de la descente de gradient est très proche de la solution optimale qui serra donc atteinte très rapidement.

Une autre solution a été introduite récemment par Tang et Mazzoni [53] pour éviter le problème des minimums locaux et réduire la complexité de cette étape de post-traitement. La méthode proposée par Tang et Mazzoni [53] consiste alors à utiliser un algorithme évolutif de type « Differential Evolution » qui est plus rapide et beaucoup moins sensible au minimums locaux qu'une descente de gradient. Cependant, ce type de

méthode étant moins précis qu'une descente de gradient, Tang et Mazzoni [53] proposent de combiner leur méthode avec celle proposée par Burges [49]. D'autre part, ils constatent que les méthodes existantes ont toutes été développées pour le cas binaire et sont appliquées au cas multi-classes, en réduisant séparément le nombre de vecteurs de support de chaque SVM. Les vecteurs alors obtenus ne sont donc utilisés que par un seul SVM ce qui ne semble pas très optimal. Ainsi, Tang et Mazzoni [53] proposent de ré-entraîner les différents SVM avec l'ensemble des nouveaux vecteurs, de manière à ce que certains vecteurs soient partagés par plusieurs SVM. Ils montrent alors expérimentalement que cette étape supplémentaire peut permettre d'améliorer les performances en généralisation tout en conservant la même complexité.

En effet, il est important de signaler que si ces différentes méthodes permettent toutes de réduire très fortement la complexité liée à la prise de décision, elles entraînent généralement une perte de précision plus ou moins importante selon le nombre de vecteurs utilisés. Par ailleurs, notons qu'une méthode de simplification exacte, c'est-à-dire sans aucune perte, a été proposée par Downs *et al.* [54] qui constatent que certains vecteurs de support sont linéairement dépendants dans l'espace de redescription et peuvent donc être éliminés sans altérer la solution. Il est alors uniquement nécessaire d'identifier ces vecteurs et de modifier les multiplicateurs de Lagrange des vecteurs non supprimés. Cependant, cette méthode ne permet généralement de supprimer que très peu de vecteurs de support.

4.1.2 Utilisation d'une procédure séquentielle d'évaluation des sorties

Partant de l'hypothèse que pour certaines données, la « bonne » décision peut être prise en n'exploitant qu'une partie des vecteurs de support, un autre type d'approche pour accélérer la prise de décision consiste à évaluer de manière séquentielle la sortie du SVM.

Une première méthode a été proposée par Romdhani *et al.* [55] qui utilisent la procédure incrémentale proposée par Schölkopf *et al.* [51] pour construire un ensemble de nouveaux vecteurs. En effet, cette procédure permet non seulement de réduire le nombre de vecteurs utilisés par le RS-SVM, mais aussi d'ordonner les vecteurs selon leur pouvoir discriminant et de déterminer à chaque itération les coefficients qui permettent d'approximer au mieux la frontière de décision initiale. Ainsi, Romdhani *et al.* [55] proposent d'évaluer la sortie du RS-SVM itérativement en faisant intervenir les vecteurs un à un, jusqu'à ce que la sortie soit supérieure à un certain seuil. Le nombre de vecteurs utilisés pour prendre la décision variera donc dynamiquement en fonction de la donnée à classer.

Cette procédure d'évaluation séquentielle est reprise par Parrado-Hernandez *et al.* [56] qui l'utilisent pour accélérer la prise de décision de leur GSVC (de l'anglais, Growing Support Vector Classifier). En effet, Parrado-Hernandez *et al.* [56] proposent une procédure pour construire incrémentalement un SVM de manière à pouvoir contrôler la complexité sans avoir besoin d'utiliser une procédure de post-traitement.

Par ailleurs, l'idée de procédure séquentielle est reprise par DeCoste et Mazzoni [57] qui proposent d'ordonner les vecteurs de support en fonction de la donnée à classer. Pour ce faire, ils utilisent l'analyse en composante principale (ACP) pour effectuer une projection dans un sous-espace vectoriel de très petite dimension, dans lequel les vecteurs de support seront ordonnés en fonction de leurs distances à la donnée traitée.

4.1.3 Exploitation de la modularité des stratégies multi-classes

Un troisième type d'approche consiste à exploiter l'aspect modulaire des SVM multi-classes. Ainsi, dans le cas de la stratégie « un contre un », Platt *et al.* [25] montrent que pour un problème à c classes, la prise de décision ne nécessite pas d'évaluer la sortie de l'ensemble des $c(c\text{-}1)/2$ SVM, mais peut être prise en n'évaluant la sortie que de $(c\text{-}1)$

SVM. L'idée, que nous avons déjà présenté au chapitre 2, consiste alors à utiliser un arbre de décision dont chaque nœud est un SVM et chaque feuille une des classes du problème. Cependant, si la méthode proposée par Platt *et al.* [25] permet bien d'accélérer la prise de décision, le facteur d'accélération n'est pas égal à $c/2$, mais beaucoup moins important, du fait que bon nombre de vecteurs de support sont partagés par plusieurs SVM et que la quantité de calcul nécessaire à la prise de décision n'est pas proportionnelle au nombre de SVM mais bien au nombre de vecteurs de support.

Un autre type d'approche est proposé par Bellili *et al.* [58] qui partent du principe que parmi les erreurs de classification commises par un MLP, la véritable classe est dans la majorité des cas en deuxième position. Ainsi, ils proposent d'utiliser un MLP pour sélectionner parmi l'ensemble des SVM issus de la stratégie « un contre un », celui qui sera chargé de prendre la décision. Cette combinaison permet alors d'améliorer les résultats du MLP, tout en limitant la quantité de calcul nécessaire à la prise de décision. Cependant, les résultats obtenus restent significativement moins bons, que ceux obtenus avec l'ensemble des SVM de la stratégie « un contre tous ».

4.1.4 Conclusion

Si l'on effectue la synthèse des approches rencontrées dans la littérature, nous pouvons constater que bien qu'il existe de nombreuses méthodes très différentes et parfois complémentaires, certaines lacunes subsistent.

Notons tout d'abord qu'aucune de ces études n'a été réalisée dans un contexte probabiliste et que la précision des méthodes proposées est toujours évaluée en estimant le taux d'erreur en généralisation. Or, comme nous l'avons vu précédemment, le taux d'erreur n'est pas un critère assez fin pour évaluer la qualité des probabilités estimées.

Ainsi, les méthodes basées sur l'utilisation d'une procédure séquentielle d'évaluation des sorties des SVM [55-57] semblent peu adaptées à l'estimation de probabilités *a posteriori*. En effet, en n'évaluant les sorties des SVM qu'avec un sous ensemble des vecteurs de support, les valeurs de sortie sont alors modifiées, ce qui peut ne pas avoir d'incidence sur la classification, mais risque d'introduire un biais non négligeable lors de l'estimation des probabilités.

De même, de part sa nature, la procédure proposée par Platt *et al.* [25] qui consiste à n'utiliser qu'un nombre restreint des SVM issus de la stratégie « un contre un », ne permet pas d'estimer des probabilités.

Par contre, la méthode de réduction du nombre de vecteurs de support proposée par Downs *et al.* [54] étant une simplification exacte, elle ne modifiera donc pas la qualité des probabilités estimées. Cependant, elle ne permet de réduire que légèrement la complexité et si les autres méthodes basées sur la réduction du nombre de vecteurs [49-53] permettent de réduire fortement la complexité, ceci se fait aux dépens de la précision.

De même, si la méthode proposée par de Bellili *et al.* [58] permet d'accélérer fortement la prise de décision en combinant MLP et SVM, elle ne permet pas d'obtenir des résultats aussi précis que ceux obtenus avec les SVM de la stratégie « un contre tous », ni de contrôler le compromis entre précision et complexité.

Finalement, nous pouvons en conclure qu'aucune des méthodes existantes ne répond parfaitement au problème de l'accélération de la prise de décision. Nous avons donc choisi d'élaborer de nouvelles méthodes mieux adaptées à notre problématique.

4.2 Description de l'approche proposée

Étant donné les conclusions du chapitre précédent, il nous a semblé intéressant de reprendre l'idée consistant à combiner MLP et SVM. En effet, lorsque le nombre d'exemples d'apprentissage est suffisamment important, nous avons pu constater que la différence de précision entre MLP et SVM n'est pas si importante, tandis que la quantité de calcul nécessaire à la prise de décision est très largement favorable au MLP. Nous chercherons alors à élaborer dans un contexte probabiliste de nouveaux schémas de combinaison qui permettront d'accélérer la prise de décision, sans dégrader la précision des probabilités estimées. Nous poserons alors l'hypothèse que le biais lié à l'estimation des probabilités par le MLP est négligeable pour les probabilités les plus faibles. Nous ne chercherons donc à ré-estimer à l'aide de SVM que les probabilités les plus fortes.

Le premier schéma de combinaison proposé est alors une simple adaptation de la méthode proposée par Bellili *et al.* [58] pour permettre l'estimation de probabilité. Pour chaque donnée inconnue, les probabilités *a posteriori* sont estimées dans un premier temps à l'aide d'un MLP, puis triées en ordre décroissant :

$$\hat{P}(\omega_{\ell(1)} \mid x) > \hat{P}(\omega_{\ell(2)} \mid x) > \ldots > \hat{P}(\omega_{\ell(c)} \mid x) \tag{4.5}$$

$\ell(1)$ correspondant donc à l'indice de la classe la plus probable et $\ell(c)$ à l'indice de la classe la moins probable. Dans un second temps, seules les deux probabilités les plus fortes sont ré-estimées à l'aide du SVM approprié :

$$\hat{P}(\omega_{\ell(1)} \mid x) = \hat{P}(\omega_{\ell(1)} \mid f_{\ell(1),\ell(2)}(x)) \times \left(1 - \sum_{i=3}^{c} \hat{P}(\omega_{\ell(i)} \mid x)\right) \text{ et} \tag{4.6}$$

$$\hat{P}(\omega_{\ell(2)} \mid x) = \hat{P}(\omega_{\ell(2)} \mid f_{\ell(1),\ell(2)}(x)) \times \left(1 - \sum_{i=3}^{c} \hat{P}(\omega_{\ell(i)} \mid x)\right) \tag{4.7}$$

où les probabilités $\hat{P}(\omega_{\ell(1)} \mid f_{\ell(1),\ell(2)}(x))$ et $\hat{P}(\omega_{\ell(2)} \mid f_{\ell(1),\ell(2)}(x))$ sont estimées en calibrant à l'aide d'une sigmoïde la sortie $f_{\ell(1),\ell(2)}(x)$ du SVM entraîné à séparer les données de la classe $\omega_{\ell(1)}$ de celles de la classe $\omega_{\ell(2)}$. Ce schéma de combinaison, qui est illustré à la figure 31, fait donc appel aux SVM issus de la stratégie « un contre un » pour raffiner les probabilités des deux classes considérées par le MLP comme étant les plus probables (Top 2), nous le nommerons donc « Top2-OAO ».

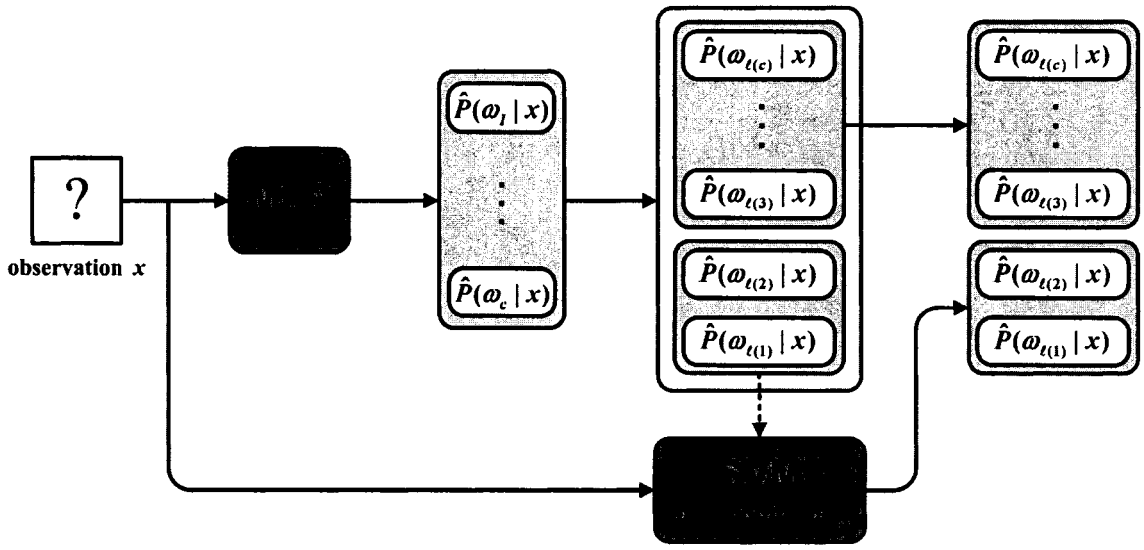


Figure 31 Illustration du schéma de combinaison Top2-OAO

Cependant, comme nous l'avons constaté dans le chapitre précédent, pour ce type de problématique, la stratégie « un contre tous » s'avère plus précise que la stratégie « un contre un ». Nous proposons donc un deuxième schéma de combinaison qui utilisera

non pas un SVM « un contre un » mais deux SVM « un contre tous » pour ré-estimer les deux probabilités les plus fortes :

$$\hat{P}(\omega_{\ell(1)} \mid x) = \frac{\exp\left(A_{\ell(1)} f_{\ell(1)}(x) + B_{\ell(1)}\right)}{\sum_{i=1}^{2} \exp\left(A_{\ell(i)} f_{\ell(i)}(x) + B_{\ell(i)}\right)} \times \left(1 - \sum_{i=3}^{c} \hat{P}(\omega_{\ell(i)} \mid x)\right) \text{ et} \tag{4.8}$$

$$\hat{P}(\omega_{\ell(2)} \mid x) = \frac{\exp\left(A_{\ell(2)} f_{\ell(2)}(x) + B_{\ell(2)}\right)}{\sum_{i=1}^{2} \exp\left(A_{\ell(i)} f_{\ell(i)}(x) + B_{\ell(i)}\right)} \times \left(1 - \sum_{i=3}^{c} \hat{P}(\omega_{\ell(i)} \mid x)\right) \tag{4.9}$$

où $f_{\ell(i)}(x)$ représente la sortie du SVM entraîné à séparer les données de la classes $\omega_{\ell(i)}$ de toutes les autres données et où les coefficients $A_{\ell(i)}$ et $B_{\ell(i)}$ sont ceux obtenus lors de l'optimisation de la fonction softmax exploitant tous les SVM de la stratégie « un contre tous ». Ce schéma de combinaison que nous nommerons donc « Top2-OAA » est illustré à la figure 32.

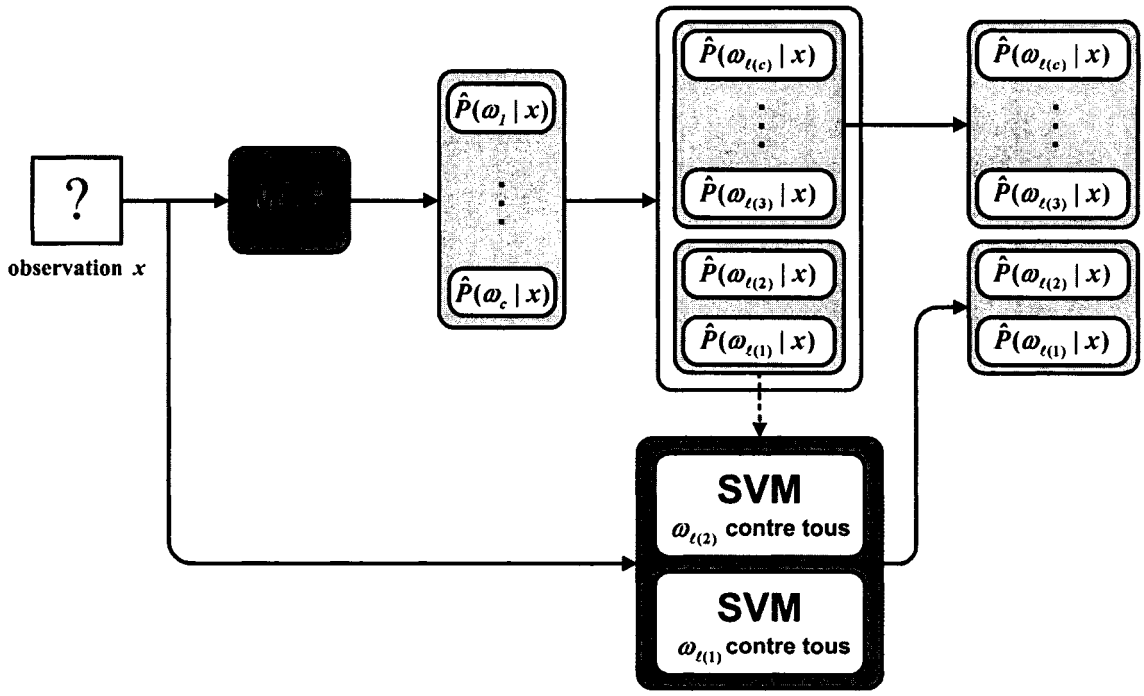


Figure 32 Illustration du schéma de combinaison Top2-OAA

Bien entendu, la quantité de calcul nécessaire à la prise de décision sera bien plus importante dans le cas du schéma de combinaison Top2-OAA que dans le cas du schéma de combinaison Top2-OAO. En effet, non seulement le système utilise deux SVM au lieu d'un seul, mais de plus, chaque SVM issu de la stratégie « un contre tous » utilise beaucoup plus de vecteurs de support que n'importe lequel des SVM issu de la stratégie « un contre un ».

Ainsi, partant du principe qu'un nombre important de données ne sont pas ambiguës, nous pouvons donc considérer qu'il n'est pas toujours nécessaire de ré-estimer les probabilités. Par contre, il est possible que parfois plus de deux des probabilités estimées par le MLP soient non négligeables. Nous proposons donc de faire varier dynamiquement le nombre de SVM à utiliser en fonction de la donnée à classer. Pour ce

faire, nous déterminerons une liste de N classes dont les probabilités estimées par le MLP ne sont pas négligeables :

$$\hat{P}(\omega_{\ell(N)} \mid x) > \varepsilon > \hat{P}(\omega_{\ell(N+1)} \mid x) \tag{4.10}$$

où ε est un seuil dont la valeur sera fixée en fonction des contraintes du problème.

Finalement, si $N = 1$, la donnée x peut être considérée comme non-ambiguë et aucun SVM ne sera alors utilisé. Par contre, si $N > 1$ alors les N probabilités les plus fortes seront ré-estimées à l'aide des SVM appropriés.

Le seuil ε permettra donc de contrôler le compromis entre précision et complexité. En effet, si la valeur de ε est trop grande, alors le système ne fera quasiment jamais appel aux SVM et n'améliorera donc que très peu la précision du MLP. Par contre, si la valeur de ε est trop petite, alors le système aura tendance à faire appel à beaucoup de SVM. La précision sera alors fortement améliorée, mais ceci au prix d'une complexité excessive.

Pour ré-estimer les N probabilités les plus fortes, il sera alors possible d'utiliser N SVM issus de la stratégie « un contre tous » :

$$\hat{P}(\omega_{\ell(i)} \mid x) = \frac{\exp\left(A_{\ell(i)} f_{\ell(i)}(x) + B_{\ell(i)}\right)}{\sum_{j=1}^{N} \exp\left(A_{\ell(j)} f_{\ell(j)}(x) + B_{\ell(j)}\right)} \times \left(1 - \sum_{j=N+1}^{c} \hat{P}(\omega_{\ell(j)} \mid x)\right) \tag{4.11}$$

Ce schéma que nous nommerons « TopN-OAA » est illustré à la figure 33.

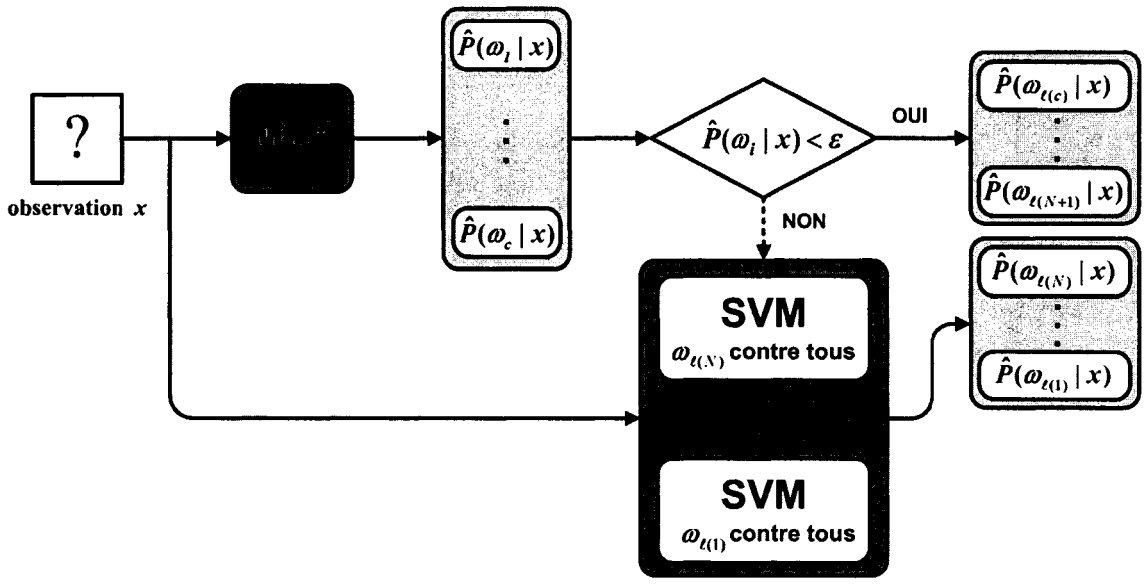


Figure 33 Illustration du schéma de combinaison TopN-OAA

Une autre solution, pour ré-estimer les N probabilités les plus fortes, consisterait à utiliser les SVM issus de la stratégie « un contre un ».

Alors, comme précédemment, si $N = 1$ aucun SVM ne sera utilisé. Si $N = 2$, un seul SVM sera utilisé à la manière du schéma Top2-OAO. Enfin, si $N > 2$ alors $N(N-1)/2$ SVM seront utilisés.

La méthode proposée par Price *et al.* [19] s'étant avérée être la mieux adaptée à notre problématique, elle sera reprise pour ré-estimer les N probabilités :

$$\hat{P}(\omega_{\ell(i)} \mid x) = \frac{1}{\sum_{j=1, j\neq i}^{N} \frac{1}{\hat{P}(\omega_{\ell(i)} \mid f_{\ell(i),\ell(j)}(x))} - (N-2)} \times C \tag{4.12}$$

où C est une constante de normalisation qui garantit que la somme des probabilités soit toujours égale à un :

$$C = \frac{1 - \sum_{j=N+1}^{c} \hat{P}(\omega_{\ell(j)} \mid x)}{\sum_{j=1}^{N} \frac{1}{\sum_{k=1,k\neq j}^{N} \frac{1}{\hat{P}(\omega_{\ell(j)} \mid f_{\ell(j),\ell(k)}(x))} - (N-2)}} \tag{4.13}$$

Ce dernier schéma que nous nommerons « TopN-OAO » est illustré à la figure 34.

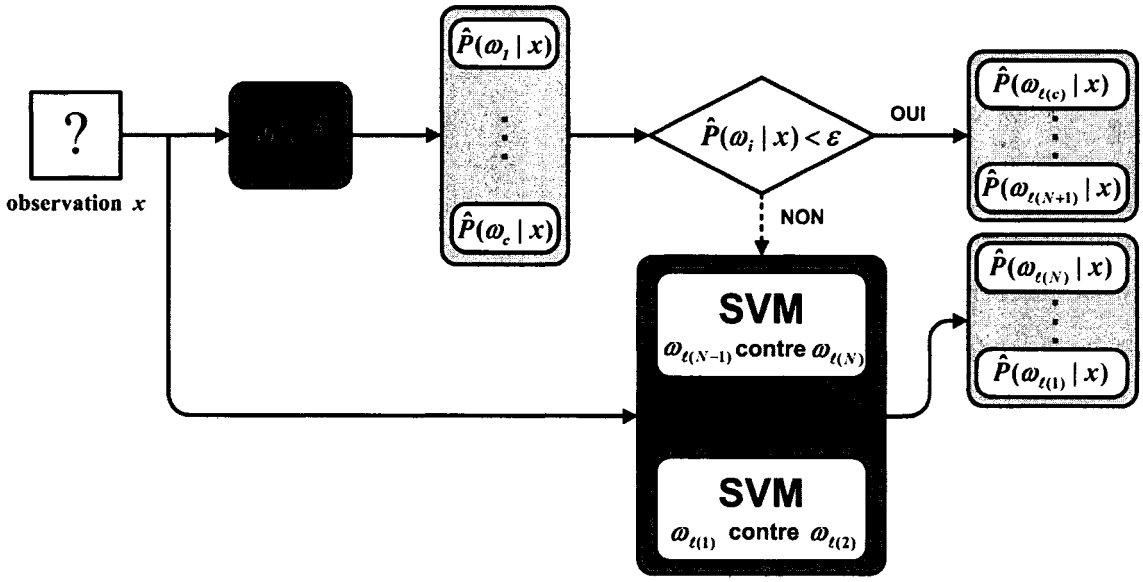


Figure 34 Illustration du schéma de combinaison TopN-OAO

4.3 Résultats expérimentaux

Dans un premier temps, nous avons testé les deux premiers schémas de combinaison, qui ré-estiment systématiquement les deux probabilités les plus fortes estimées par un MLP. Les résultats, en termes de précision et de complexité, sont reportés aux tableaux XXVIII et tableau XXIX.

Tableau XXVIII

Résultats obtenus avec le schéma Top2-OAO et Top2-OAA sur la base de chiffres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL	KFLOP
MLP	0,73	3,61	1 710	23
Top2-OAO	0,73	3,56	1 644	179
Top2-OAA	0,64	2,56	1 488	666
SVM-OAA	0,63	2,30	1 310	2 300

Tableau XXIX

Résultats obtenus avec le schéma Top2-OAO et Top2-OAA sur la base de lettres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL	KFLOP
MLP	3,47	33,58	1 611	32
Top2-OAO	3,35	27,87	1 540	88
Top2-OAA	3,30	24,83	1 495	373
SVM-OAA	3,18	25,52	1 375	3 007

On peut alors constater que les résultats obtenus avec le schéma de combinaison Top2-OAO ne sont pas très satisfaisants. En effet, sur la base de chiffres, les résultats en termes de précision ne s'avèrent pas significativement meilleurs que ceux qui sont obtenus avec le MLP seul. De plus, si ce premier schéma de combinaison permet d'améliorer légèrement la qualité des probabilités du MLP sur la base de lettres, les résultats restent moins bons que ceux qui sont obtenus avec l'ensemble des SVM de la

stratégie « un contre tous ». Par contre, les résultats obtenus avec le schéma Top2-OAA sont beaucoup plus encourageants. En effet, ce deuxième schéma de combinaison permet d'obtenir des résultats proches en termes de précision de ceux qui sont obtenus avec l'ensemble des SVM. Si les valeurs de NLL restent légèrement plus importantes, les taux d'erreur et de rejet sont tout à fait comparables. Nous pouvons même remarquer que le taux de rejet obtenu sur la base de lettres avec la combinaison Top2-OAA est légèrement plus faible que celui obtenu avec l'ensemble des SVM de la stratégie « un contre tous ». Cependant, comme nous l'avions présagé, le second schéma de combinaison s'avère beaucoup plus complexe que le premier et reste finalement 29 fois plus lent que le MLP pour classer les chiffres et environ 12 fois plus lent pour les lettres.

Dans un second temps, nous avons donc testé le schéma de combinaison TopN-OAA qui devrait permettre de réduire la complexité. Nous avons alors évalué sur la base de chiffres l'effet du seuil ε sur la précision et la complexité du système. Concernant la complexité, nous pouvons vérifier à la figure 35 que comme nous pouvions le prévoir, plus la valeur du seuil ε est petite, plus le système utilise en moyenne un nombre important de vecteurs de support. De même, concernant la précision, nous pouvons constater à la figure 36 que plus la valeur du seuil ε est petite, plus la valeur de la NLL est faible et donc plus les probabilités estimées sont précises. Ainsi, comme nous l'avions envisagé précédemment, ce seuil permet bien de contrôler le compromis entre complexité et précision.

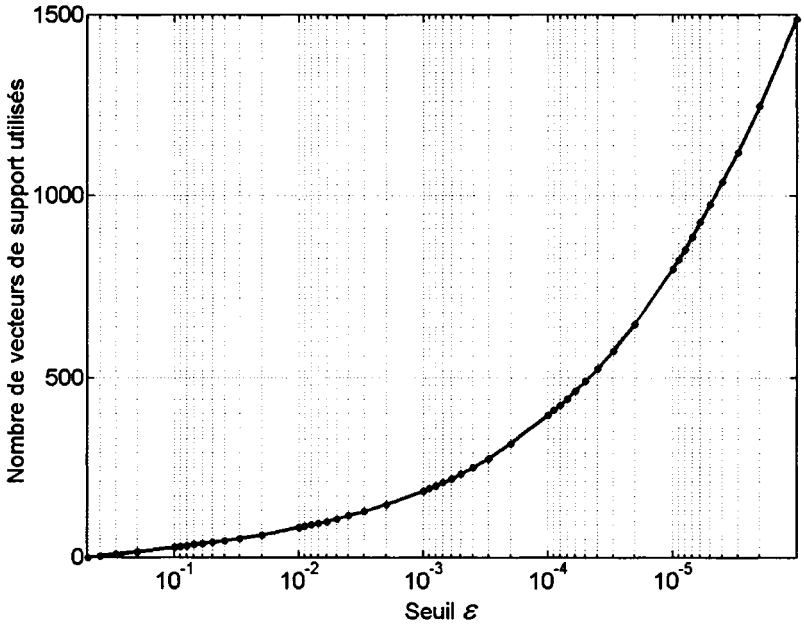


Figure 35 Effet du seuil ε sur la complexité

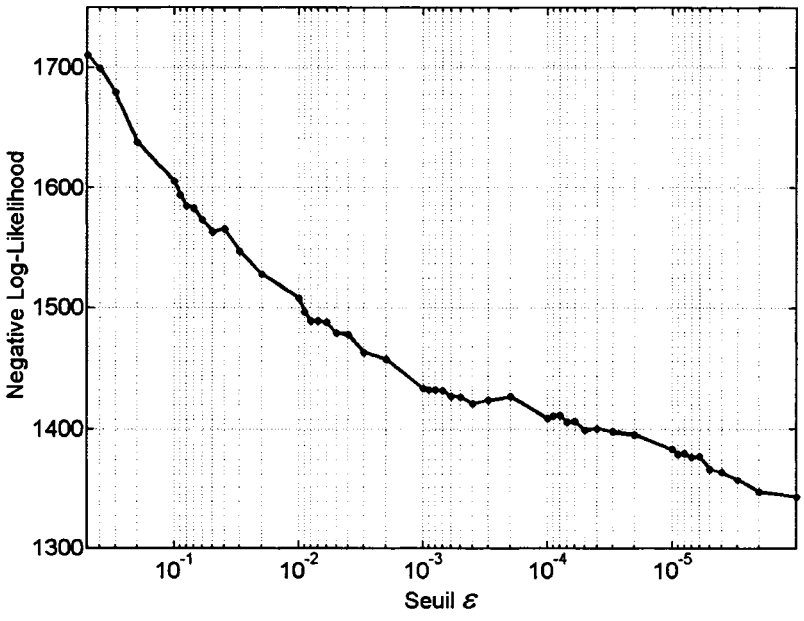


Figure 36 Effet du seuil ε sur la NLL

D'autre part, nous pouvons observer à la figure 37 qu'il n'est pas intéressant d'utiliser une valeur de seuil trop petite. En effet, le taux d'erreur de 0,64 % obtenu lors de l'utilisation de l'ensemble des SVM est atteint dès $\varepsilon = 10^{-1}$ et se stabilise ensuite.

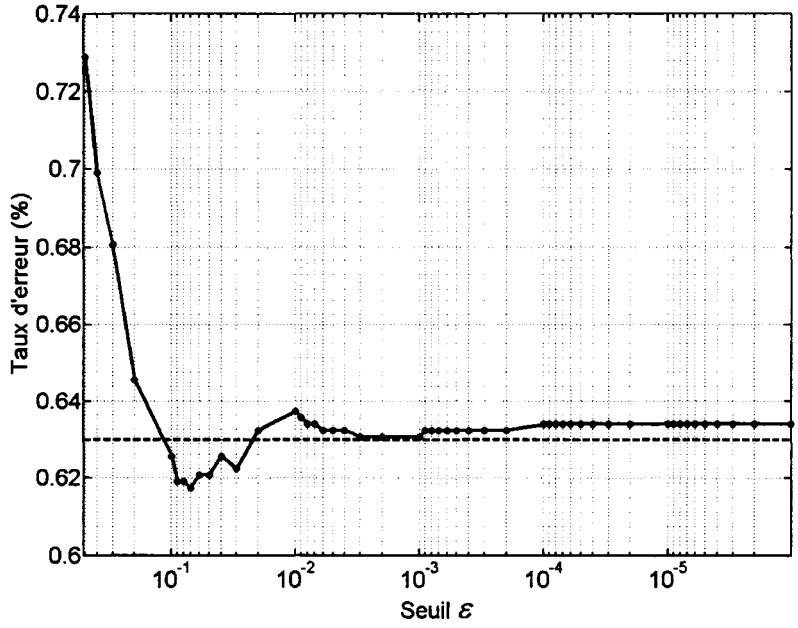


Figure 37 Effet du seuil sur le taux d'erreur

Par contre, pour cette même valeur de seuil, nous pouvons constater à la figure 38 que le taux de rejet est équivalent à celui obtenu avec le MLP seul. Il est alors nécessaire d'utiliser une valeur de seuil plus faible pour atteindre le taux de rejet de 2,30 % obtenu avec l'ensemble des SVM.

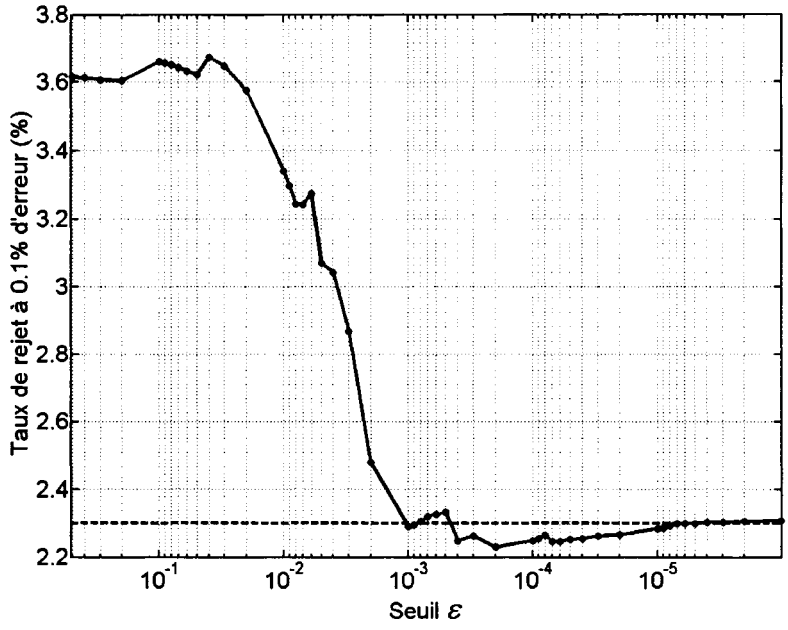
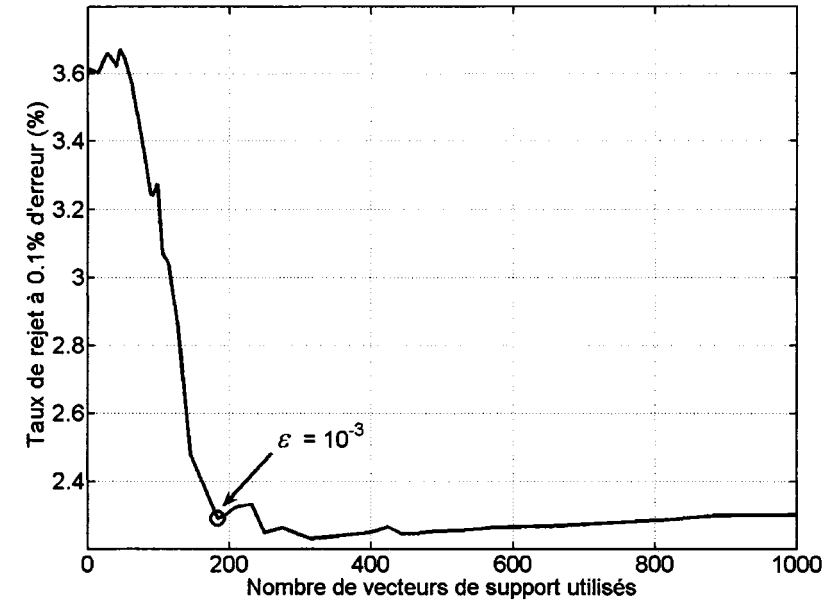


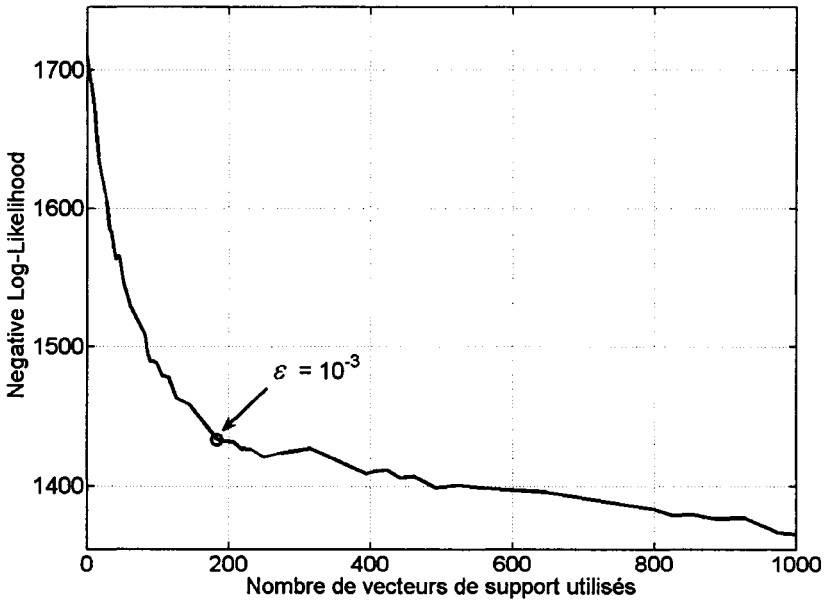
Figure 38 Effet du seuil sur le taux de rejet

De manière à mieux visualiser le compromis entre précision et complexité, nous avons reporté à la figure 39, les résultats obtenus en combinant les résultats présentés à la figure 35 avec ceux de la figure 36 et ceux de la figure 38.

La valeur de seuil $\varepsilon = 10^{-3}$ semble alors permettre de réaliser un compromis intéressant entre précision et complexité. En effet, comme nous pouvons constater à la figure 39-a, cette valeur permet d'obtenir un taux de rejet équivalent à celui obtenu avec l'ensemble des SVM et ce en n'utilisant qu'un nombre très restreint de vecteurs de support. D'autre part, comme nous pouvons le constater à la figure 39-b, s'il est possible d'obtenir des valeurs de NLL légèrement plus faibles, le coût supplémentaire en termes de complexité ne justifie pas nécessairement l'utilisation d'une valeur de seuil plus faible.



(a)



(b)

Figure 39 Compromis entre précision et complexité

Finalement, les résultats obtenus en utilisant cette valeur de seuil sont reportés aux tableaux XXX et XXXI.

Nous pouvons alors constater que le schéma TopN-OAA permet bien de réduire plus fortement la complexité que le schéma Top2-OAA, tout en améliorant légèrement la précision. Les probabilités estimées par ce schéma de combinaison s'avèrent alors tout a fait comparables à celles estimées avec les dix SVM de la stratégie « un contre tous ».

Tableau XXX

Résultats obtenus avec le schéma TopN-OAA sur la base de chiffres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL	KFLOP
MLP	0,73	3,61	1 710	23
Top2-OAA	0,64	2,56	1 488	666
TopN-OAA	**0,63**	**2,29**	**1 433**	**74**
SVM-OAA	0,63	2,30	1 310	2 300

Par ailleurs, rappelons que la valeur de N varie dynamiquement en fonction des données à classer et que le nombre de KFLOP est en fait une moyenne calculée sur l'ensemble des données de la base de test. En pratique, sur la base de chiffres, dans 93 % des cas seul le MLP est utilisé, dans 5 % des cas le système ne fait appel qu'à deux SVM et dans seulement 2 % des cas plus de deux SVM sont utilisés pour ré-estimer les probabilités. Alors, le schéma TopN-OAA s'avère en moyenne 9 fois plus rapide que le schéma Top2-OAA et n'est plus qu'environ 3 fois plus lent que le MLP. Comparativement à l'utilisation des 10 SVM de la stratégie « un contre tous », ce schéma permet finalement d'accélérer la prise de décision d'un facteur 30, sans dégrader la précision du système.

Tableau XXXI

Résultats obtenus avec le schéma TopN-OAA sur la base de lettres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL	KFLOP
MLP	3,47	33,58	1 611	32
Top2-OAA	3,30	24,83	1 495	373
TopN-OAA	**3,18**	**23,68**	**1 457**	**260**
SVM-OAA	3,18	25,52	1 375	3 007

Sur la base de lettres, la différence de complexité entre le schéma TopN-OAA et le schéma Top2-OAA s'avère beaucoup moins importante qu'avec les chiffres. En effet, le MLP étant beaucoup moins précis, il ne permet de traiter directement qu'un peu plus de la moitié des données, tandis que dans environ 20% des cas plus de deux SVM seront nécessaires pour ré-estimer les probabilités. Cependant, la différence de précision entre les deux schémas s'avère alors plus importante qu'avec les chiffres. Comparativement à l'utilisation des 26 SVM de la stratégie « un contre tous », le schéma TopN-OAA permet d'accélérer la prise de décision d'environ un facteur 12, mais reste environ 8 fois plus lent que le MLP.

Pour finir, nous avons testé le schéma TopN-OAO. Les résultats sont reportés aux tableaux XXXII et XXXIII. Nous pouvons alors constater que sur la base de chiffres, ce schéma s'avère largement moins complexe et légèrement plus précis que le schéma Top2-OAO, mais reste significativement moins précis que le schéma TopN-OAA. Ce résultat n'est pas surprenant car sur cette base de données, la stratégie « un contre un » est significativement moins précise que la stratégie « un contre tous ».

Tableau XXXII

Résultats obtenus avec le schéma TopN-OAO sur la base de chiffres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL	KFLOP
MLP	0,73	3,61	1 710	23
Top2-OAO	0,73	3,56	1 644	179
TopN-OAO	**0,70**	**3,31**	**1 567**	**29**
TopN-OAA	0,63	2,29	1 433	74

Par contre, sur la base de lettres, la différence de précision entre ces deux derniers schémas est beaucoup moins importante, ce qui n'est pas étonnant car sur cette base de données, la différence de précision entre la stratégie « un contre un » et la stratégie « un contre tous » est relativement faible.

Tableau XXXIII

Résultats obtenus avec le schéma de combinaison TopN-OAO sur la base de lettres

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	NLL	KFLOP
MLP	3,47	33,58	1 611	32
Top2-OAO	3,35	27,87	1 540	88
TopN-OAO	**3,18**	**27,52**	**1 467**	**86**
TopN-OAA	3,18	23,68	1 457	260

Concernant la complexité, nous pouvons constater à la figure 40 que dans plus de 50% des cas aucun SVM n'est utilisé, dans environ 25% des cas un seul SVM est utilisé et donc dans plus de 20% des cas plus d'un SVM est utilisé. Cependant, nous sommes alors loin des 325 SVM utilisés par la stratégie « un contre un » et comme nous pouvons le constater au tableau XXXIII, le schéma TopN-OAO s'avère finalement de complexité comparable au schéma Top2-OAO.

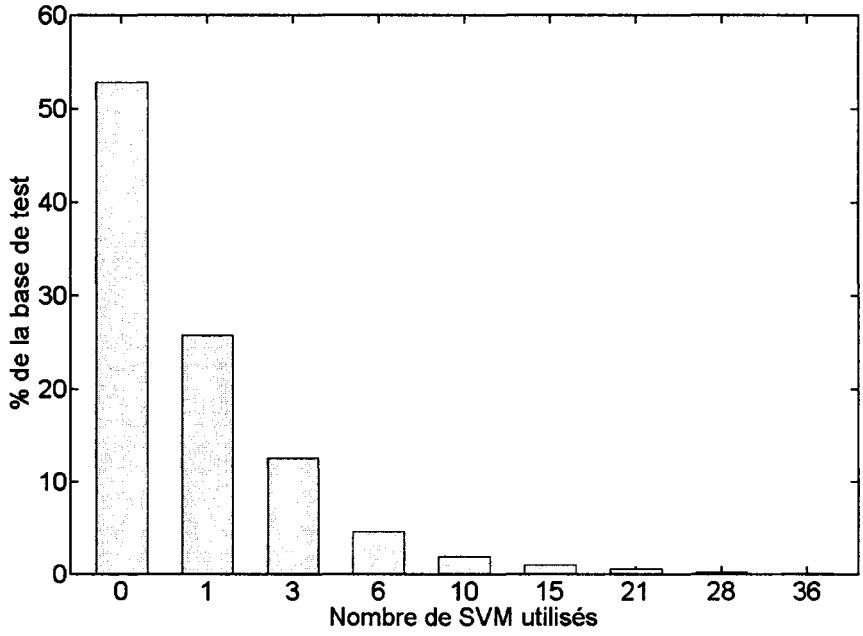


Figure 40 Distribution du nombre de SVM utilisés pour classer les données de test de la base de chiffres avec le schéma de combinaison TopN-OAO

D'autre part, le schéma TopN-OAO s'avère alors trois fois moins complexe que le schéma TopN-OAA. Ainsi, comparativement à l'utilisation des 26 SVM de la stratégie « un contre tous », le schéma TopN-OAO permet d'accélérer la prise de décision d'environ un facteur 35 et n'est donc que 2,7 fois plus lent que le MLP.

Pour finir, nous avons reporté aux figures 41 et 42, les valeurs des taux de rejet et de NLL obtenues avec les schémas de combinaison TopN-OAA et TopN-OAO, pour différentes valeurs de seuil. On peut alors constater que sur cette base de données, le schéma TopN-OAO peut s'avérer aussi précis que le schéma TopN-OAA et permet un meilleur compromis entre précision et complexité. Notons que les pointillés de la figure 41 indiquent le résultat de référence obtenu avec l'ensemble des SVM issus de la stratégie « un contre tous » qui utilise 11 109 vecteurs de support.

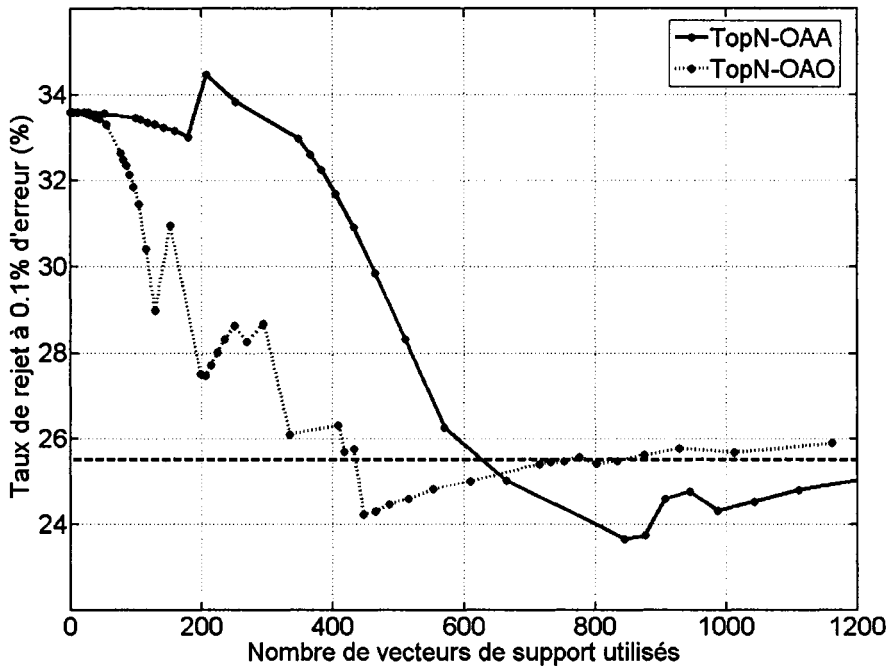


Figure 41 Comparaison en termes de taux de rejet sur la base de lettres

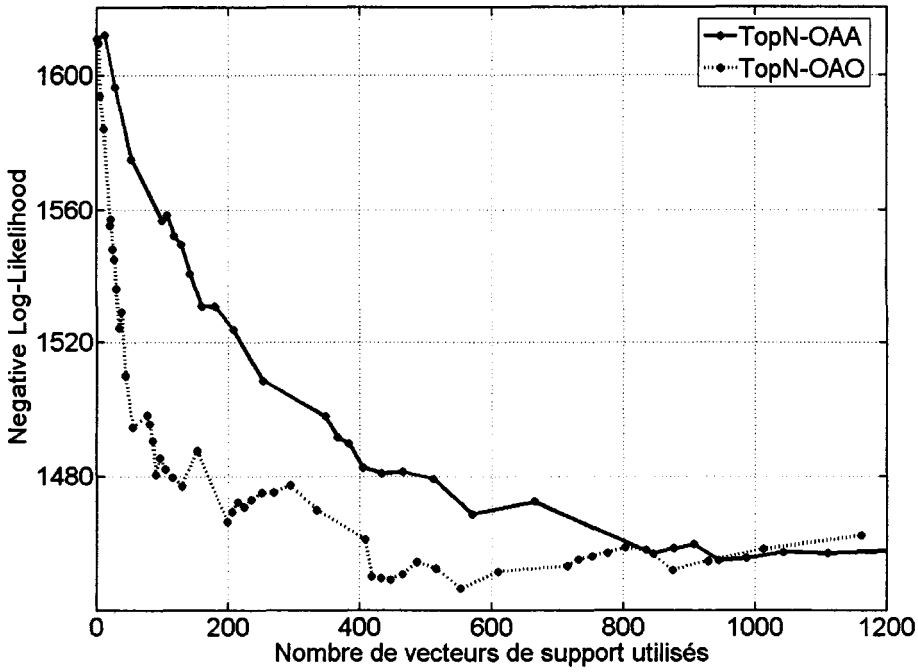


Figure 42 Comparaison en termes de NLL sur la base de lettres

4.4 Conclusion

Nous avons donc proposé dans ce chapitre, deux nouveaux schémas de combinaison dont l'originalité est de faire varier dynamiquement le nombre de SVM à utiliser pour raffiner les probabilités estimées par le MLP. De plus, ces méthodes permettent de contrôler le compromis entre précision et complexité et finalement d'accélérer fortement la prise de décision sans dégrader la qualité des probabilités estimées.

Par ailleurs, notons que les méthodes proposées sont complémentaires d'une approche de réduction du nombre de vecteurs de support. Ainsi, l'utilisation de la méthode de simplification exacte proposée par Downs *et al.* [54] permettrait d'accélérer encore un peu la prise de décision, sans avoir d'impact sur la précision du système.

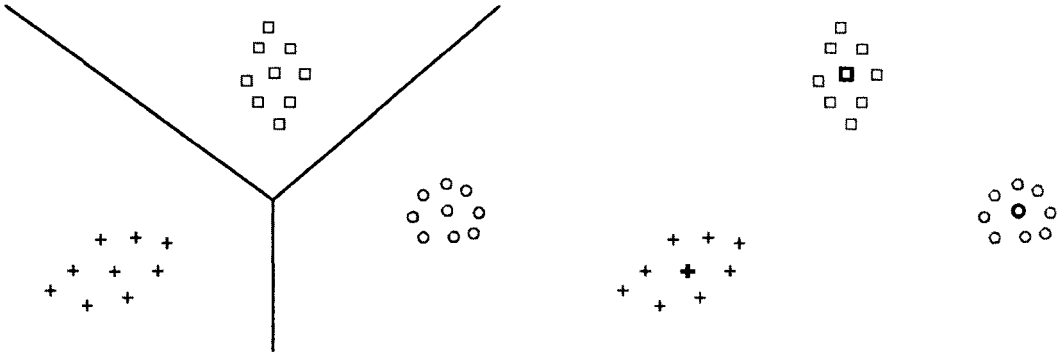
Une autre perspective intéressante pour améliorer le compromis entre complexité et précision consisterait à utiliser des seuils différents pour chacune des classes afin de déterminer quelles probabilités doivent être ré-estimées. Nous pourrions alors nous inspirer des travaux de Oliveira *et al.* [59], qui proposent d'utiliser un algorithme évolutif pour optimiser les différents seuils d'une combinaison en cascade de plusieurs MLP exploitant différents espaces de représentation. De même, il pourrait être pertinent de chercher à réduire la complexité du MLP, soit en réduisant le nombre de neurones cachés, soit en exploitant qu'un sous-ensemble de caractéristiques.

Pour finir, notons que le MLP peut facilement être remplacé par n'importe quel autre classifieur qui permet d'estimer des probabilités *a posterirori*. Ainsi, nous verrons dans le prochain chapitre, que parfois il peut être plus avantageux de combiner les SVM avec d'autres types d'approches de classification, comme par exemple une approche agissant par modélisation.

CHAPITRE 5

COMBINAISON AVEC UNE APPROCHE PAR MODÉLISATION

Parmi l'ensemble des techniques de classification, il est possible de distinguer deux catégories d'approches : celles agissant par séparation et celles agissant par modélisation. L'objectif du premier type d'approche est de déterminer des frontières de décision qui séparent au mieux les classes ; tandis que l'objectif du second type d'approche est de déterminer un modèle le plus fidèle possible de chacune des classes. Un exemple de chaque approche est illustré à la figure 43. Dans le cas de l'approche par séparation, des frontières linéaires, qui sont représentées en gras à la figure 43-a, sont utilisées pour séparer les données des trois classes. Seuls les coefficients de ces trois droites sont alors sauvegardés et ensuite utilisés pour classer les données inconnues en se basant sur leurs positions par rapport à ces droites. Dans le cas de l'approche par modélisation, chaque classe est ici modélisée par un prototype, qui est représenté en gras à la figure 43-b. Seul ces trois prototypes sont alors sauvegardés et ensuite utilisés pour classer les données inconnues en utilisant une mesure de similarité, ou de dissimilarité, tel que la distance Euclidienne.



(a) par séparation (b) par modélisation

Figure 43 Illustration des deux types d'approche de classification

Par ailleurs, il est possible de distinguer deux types de données pouvant causer des problèmes à un classifieur : les données ambiguës qui pourraient appartenir à plusieurs classes et les données aberrantes, plus connues sous le terme anglais d'« outliers », qui n'appartiennent à aucune des classes du problème. Ces deux types de données problématiques peuvent être associés à deux types de rejet : le rejet d'ambiguïté et le rejet d'ignorance, aussi nommé rejet de distance.

Or, si les approches de classification qui agissent par séparation sont, de par leur nature, bien adaptées au premier type de rejet, les données ambiguës se situant au niveau des frontières de décision, elles s'avèrent généralement peu efficaces pour rejeter les données aberrantes. Par contre, les approches de classification qui agissent par modélisation semblent mieux adaptées au second type de rejet, mais s'avèrent généralement moins efficaces pour rejeter les données ambiguës.

Il semble donc intéressant de chercher à combiner les deux types d'approches de classification, de manière à pouvoir rejeter efficacement les deux types de données problématiques. Ainsi, les SVM faisant partie des approches agissant par séparation, nous proposerons dans ce chapitre de combiner les SVM avec une approche agissant par modélisation plutôt qu'avec un MLP qui est aussi une approche agissant par séparation.

De plus, contrairement à un MLP qui est une approche globale, les approches par modélisation présentent l'avantage d'être modulaires. De ce fait, ce type d'approche permet de traiter des problèmes comprenant un grand nombre de classes, comme par exemple dans le cas de la reconnaissance d'idéogrammes chinois ou japonais, où il peut être nécessaire de traiter plusieurs milliers de classes.

5.1 État de l'art

Bien entendu, des nombreux systèmes de classification combinant modélisation et séparation ont déjà été proposés dans la littérature. Parmi les approches existantes il est possible de distinguer deux catégories d'approches : les classifieurs hybrides et les combinaisons de classifieurs.

5.1.1 Classifieurs hybrides

La première catégorie regroupe les classifieurs qui combinent de manière interne des propriétés propres à chacune des deux approches. On parlera alors d'approche hybride.

La façon la plus courante d'obtenir un classifieur hybride consiste à modifier l'algorithme d'apprentissage d'une approche par modélisation, de manière à affiner les frontières de décision en modifiant les modèles qui génèrent des erreurs de classification. Ce type d'approche a notamment été utilisé par Oja et Kohonen [60] qui modélisent chaque classe par un sous-espace vectoriel, par Schwenk [61] qui lui utilise des réseaux auto-associatifs ou encore par Biem [62] avec des HMM.

Dans le même esprit, Liu *et al.* [63] montrent sur un problème de reconnaissance de chiffres manuscrits que le MQDF (Modified Quadratic Discriminant Function), proposé initialement par Kimura *et al.* [64] dans le cadre de la reconnaissance de caractères chinois, permet d'obtenir de meilleurs résultats en terme de rejet d'ignorance qu'un MLP ou un réseau RBF. Par contre, ce classifieur agissant par modélisation s'avère bien moins efficace en termes de taux de classification. Partant de ces constatations, Liu *et al.* [65] proposent d'utiliser un apprentissage discriminant afin de raffiner les paramètres des différents modèles et réduire ainsi le nombre d'erreurs de classification. Leur classifieur hybride, qui est rebaptisé LQDF (Learning Quadratic Discriminant Function), permet finalement d'obtenir des taux d'erreur comparables à ceux obtenus avec un MLP ou un réseau RBF et des résultats en termes de rejet d'ignorance très proche de ceux

obtenus avec le MQDF initial. Cependant, dans un article plus récent, Liu *et al.* [1] montrent que les taux d'erreurs obtenus avec ces différents classifieurs restent supérieurs à ceux rendus possible par l'utilisation de SVM.

Le réseau RBF est un autre exemple d'approche hybride. En effet, comme Bishop le montre dans le chapitre 5 de son livre [66], les poids de la première couche peuvent être fixés en utilisant un algorithme d'apprentissage non-supervisé, alors que les poids de la seconde couche sont eux optimisés à l'aide d'un algorithme d'apprentissage supervisé. La première couche agit donc par modélisation, tandis que la seconde couche agit par séparation. Cependant, notons qu'en procédant ainsi, les résultats en termes de généralisation sont inférieurs à ceux qu'il est possible d'obtenir lorsque les poids de la première couche sont uniquement initialisés avec un algorithme non-supervisé, puis modifiés à l'aide d'un algorithme d'apprentissage supervisé. Par contre, en modifiant ainsi les poids de la première couche, rien ne garantit alors que ceux-ci modélisent encore bien les différentes classes.

5.1.2 Combinaison de classifieurs

La seconde catégorie regroupe les approches qui combinent de manière externe des classifieurs entraînés séparemment. On parlera alors plus de classifieur hybride, mais plutôt de combinaison de classifieurs.

Un premier exemple de ce type d'approche est présenté par Francesconi *et al.* [67] qui proposent de combiner un réseau MLP entraîné à séparer les différentes classes avec des réseaux auto-associatifs entraînés à modéliser chacune des classes. L'objectif visé est alors d'améliorer les capacités de rejet du MLP. Pour ce faire, le MLP est utilisé pour établir une liste des classes les plus probables, puis les réseaux auto-associatifs correspondant à ces classes sont utilisés pour prendre la décision ou rejeter la donnée.

Notons cependant que les deux types de rejet ne sont alors pas dissociés et que l'approche ne s'avère efficace que pour obtenir des taux d'erreur proches de zéro.

Un autre exemple de combinaison est proposé par Ragot et Anquetil [68] qui utilisent un système d'inférence floue hiérarchisé. Contrairement à Francesconi *et al.* [67], ils utilisent une approche par modélisation pour effectuer une pré-classification et une approche par séparation pour rafiner la décision. D'autre part, leur objectif principal est alors d'obtenir un classifieur performant ne nécessitant que peu de ressource mémoire, de manière à pouvoir être implanté sur un agenda électronique ou un téléphone mobile. Finalement, leur système de classification s'avère légerement moins performant que des SVM en termes de taux de classification mais extremement plus léger en termes de ressource mémoire.

De même, afin d'améliorer les performances d'une approche par modélisation basée sur un ensemble de prototypes, Prevost *et al.* [69] proposent d'exploiter au sein d'un second niveau de décision, un ensemble de réseaux MLP entraînés à séparer uniquement les données de deux classes. Partant de la constatation que seulement quelques paires de classes sont responsables de la majorité des confusions observées au premier niveau de décision, ils proposent de n'entraîner qu'un nombre restreint de réseaux. Une approche similaire est proposée par Chou *et al.* [70] pour la reconnaissance d'idéogrammes chinois, sauf que les classifieurs utilisés au second niveau sont alors des SVM.

Vuurpijl et Schomaker [71] proposent eux aussi une approche faisant intervenir un ensemble de prototypes et des SVM. Cependant, la stratégie adoptée pour construire les SVM qui seront exploités au second niveau de leur système est totalement différente. En effet, ils utilisent une seconde base d'apprentissage indépendante de celle utilisée pour déterminer les prototypes. Dans un premier temps, ils associent chaque donnée de cette base au prototype le plus proche, puis ils vérifient ensuite pour chaque prototype, que l'ensemble des données qui lui ont été associées appartiennent bien à la même classe que

lui. Si tel est le cas, cela signifie que ce prototype modélise un ensemble de données non-ambiguës et que la décision pourra donc être prise directement lorsqu'une donnée inconnue lui sera associée. Par contre, si parmi les données associées à ce prototype plusieurs n'appartiennent pas à la même classe que lui, cela signifie que ce prototype risque d'être une source d'erreur de classification. Il semble alors préférable d'utiliser un SVM qui sera entraîné avec les données associées à ce prototype, de manière à séparer les données appartenant à la même classe que le prototype de toutes celles appartenant à des classes différentes. Ainsi, lors de la classification d'une donnée inconnue, s'il existe un SVM associé au prototype le plus proche de la donnée, celui-ci sera utilisé pour prendre la décision finale.

Pour finir, Abou-Moustafa *et al.* [72] proposent de combiner HMM et SVM. L'objectif est alors de pouvoir traiter des séquences temporelles de longueur variable. Chaque classe est donc modélisée à l'aide de deux chaînes de Markov discrètes, l'une verticale et l'autre horizontale. L'ensemble des mesures de vraisemblance obtenues en sortie des différents HMM forme ensuite un vecteur de dimension fixe qui pourra être exploité par des SVM afin de raffiner la décision. Néanmoins, si l'utilisation de SVM permet d'améliorer légèrement les taux de classification, les performances restent limitées par la qualité de la modélisation des HMM.

5.1.3 Conclusion

De nombreuses approches combinant modélisation et séparation ont donc déjà été proposées dans la litttérature. Cependant, les motivations qui nous poussent à proposer une nouvelle approche sont quelque peu différentes de celles qui sont à l'origine des systèmes existants. En effet, à l'exception de Liu *et al.* [65], aucun des auteurs cités précédemment n'a cherché à dissocier les deux types de rejet. De même, à l'exeption de Liu *et al.* [1], aucun des auteurs n'a comparé ces résultats avec ceux qu'il serait possible d'obtenir en utilisant des SVM. Or, tout nous laisse à penser que l'ensemble des

approches existantes s'avereraient moins efficaces en termes de taux de reconnaisssance qu'un système complet à base de SVM.

Nous avons donc choisi d'élaborer une nouvelle approche dont l'objectif sera de tirer le meilleur des deux types d'approches de classification, afin de traiter efficacement les deux types de données problématiques. De plus, dans la continuation du chapitre précédent, nous chercherons à proposer une approche qui permettra d'accélérer la prise de décision, tout en conservant des performances en termes de précision proches de celles rendues possible lors de l'utilisation de l'ensemble des SVM.

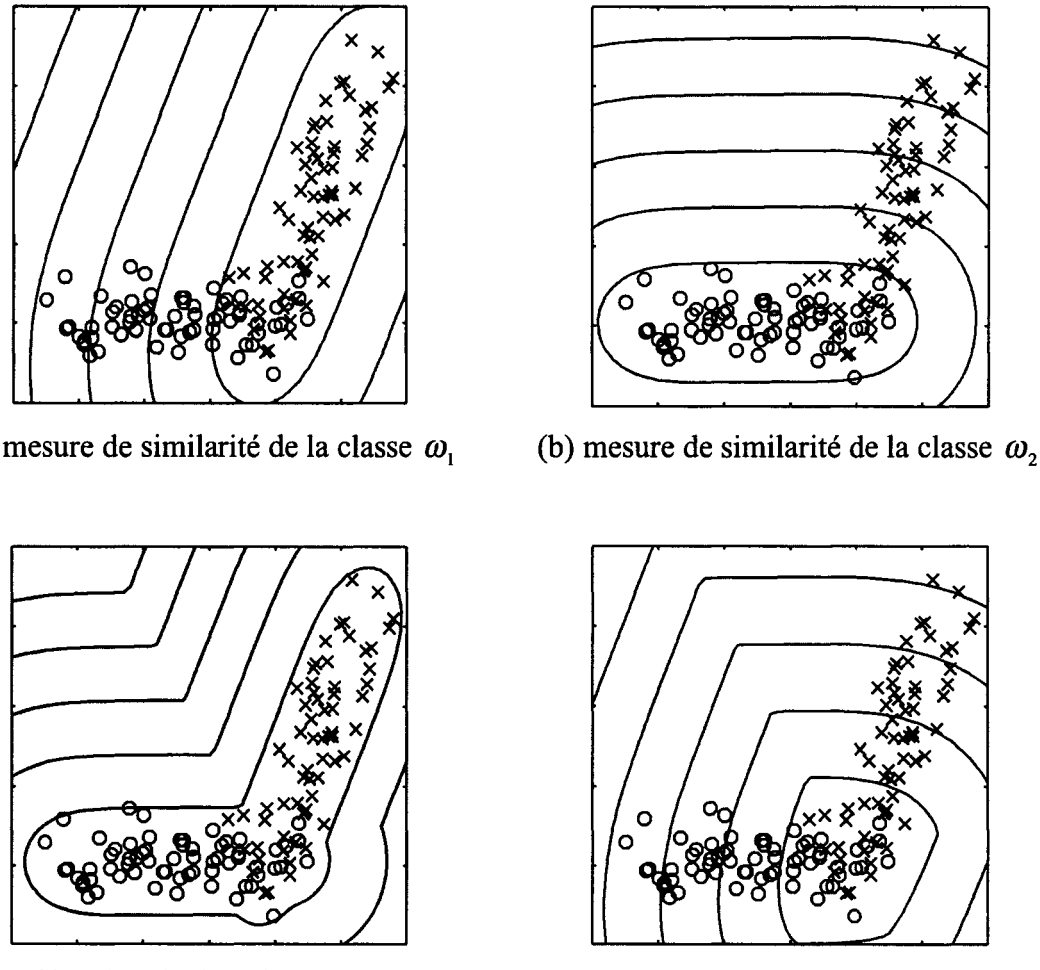
5.2 Description de l'approche proposée

L'approche que nous proposons dans ce chapitre consiste donc à combiner SVM et approche par modélisation au sein d'une architecture modulaire à deux niveaux de décision. Le MLP qui était utilisé dans le chapitre précédent est alors remplacé par une approche par modélisation qui permettra, contrairement au MLP, de rejeter efficacement les données aberrantes. De plus, la modularité de ce type d'approche permettra de traiter des problèmes de classification où le nombre de classes est important.

Ainsi, bien que généralement peu discriminante, une approche par modélisation peut tout de même permettre de caractériser efficacement le problème de reconnaissance. Trois cas de figure sont alors envisageables :

- Aucune des mesures de similarité n'est significative. Il s'agit alors d'une donnée aberrante qu'il est préférable de rejeter.
- Une seule mesure de similarité est significative. La décision peut donc être prise directement par l'approche par modélisation.
- Plusieurs mesures de similarité sont significatives. Il s'agit alors d'une donnée ambiguë et il est donc préférable d'utiliser des SVM pour prendre la décision.

Un exemple en deux dimensions est présenté à la figure 44. Les mesures de similarité correspondant aux modèles des deux classes sont représentées par des lignes de niveaux en (a) et (b), alors que la combinaison de ces deux mesures montre en (c) comment il est possible de détecter les données aberrantes en utilisant le minimum des deux mesures et en (d) comment isoler les données ambiguës en utilisant le maximum des deux mesures.



(a) mesure de similarité de la classe ω_1 (b) mesure de similarité de la classe ω_2

(c) détection de données aberrantes (d) détection de données ambiguës

Figure 44 Caractérisation du problème à l'aide des mesures de similarité

Concernant le choix de l'approche par modélisation, un de nos objectifs étant toujours d'accélérer la prise de décision, nous opterons pour une approche de faible complexité.

Ainsi, partant de l'hypothèse que la distribution des données de chacune des classes est unimodale, nous avons choisi de modéliser chaque classe ω_i à l'aide d'un sous-espace vectoriel défini par la moyenne μ_i des données de la classe et la matrice Ψ_i qui contient les k premiers vecteurs propres ϕ_i^j extraits de la matrice de covariance. Notons que les ϕ_i^j sont des vecteurs colonnes alors que nos exemples, ainsi que les moyennes μ_i, sont représentés par des vecteurs lignes.

La mesure de similarité utilisée (ou plus exactement de dissimilarité) est alors la distance de projection sur le sous-espace :

$$d_i(x) = \|x - f_i(x)\| \tag{5.1}$$

Pour tout point x de l'espace de représentation, le degré d'appartenance à une classe ω_i peut alors être évalué en calculant la distance d_i entre le point x considéré et sa projection sur le sous-espace qui modélise la classe ω_i :

$$f_i(x) = (x - \mu_i)\Psi_i\Psi_i^T + \mu_i \tag{5.2}$$

Par ailleurs, afin de réduire la complexité de calcul, notons qu'il est possible de calculer directement la distance de projection d_i sans avoir à calculer $f_i(x)$:

$$d_i(x) = \|x - \mu_i\|^2 - \sum_{j=1}^{k} \{(x - \mu_i)\phi_i^j\}^2 \tag{5.3}$$

Cette approche ne nécessite donc que l'optimisation du paramètre k correspondant à la dimension des sous-espaces vectoriels. Toutefois, comme nous le verrons dans la prochaine section, il est important de ne pas négliger ce paramètre qui joue un rôle important dans la qualité de la modélisation. En effet, si k est trop petit, la perte d'information est importante et la modélisation peu précise. Si l'on considère le cas extrême où $k=0$, une classe ω_i n'est alors modélisée que par le prototype μ_i correspondant à la moyenne des données de la classe. Par contre, si k est trop grand, les modèles engendrés ne sont plus discriminants. Si l'on considère l'autre cas extrême où $k=d$, d étant le nombre de caractéristiques, une classe ω_i n'est alors plus modélisée par un sous-espace vectoriel mais par un espace de même dimension que l'espace de représentation. Alors, quel que soit le point x, la distance de projection est nulle.

Un exemple de modélisation à l'aide de sous-espaces est illustré à la figure 45. Les données de chaque classe sont alors modélisées par leur axe principal ($k=1$). La donnée inconnue x est donc projetée en $f_1(x)$ et $f_2(x)$. Ainsi, la distance de projection $d_1(x)=\|x-f_1(x)\|$ étant nettement inférieure à la distance de projection $d_2(x)=\|x-f_2(x)\|$, nous pouvons en conclure que la donnée x appartient à la classe ω_1.

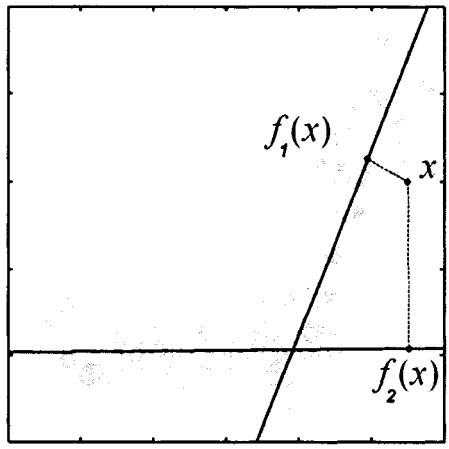


Figure 45 Exemple de projection sur les sous-espaces modélisant les deux classes

Les distances de projection peuvent donc être utilisées au premier niveau de décision pour classer les données ne présentant aucune ambiguïté, mais aussi pour détecter d'éventuelles données aberrantes et effectuer alors un rejet d'ignorance. En effet, si toutes les distances de projection sont supérieures à un certain seuil, il est raisonnable de penser que la donnée n'appartient à aucune des classes du problème. Notons toutefois que le fait que les sous-espaces ne soient pas bornés risque de s'avérer problématique. En effet, ceci implique que les frontières de décision correspondant au rejet d'ignorance ne se referment pas autour des classes comme dans le cas de la figure 44-c et donc que certaines données aberrantes peuvent très bien avoir une distance de projection faible. Néanmoins, ce problème est généralement limité par le fait que les valeurs des caractéristiques sont bornées et que la dimension des sous-espaces est beaucoup plus faible que le nombre de caractéristiques. Ainsi, comme nous le verrons dans la section expérimentale, cette méthode très simple, que nous dénommerons SSC (Sub-Space Classifier), permet tout de même d'obtenir des résultats acceptables et de valider notre proposition.

D'autre part, si au moins une des distances de projection est inférieure au seuil de rejet, il est raisonnable de penser que la donnée appartient bien à l'une des classes du problème. Il est alors possible d'estimer, à ce premier niveau du système, des probabilités *a posteriori* d'appartenance aux différentes classes. De ce fait, ayant observé que les distributions des distances de projection correspondant aux exemples ambigus suivent un comportement exponentiel, nous avons choisi d'utiliser une fonction softmax pour estimer les probabilités *a posteriori* :

$$\hat{P}_{SSC}(\omega_i \mid x) = \frac{\exp\left(-\alpha d_i(x)\right)}{\sum_{j=1}^{c} \exp\left(-\alpha d_j(x)\right)} \tag{5.4}$$

Notons qu'il s'agit d'une version simplifiée de la fonction softmax ne comprenant qu'un seul paramètre α. Bien entendu, l'utilisation d'un seul paramètre n'est pas optimale, mais étant donné que les probabilités les plus fortes sont ré-estimées à l'aide des SVM du second niveau de décision, ceci n'aura probablement aucun effet sur les résultats du système complet.

Concernant le second niveau de décision, nous avons choisi d'adopter le schéma de combinaison TopN-OAO présenté dans le chapitre précédent. L'objectif est alors de privilégier la modularité du système, de manière à pouvoir par exemple traiter des problèmes pour lesquels le nombre de classes est très grand. En effet, lorsque le nombre de classe augmente, la complexité liée à l'apprentissage des SVM peut rapidement devenir un problème. Il est alors possible d'utiliser l'approche par modélisation pour déterminer les paires de classes qui sont impliquées dans la majorité des conflits et d'entraîner ainsi qu'un nombre réduit de SVM « un contre un ». De plus la modularité de cette approche permettra de pouvoir utiliser le système dans des environnements dynamiques où il est nécessaire de pouvoir ajouter ou retirer facilement des classes.

Notons par ailleurs que la formule de ré-estimation des probabilités est légèrement différente de celle utilisée dans le chapitre précédent :

$$\hat{P}_{SVM}(\omega_{\ell(i)} \mid x) = \frac{\prod_{j=1, j\neq i}^{N} \hat{P}\left(\omega_{\ell(i)} \mid f_{\ell(i),\ell(j)}(x)\right)}{\sum_{k=1}^{N} \prod_{j=1, j\neq k}^{N} \hat{P}\left(\omega_{\ell(k)} \mid f_{\ell(k),\ell(j)}(x)\right)} \times \left(1 - \sum_{i=N+1}^{c} \hat{P}_{SSC}(\omega_{\ell(i)} \mid x)\right) \tag{5.5}$$

En effet, ces expériences ayant été réalisées avant celles du chapitre 4, nous nous basions alors sur la méthode de combinaison proposée par Hamamura *et al.* [15].

Une vue d'ensemble du système proposé dans ce chapitre est présentée à la figure 46. Comme nous pouvons le constater, cette approche permet donc de dissocier les deux

types de rejet. L'approche par modélisation utilisée au premier niveau de décision permet alors de rejeter efficacement les données aberrantes, tandis qu'au second niveau de décision les SVM permettent de rejeter les données ambiguës.

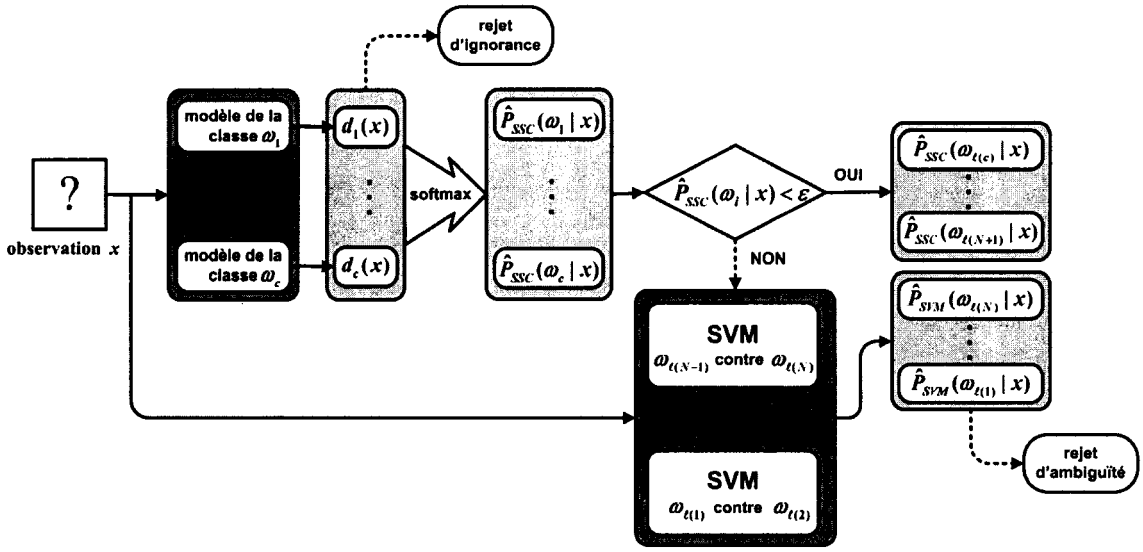


Figure 46 Vue d'ensemble de la combinaison avec une approche par modélisation

5.3 Résultats expérimentaux

L'approche proposée dans ce chapitre n'a pas été testée sur les même bases de données que précédement. La base de données alors utilisée est la base MNIST[2] (Modified NIST), qui est constituée d'images de chiffres manuscrits normalisées en dimension (20×20) puis centrées dans une rétine 28×28 en faisant coïncider le centre de gravité du caractère avec le centre géométrique de la rétine. Quelques exemples de rétines sont présentés à la figure 47.

---

[2] disponible à l'adresse suivante : http://yann.lecun.com/exdb/mnist/

La base d'apprentissage est formée de 60 000 images et la base de test de 10 000 données indépendantes. Pour nos expériences, nous avons scindé la base d'apprentissage en deux corpus. Les 50 000 premiers exemples sont utilisés pour l'entraînement des classifieurs et les 10 000 suivants forment une base de validation, qui servira à l'optimisation des hyper-paramètres des classifieurs.

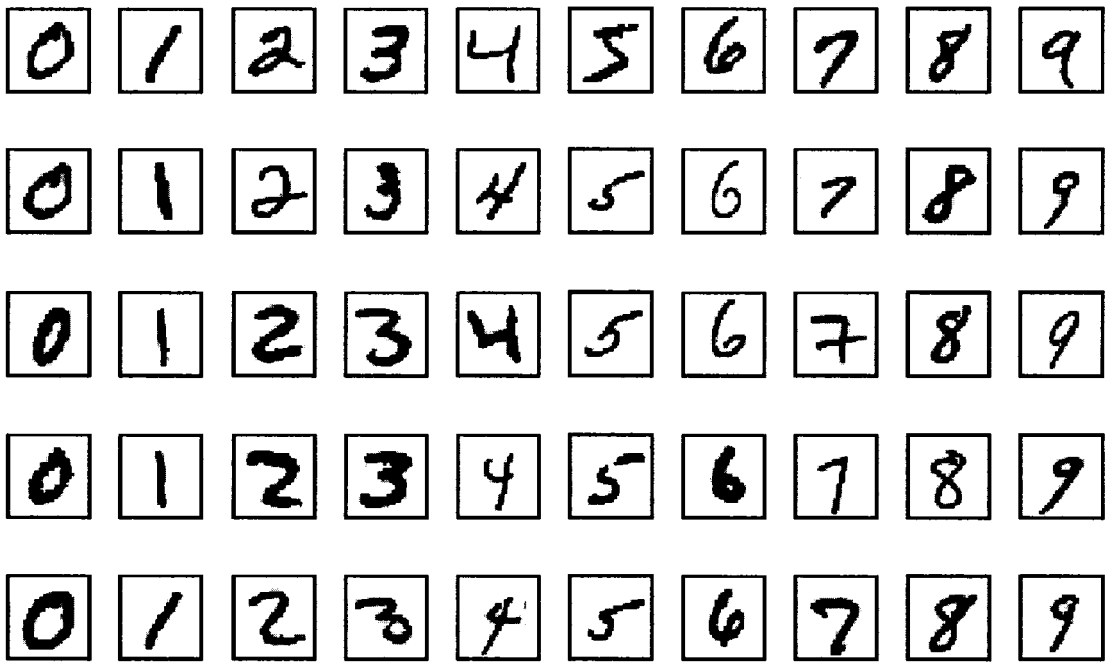


Figure 47 Exemples d'images de la base MNIST

Il s'agit donc d'une base publique qui est couramment utilisée pour évaluer les performances d'une nouvelle approche de classification. En effet, les résultats obtenus avec la plupart des approches classiques ont été reportés par Le Cun *et al.* [73], ou plus récemment par Liu *et al.* [1]. Comme nous pouvons le constater au tableau XXXIV, les SVM s'avèrent une nouvelle fois plus discriminants que les autres classifieurs, mais aussi beaucoup moins rapide que des réseaux de neurones tels que le MLP ou le RBF.

Tableau XXXIV

Résultats reportés par Liu *et al.* [1] sur la base MNIST

	k-NN	RBF	MLP	SVM
Taux d'erreur en test (%)	3,66	2,53	1,91	**1,41**
Temps de classification (ms)	98,68	1,03	**0,80**	62,8

Notons qu'il est donc possible d'obtenir de meilleurs résultats en exploitant des connaissances explicites sur la nature des données. Ainsi, en termes de taux d'erreur, le meilleur résultat obtenu jusqu'à présent est de 0,4%. Pour ce faire, Simard *et al.* [74] ont utilisé un réseau de neurones à convolution dont l'architecture exploite le fait que les éléments constituant les vecteurs d'entrées ne sont pas indépendants, mais résultent d'une structure spatiale. Les deux premières couches de leur réseau peuvent donc être vues comme un extracteur de caractéristiques, dont les paramètres seraient obtenus par apprentissage. De plus, ils génèrent artificiellement des données d'apprentissage supplémentaires en déformant les données existantes, ce qui permet d'améliorer la capacité à généraliser de leur classifieur. Par ailleurs, ce taux d'erreur de 0,4% a aussi été obtenu par Liu *et al.* [1] en utilisant un opérateur de gradient de type Sobel, afin d'extraire des rétines un ensemble de 200 caractéristiques discriminantes, qui sont ensuite exploitées par des SVM.

Ainsi, bien que l'utilisation d'une procédure d'extraction de caractéristiques permette d'améliorer significativement les performances, nous avons choisi d'utiliser les données brutes, ce qui nous permettra notamment de visualiser les projections sur les différents sous-espaces vectoriels en reconstituant les rétines, mais surtout de comparer nos résultats avec ceux reportés dans la littérature.

Dans un premier temps, il est nécessaire de fixer la dimension k des sous-espaces vectoriels. La base de validation a alors été utilisée pour estimer l'effet de k sur les performances en classification. Ainsi, comme il est possible de le constater à la figure 48, ce paramètre est très influent.

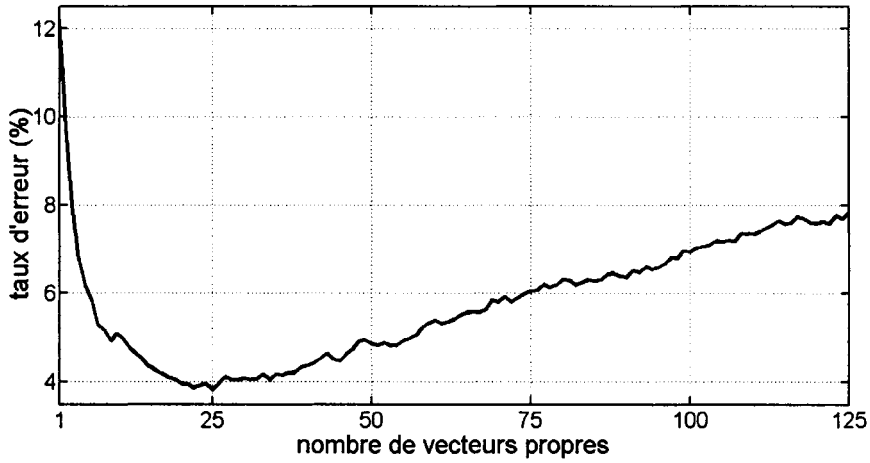


Figure 48 Effet de la dimension des sous-espaces vectoriels

Nous avons donc choisi d'utiliser les 25 premiers vecteurs propres pour modéliser indépendamment chacune des classes. Le taux d'erreur obtenu sur la base de test est alors de 4,09%. Ce résultat est donc nettement moins bon que ceux obtenus avec des approches par séparation tels qu'un MLP ou des SVM, mais est comparable à celui obtenu avec un k-NN qui est aussi une approche par modélisation. De plus, l'utilisation de sous-espaces s'avère beaucoup plus rapide lors de la prise de décision.

Dans un second temps, il est nécessaire d'optimiser le paramètre α de la fonction softmax qui est utilisé pour estimer les probabilités *a posteriori* à partir des distances de projection. Pour ce faire, la valeur de α a été discrétisée entre 1 et 10 par pas de 0,1 et pour chacune de ces valeurs, la NLL a été évaluée sur la base de validation. Finalement,

la NLL la plus faible a été obtenue avec $\alpha = 5,6$. Les résultats en termes de rejet d'ambiguïté sont reportés à la figure 49. Comme nous pouvons le constater, cette fonction softmax simplifiée permet déjà d'améliorer fortement le compromis erreur-rejet de l'approche par modélisation. En effet, nous pouvons par exemple constater que si l'on exploite directement la distance de projection la plus faible, il est nécessaire de rejeter environ 30% des données de la base de validation pour obtenir 1% d'erreur parmi les exemples restants ; alors qu'en exploitant les probabilités estimées avec la fonction softmax, il n'est nécessaire d'en rejeter que 8% pour atteindre le même taux d'erreur.

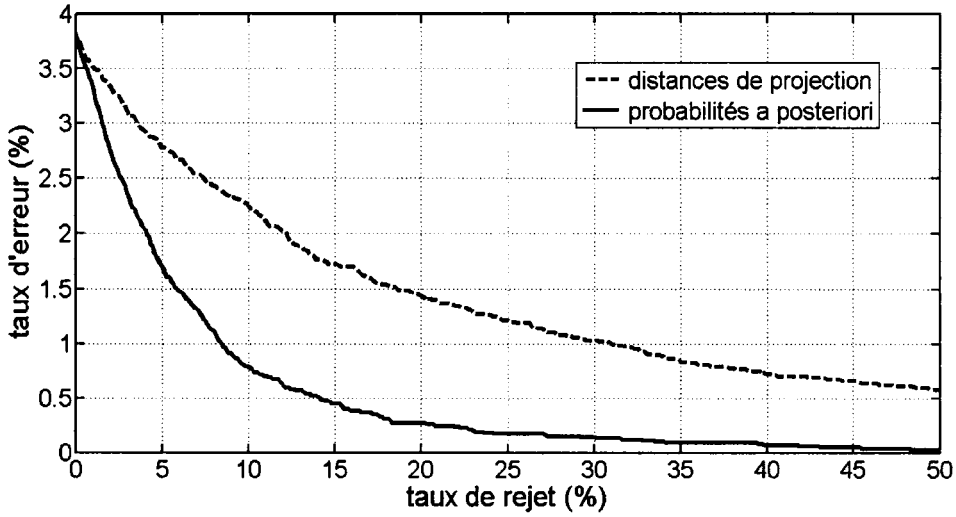


Figure 49 Capacité au rejet d'ambiguïté de l'approche par modélisation

Ainsi, bien que peu discriminante, cette approche par modélisation permet tout de même de classer environ la moitié des données, avec un risque d'erreur proche de zéro. Un exemple de donnée pouvant être considérée comme non-ambigüe est présenté à la figure 50. Nous pouvons alors constater que seule l'image correspondant à la projection sur le sous-espace modélisant les images du chiffre 6 est très proche de l'exemple à classer.

Par conséquent, la distance de projection correspondant à cette classe est relativement faible, la donnée ne sera donc pas considérée comme étant aberrante. De plus, nous obtenons alors une seule probabilité *a posteriori* proche de un et toutes les autres proches de zéro. De ce fait, si le seuil de tolérance est fixé par exemple à 10^3, le système n'utilisera alors pas les SVM pour ré-estimer les probabilités.

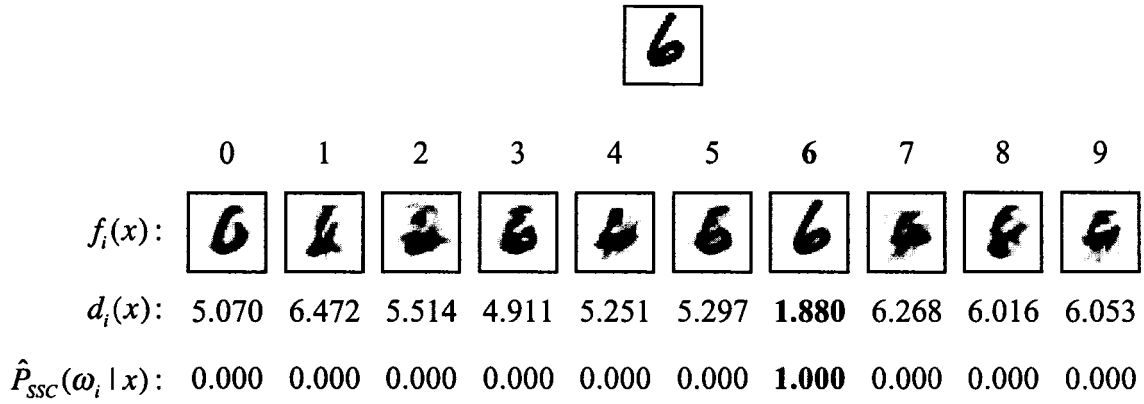


Figure 50 Résultats obtenus avec une donnée de test non-ambiguë

Concernant les SVM, les hyper-paramètres ont été fixés en cherchant à minimiser le taux d'erreur sur la base de validation. Les valeurs finalement utilisées pour l'apprentissage des différents SVM sont $C = 10$ et $\gamma = 0{,}0185$. Ensuite, de manière à obtenir des résultats de référence, nous avons évalué sur la base de test les résultats obtenus lors de l'utilisation de l'ensemble des SVM de la stratégie « un contre un », mais aussi de la stratégie « un contre tous ». Les résultats sont reportés au tableau XXXV. Notons que dans le cas de la stratégie « un contre un », les sorties sont calibrées à l'aide de sigmoïdes, puis combinées selon la méthode proposée par Hamamura *et al.* [15] de manière à estimer des probabilités a posteriori. De même, dans le cas de la stratégie « un contre tous », les sorties sont calibrées à l'aide d'une fonction softmax.

Tableau XXXV

Résultats obtenus avec les SVM sur la base MNIST

	Taux d'erreur sur la base de test	Nombre de vecteurs de support
SVM-OAO	1,48 %	11 118
SVM-OAA	1,41 %	13 743

Nous pouvons donc constater, que la stratégie « un contre tous » est une nouvelle fois légèrement plus précise que la stratégie « un contre un ». À titre indicatif, nous avons reporté à la figure 51 le compromis erreur-rejet obtenu avec l'approche par modélisation et celui obtenu avec les SVM de la stratégie « un contre tous ». Nous pouvons donc constater que l'approche par modélisation s'avère effectivement bien moins adaptée au rejet d'ambiguïté qu'une approche par séparation. Ce premier résultat justifie donc en partie le fait de vouloir combiner les deux types d'approche de classification.

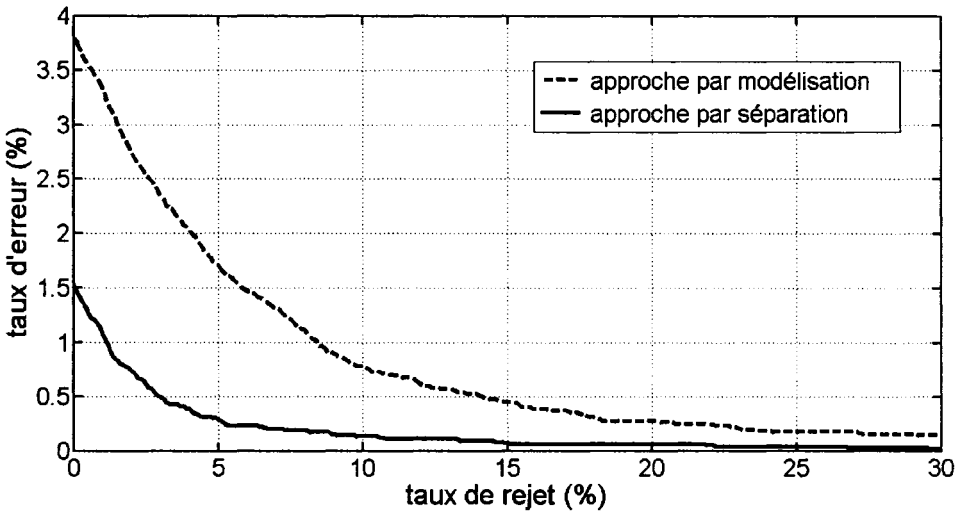


Figure 51 Comparaison des deux types d'approches pour le rejet d'ambiguïté

Par la suite, nous avons donc évalué sur la base de validation, l'effet du seuil de tolérance sur le compromis entre complexité et précision. Comme dans le chapitre précédent, nous pouvons constater à la figure 52 que l'utilisation d'une valeur de seuil de 10^{-3} permet d'obtenir le même taux d'erreur en validation que celui obtenu avec l'ensemble des SVM de la stratégie « un contre un », à savoir 1,53%. De plus, le système n'utilise alors en moyenne que 1 120 vecteurs de support, soit environ dix fois moins que lors de l'utilisation systématique des 45 SVM.

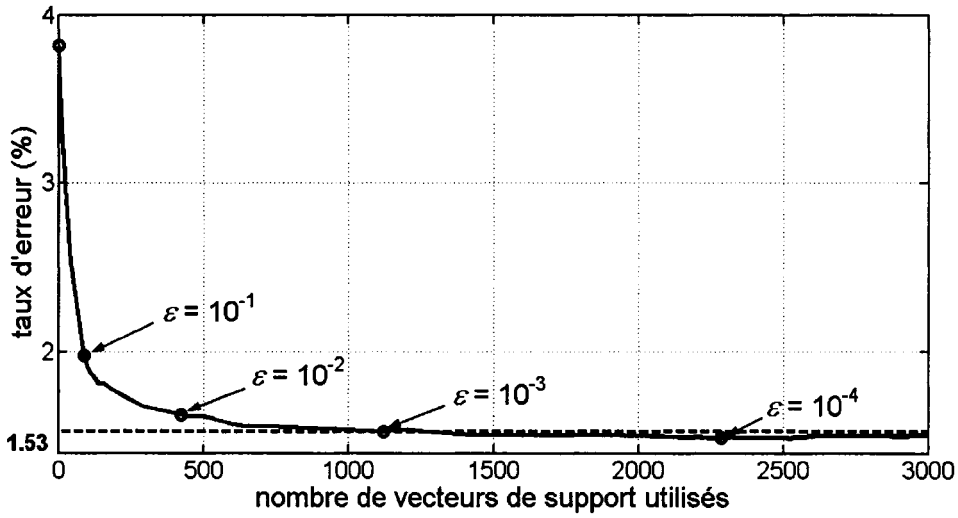


Figure 52 Compromis entre précision et complexité

D'autre part, nous avons évalué sur la base de validation le comportement du système concernant le rejet d'ambiguïté. Les résultats obtenus avec différentes valeurs de seuil sont reportés à la figure 53. Nous pouvons y constater que l'utilisation d'une valeur de seuil de 10^{-3} permet d'obtenir des performances très proches de celles obtenues avec les 45 SVM.

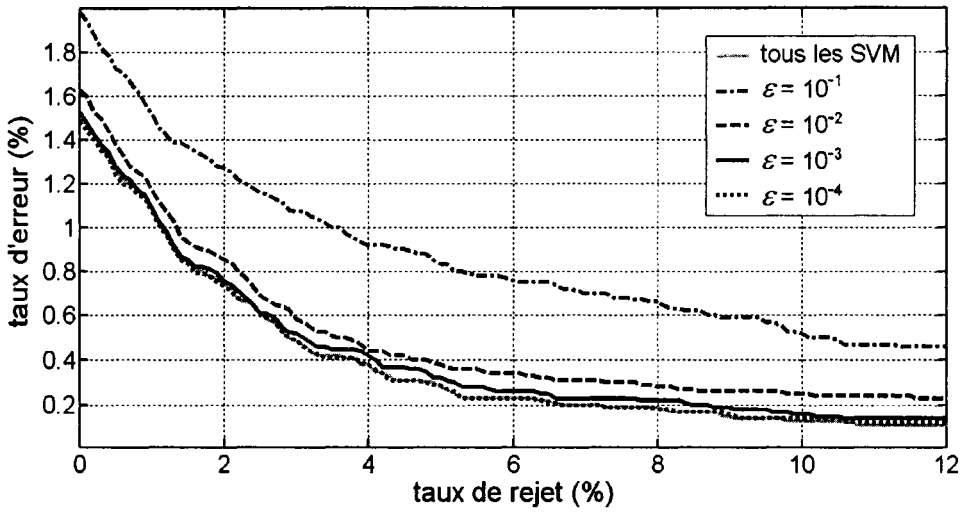


Figure 53 Effet du seuil de tolérance sur le rejet d'ambiguïté

Concernant la complexité, nous avons évalué sur la base de validation la répartition du nombre de SVM utilisés par le système pour différentes valeurs de seuil. Les résultats obtenus sont reportés à la figure 54. Il est alors intéressant de constater que pour $\varepsilon = 10^{-3}$, la moitié des données sont traitées directement au premier niveau, ce qui recoupe le commentaire concernant la figure 49, à savoir que l'approche par modélisation permet de classer environ la moitié des données, avec un risque d'erreur proche de zéro.

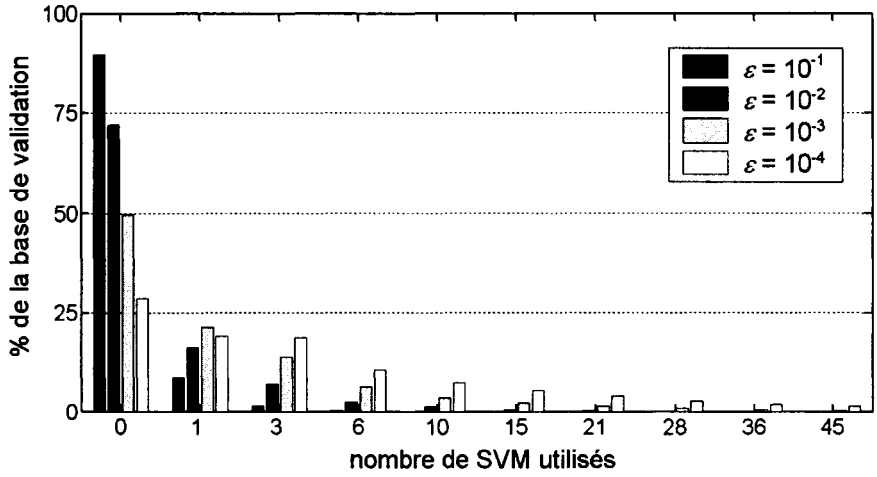


Figure 54 Répartition du nombre de SVM utilisés en fonction du seuil de tolérance

Finalement, les résultats obtenus sur la base de test en utilisant une valeur de seuil de 10^{-3} sont reportés au tableau XXXVI. Nous pouvons y constater que le schéma de combinaison proposé permet d'obtenir des résultats très proches en termes de précision de ceux obtenus avec les 45 SVM de la stratégie « un contre un », tout en limitant fortement la complexité liée à la prise de décision.

Tableau XXXVI

Résultats obtenus sur la base de test avec $\varepsilon = 10^{-3}$

	Taux d'erreur (%) *- 0% de rejet -*	Taux de rejet (%) *- 0,1% d'erreur -*	MFLOP[3]
SSC	4,09	28,59	0,4
SSC + SVM-OAO	**1,50**	**9,85**	**3,0**
SVM-OAO	1,48	9,55	26,2
SVM-OAA	1,41	6,32	32,4

[3] 1 MFLOP = 1000 KFLOP

À titre indicatif, les 150 erreurs de classification commises par le système sont reportées à la figure 55. Nous pouvons alors constater que certaines de ces données peuvent quasiment êtres considérées comme étant des données aberrantes et pourraient donc certainement être rejetées au premier niveau. En effet, rappelons que jusqu'à présent aucun rejet d'ignorance n'est effectué par le système.

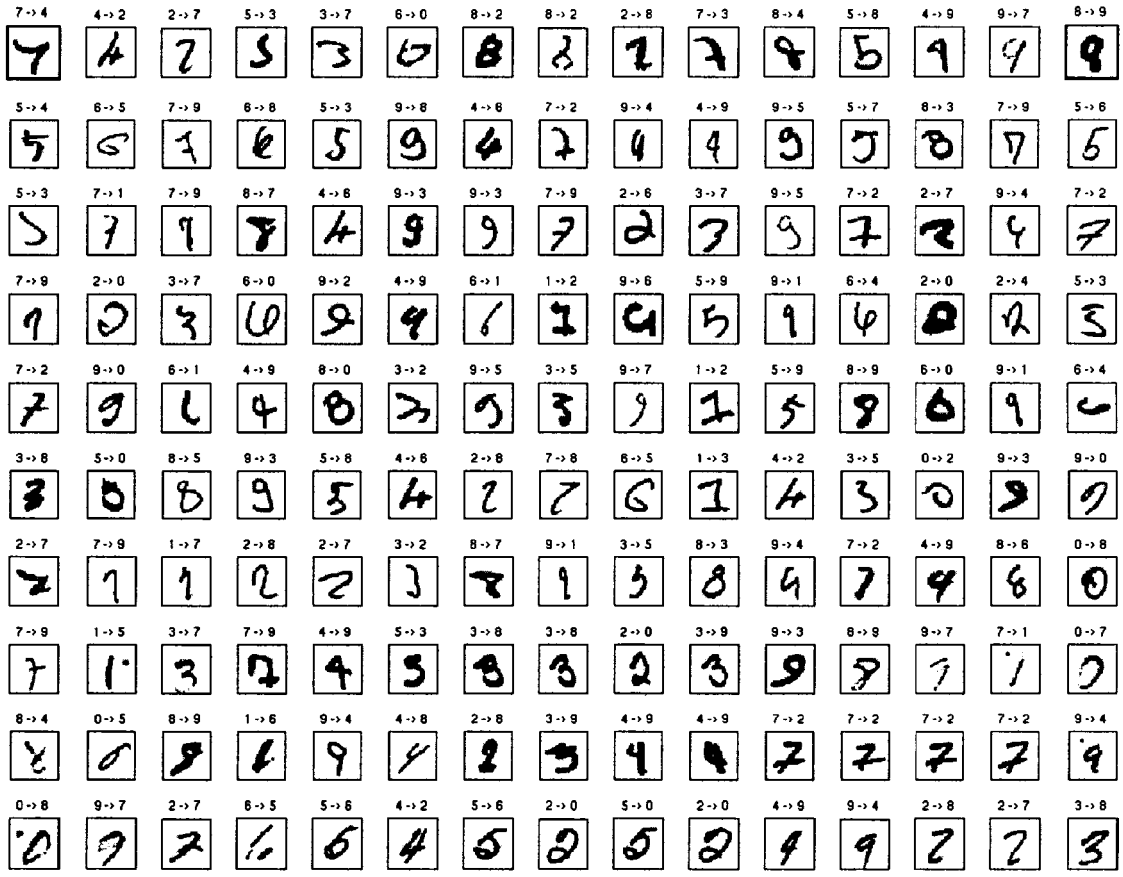


Figure 55 Erreurs de classification commise par notre système (classe –> décision)

Un exemple de données ayant nécessité l'utilisation de trois SVM est présenté à la figure 56. Dans ce cas, nous pouvons remarquer que l'approche par modélisation ne permet pas de classer correctement la donnée, mais que l'utilisation des trois SVM, activés par le premier niveau, permettra de prendre la bonne décision au second niveau.

	0	1	2	3	4	5	6	7	8	9
$f_i(x)$:										
$d_i(x)$:	4.932	5.366	4.869	4.802	**3.241**	5.146	5.446	**3.503**	4.780	**3.473**
$\hat{P}_{SSC}(\omega_i \mid x)$:	0.000	0.000	0.000	0.000	**0.664**	0.000	0.000	**0.153**	0.000	**0.182**
$\hat{P}_{SVM}(\omega_i \mid x)$:	0.000	0.000	0.000	0.000	**0.000**	0.000	0.000	**0.997**	0.000	**0.003**

Figure 56 Résultats obtenus avec une donnée de test ambiguë ($\varepsilon = 10^{-3}$)

Afin d'évaluer la capacité au rejet d'ignorance des différents classifieurs, une base de données aberrantes (« outliers ») a été générée artificiellement. Pour ce faire, nous avons assemblé chaque donnée de la base de test avec la donnée suivante, comme illustré à la figure 57. D'un point de vue pratique, nous avons extrait les rétines de 20×20 des rétines de 28×28, nous avons ensuite superposé les deux images de manière à ce que les deux chiffres soient en contact, puis nous avons redimensionné l'image résultante de manière à obtenir une nouvelle rétine de 20×20 qui sera centrée dans une rétine de 28×28 en faisant coïncider le centre de gravité avec le centre géométrique de la rétine.

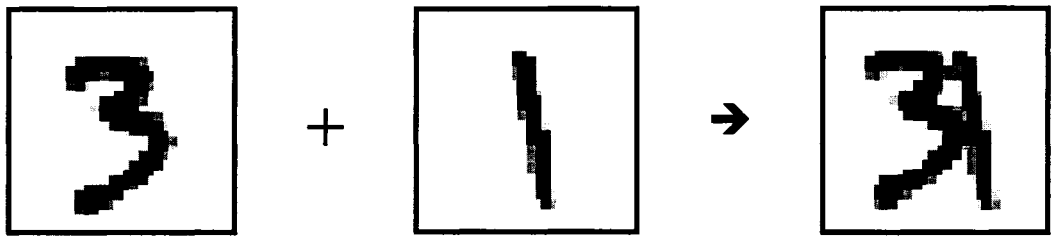


Figure 57 Illustration de la procédure de création de données aberrantes

Bien que cette procédure ne soit pas idéale, conduisant parfois à l'obtention de données très proches de chiffres réels, elle présente cependant l'avantage de permettre de générer une grande diversité de formes. Les données aberrantes générées ainsi correspondent de plus à un problème réel de sous-segmentation. À titre indicatif, les 100 premières images de la base de test et les 100 « outliers » correspondants sont présentés à la figure 60. De plus, cette procédure est facilement reproductible, ce qui offre la possibilité à d'autres chercheurs de comparer leurs résultats sur les mêmes données.

Un exemple de résultat obtenu avec notre approche par modélisation pour l'une des données aberrantes est présenté à la figure 58. Nous pouvons y constater que la plus petite distance de projection (5,083) est bien plus grande que celles observées précédemment à la figure 50 (1,880) ou à la figure 56 (3,241).

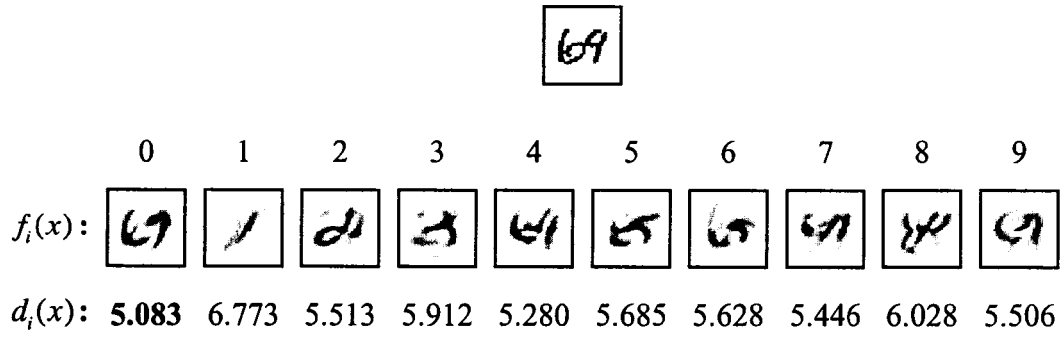


Figure 58 Résultats obtenus avec une donnée aberrante

Or, si nous observons l'histogramme des valeurs de la distance de projection la plus faible, présenté à la figure 59, nous pouvons constater que la valeur du seuil de rejet d'ignorance pourrait être fixée aux alentours de 4,5. La donnée de l'exemple présenté à la figure 58 serait alors rejetée (5,083 > 4,5), tandis que celles des exemples présentés aux figures 50 et 56 ne le seraient pas (1,880 < 4,5 et 3,241 < 4,5).

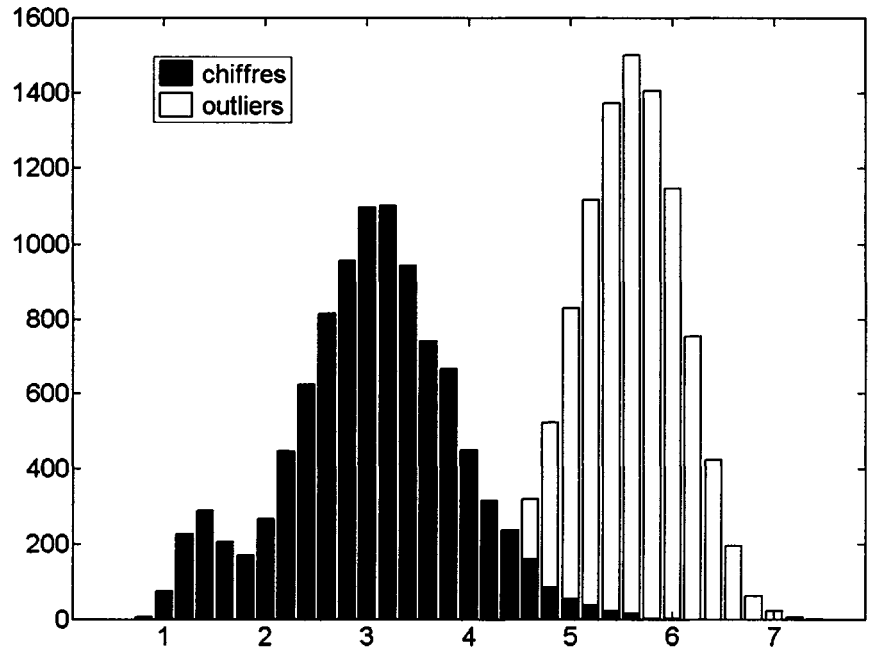
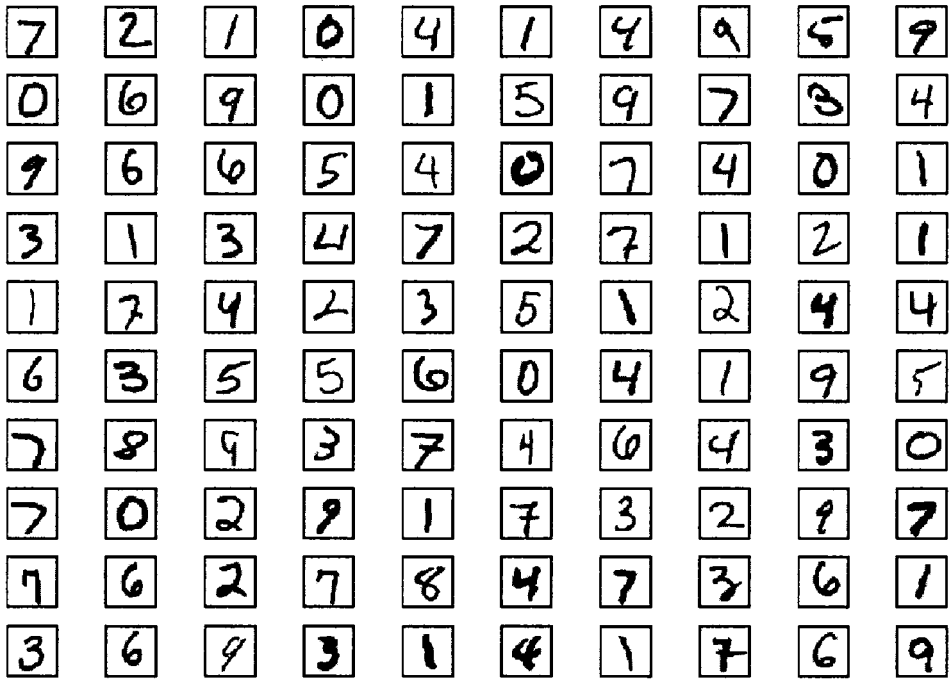
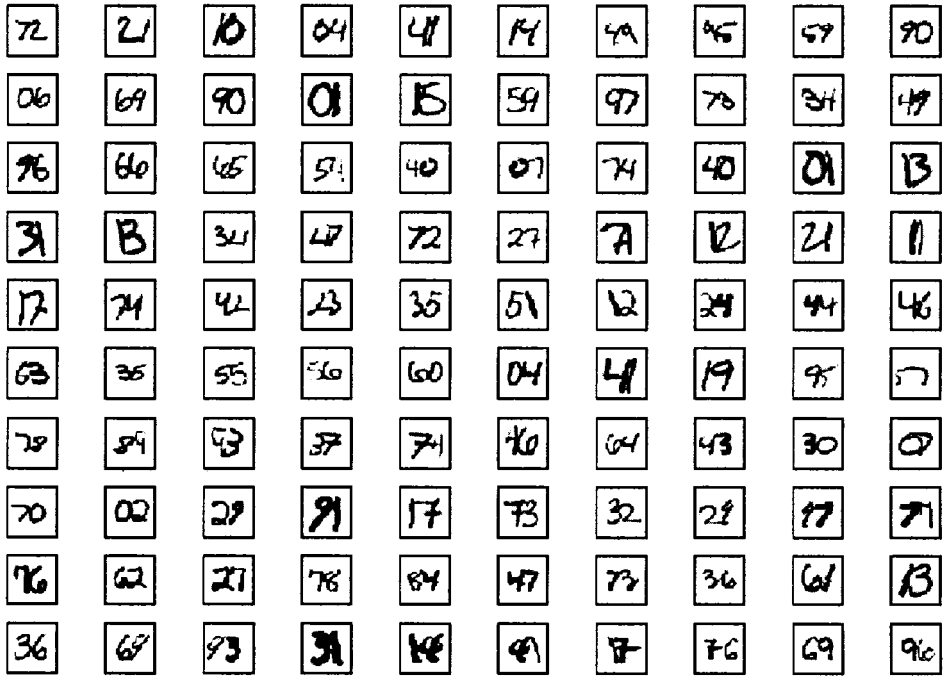


Figure 59 Histogramme des valeurs de la distance de projection la plus faible

D'autre part, notons qu'il n'est pas étonnant que les deux distributions se chevauchent. En effet, au-delà du fait que notre approche par modélisation pourrait être améliorée, notre procédure de création de données aberrantes n'est pas parfaite et conduit parfois à l'obtention de données très proches de chiffres réels. De même, comme nous l'avons déjà mentionné, certains chiffres de notre base de données peuvent quasiment être considérées comme étant des données aberrantes.



(a)



(b)

Figure 60 Les 100 premiers chiffres (a) et les 100 « outliers » correspondants (b)

Nous avons alors cherché à comparer la capacité au rejet d'ignorance de notre approche basée sur l'utilisation de sous-espaces vectoriels (SSC) avec une approche par modélisation plus classique, en l'occurrence un plus-proche-voisin (1NN). Pour ce faire, nous avons fait varier le seuil de rejet d'ignorance et pour chaque valeur nous avons mesuré parmi les 10 000 chiffres la quantité qui est rejetée et parmi les 10 000 outliers la quantité qui est acceptée. Ainsi, comme nous pouvons le constater à la figure 61, les deux approches sont tout à fait comparables en termes de rejet d'ignorance. Les résultats obtenus avec les sous-espaces vectoriels sont même très légèrement meilleurs. Ce résultat confirme donc que le fait que les sous-espaces ne soient pas bornés n'est pas un problème majeur dans le cas de ce type de données.

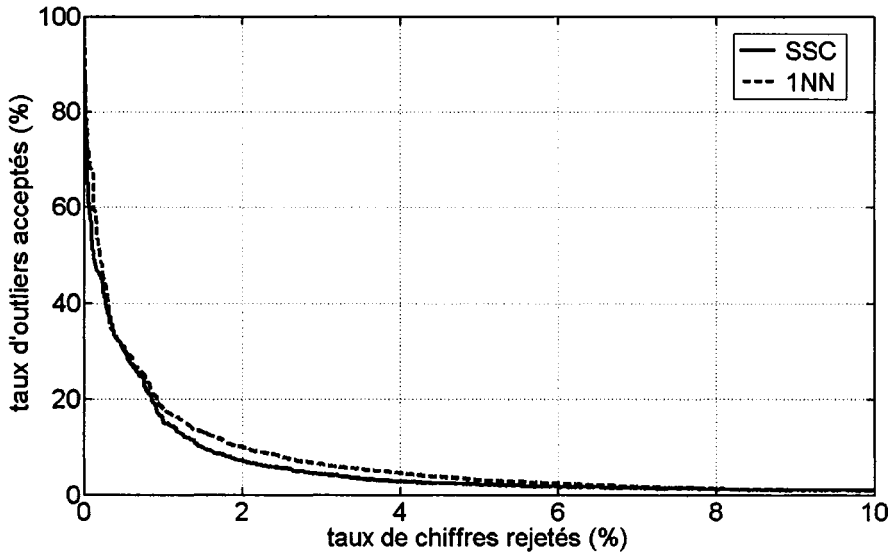


Figure 61 Capacité au rejet d'ignorance de l'approche par modélisation

Ensuite, nous avons comparé la capacité au rejet d'ignorance de notre approche par modélisation avec celle d'une approche par séparation, en l'occurrence les SVM de la stratégie « un contre tous ». Les résultats obtenus avec les deux approches sont reportés à la figure 62.

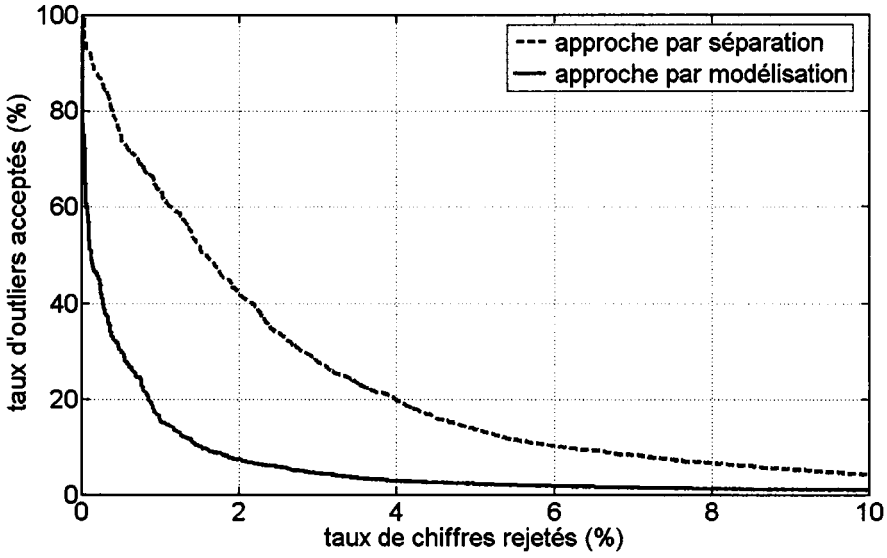


Figure 62 Comparaison des deux types d'approches pour le rejet d'ignorance

On peut alors constater que l'approche par modélisation est beaucoup plus efficace que l'approche par séparation. Par exemple, pour filtrer 95 % des données aberrantes, l'approche par modélisation ne rejette que 2,8 % des chiffres, alors que l'approche par séparation en rejette 9,3 %, soit environ trois fois plus.

Pour finir, nous avons évalué l'effet du rejet d'ignorance sur le rejet d'ambiguïté. Le seuil de rejet d'ignorance a alors été fixé à 4,63 qui est la valeur ayant permis de filtrer 95 % des données aberrantes. Puis, nous avons fait varier la valeur du seuil de rejet d'ambiguïté. Les résultats obtenus sont présentés à la figure 63. Nous pouvons alors constater qu'en l'absence de rejet d'ambiguïté, les 2,8 % de chiffres rejetés au premier niveau ne permettent de faire diminuer le taux d'erreur du système que de 0,26 % (1,24 % au lieu de 1,5 %). Par contre, lorsque les deux types de rejet sont utilisés, le taux de rejet total nécessaire pour atteindre des taux d'erreur très faibles est très proche des valeurs obtenues par le système en l'absence de rejet d'ignorance.

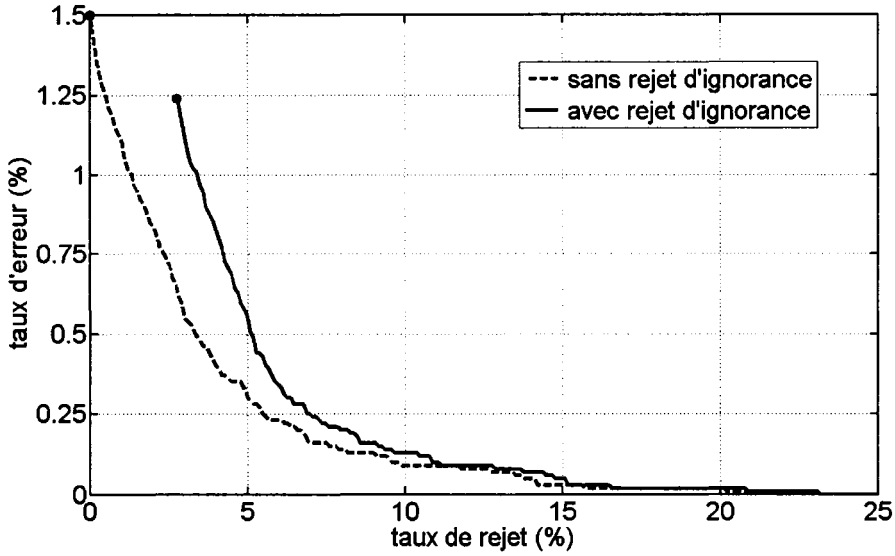


Figure 63 Effet du rejet d'ignorance sur le rejet d'ambiguïté

Nous pouvons donc en conclure que l'ajout d'une option de rejet d'ignorance au premier niveau du système ne semble pas détériorer significativement les performances globales du système en termes de rejet.

5.4 Conclusion

Les résultats expérimentaux que nous avons obtenus permettent donc de valider le concept proposé dans ce dernier chapitre. En effet, nous avons pu vérifier que si les approches par modélisation sont bien moins efficaces que les SVM pour traiter les données ambiguës, ce type d'approche s'avère par contre bien plus approprié pour détecter des données aberrantes. Il est donc tout à fait justifié de combiner approche par modélisation et approche par séparation au sein d'un système à deux niveaux de décision. Ce type d'architecture permettra effectivement de pouvoir dissocier les deux types de rejet et ainsi d'exploiter au mieux les capacités des deux types d'approche de classification.

De plus, nous avons également montré qu'en choisissant une approche par modélisation adaptée, ce type d'architecture peut aussi permettre d'exploiter la capacité à généraliser des SVM tout en limitant la complexité liée à la prise de décision.

Enfin, notons que les résultats pourraient être améliorés en utilisant une approche par modélisation plus précise. En effet, il est important de rappeler que l'approche par modélisation que nous avons utilisée comporte certaines limitations. Ainsi, il pourrait par exemple être préférable de modéliser chaque classe à l'aide d'un réseau auto-associatif qui comme l'a montré Kimura *et al.* [75], permet de générer des modèles compacts et donc de mieux refermer les frontières de décision autour des données de chaque classe. De plus, en utilisant plus qu'une couche cachée, il est possible d'obtenir une modélisation non-linéaire des données. Toute fois, si la distribution des données de certaine classe est de nature multi-modale, il sera alors préférable d'utiliser plusieurs réseaux auto-associatifs pour modéliser chacune de ces classes.

CONCLUSION

Dans le cadre de cette thèse de doctorat, nous avons donc apporté un certain nombre d'éléments de réponse concernant certaines questions liées à l'intégration des SVM au sein de systèmes de reconnaissance de formes. Notons que nous parlons bien d'éléments de réponse et non de réponses, car comme nous le rappellent Duda *et al.* [76] en s'appuyant sur le « No Free Lunch Theorem », si une approche est supérieure à une autre pour un problème de classification donné, rien ne permet de prédire qu'elle le sera aussi pour un problème différent. Ainsi, le choix de l'approche à utiliser dépendra du nombre de classes, du nombre de données d'apprentissage, du type de distribution, mais aussi des contraintes liées à l'application visée.

Pour conclure ce manuscrit, nous proposons donc de faire un bilan des contributions qui ressortent de nos travaux de recherche et des perspectives futures qui en découlent.

Contributions

Partant de la constatation que la revue de la littérature ne permet pas de déterminer quelle approche est la mieux adaptée pour résoudre des problèmes multi-classes à l'aide de SVM, nous avons effectué notre propre comparaison expérimentale. Nous avons alors choisi de nous concentrer uniquement sur les deux approches les plus courantes que sont la stratégie «un contre un » et la stratégie « un contre tous ». Il en est ressorti que la stratégie « un contre tous » semble légèrement plus précise que la stratégie « un contre un », mais que la différence semble s'amoindrir lorsque le nombre de classes augmente. De plus, la stratégie « un contre un » s'avère extrêmement moins complexe en termes d'apprentissage et semble donc mieux adaptée à la résolution de problème comprenant un grand nombre de classes.

Nous avons ensuite repris cette comparaison en nous intéressant à l'estimation de probabilités *a posteriori*. Nous avons alors proposé un critère original pour comparer les différentes méthodes d'estimation, basé sur le compromis erreur-rejet. Cette mesure a notamment permis de faire ressortir des différences qu'un simple taux d'erreur ne permet pas de détecter. D'autre part, nous avons aussi proposé une nouvelle méthode basée sur l'utilisation de la fonction softmax pour calibrer les sorties des SVM issus de la stratégie « un contre tous », qui s'est avéré significativement plus efficace que la méthode classique qui utilise des fonctions sigmoïdes. Finalement, nous avons vérifié expérimentalement qu'il est possible d'estimer avec des SVM, des probabilités significativement plus précises qu'avec un MLP qui est couramment utilisé au sein de système de reconnaissance de formes. Cependant, nous avons aussi pu constater que ce gain en précision se fait aux dépens de la complexité liée à la prise de décision.

La seconde partie de notre travail a donc porté sur l'accélération de la prise de décision. Nous avons ainsi proposé deux schémas de combinaison, dont l'originalité est de faire varier dynamiquement le nombre de SVM à utiliser pour raffiner les probabilités estimées par un MLP. Nous avons alors pu vérifier expérimentalement que ces schémas permettent d'accélérer fortement la prise de décision sans dégrader la qualité des probabilités estimées. Le premier schéma de combinaison, qui est basé sur l'utilisation de SVM issus de la stratégie « un contre tous », s'avère parfaitement adapté aux problèmes avec peu de classes, comme la reconnaissance de chiffres; tandis que le second schéma de combinaison, qui est basé sur l'utilisation de SVM issus de la stratégie « un contre un », s'avère mieux adapté aux problèmes avec un nombre plus important de classes, comme la reconnaissance de lettres.

Enfin, la dernière partie de notre travail avait pour objectif l'élaboration d'une approche permettant de traiter efficacement à la fois les données ambiguës et les données aberrantes. Nous avons alors proposé de combiner approche par modélisation et approche par séparation au sein d'un système à deux niveaux de décision. Le schéma de

combinaison présenté permet ainsi de dissocier le rejet d'ambiguïté et le rejet d'ignorance. Nous avons alors montré expérimentalement que notre approche permet d'exploiter au mieux les capacités des deux types d'approche de classification.

Par ailleurs, notons que les travaux de recherche présentés dans ce manuscrit ont fait l'objet de plusieurs publications. Concernant la combinaison entre approche par modélisation et approche par séparation, un première communication a été présentée à ICPR'04 (International Conference on Pattern Recognition) [77], puis une version allongée a été publiée dans le journal ELCVIA (Electronic Letters on Computer Vision and Image Analysis) [78]. De plus, des versions en français de ces deux articles ont respectivement été présentée à CIFED'04 (Colloque International Francophone sur l'Ecrit et le Document) [79] et publiée dans le journal Traitement du Signal [80]. L'accélération de la prise de décision, a été l'objet d'une communication présentée à IWFHR'04 (International Workshop on Frontiers in Handwriting Recognition) [81]; tandis que la résolution de problèmes multi-classes et l'estimation de probabilités *a posteriori,* ont été l'objet d'une première communication présentée à IJCNN'05 (International Join Conference on Neural Networks) [82] et d'une seconde présentée à IWFHR'06 [38]. Pour finir, un article faisant la synthèse des résultats concernant l'estimation de probabilités et l'accélération de la prise de décision, devrait être soumis prochainement dans un journal tel que PR (Pattern Recognition) ou IJDAR (International Journal of Document Analysis and Recognition).

Perspectives

Bien entendu, les approches que nous avons proposées sont encore largement perfectibles et plusieurs perspectives de travaux de recherche sont envisageables.

Pour améliorer la qualité des probabilités estimées par les SVM, une perspective intéressante consisterait à revenir au niveau du processus d'extraction de

caractéristiques. L'objectif serait alors de déterminer des ensembles de caractéristiques optimisées pour chaque sous-problème de la stratégie « un contre un ». Les probabilités « locales » seraient donc estimées sur des espaces de représentation différents, tandis que les probabilités « globales » seraient estimées à partir des sorties des SVM de la stratégie « un contre tous » qui eux seraient entraînés sur un même espace de représentation et permettraient de combiner efficacement les probabilités « locales ».

Concernant l'accélération de la prise de décision, une perspective pour améliorer les performances du système à deux niveaux de décision, consisterait à utiliser des seuils différents pour chacune des classes afin de déterminer les probabilités qui doivent être ré-estimées. Il pourrait alors être intéressant d'utiliser un algorithme multi-critères pour optimiser ces différents seuils, mais aussi d'autres paramètres comme le nombre de neurones cachés du MLP.

Enfin, de part sa grande modularité, l'approche proposée dans le dernier chapitre semble être parfaitement adaptée aux problèmes de classification où le nombre de classes est très important. Il serait donc interressant de tester ce type d'approche sur des problèmes de reconnaissance d'idéogrammes chinois ou japonais. De même, ce systême présente l'avantage de pouvoir faire varier dynamiquement le nombre de classes, en fonction d'une source d'information supplémentaire tel qu'un modèle de langage dans le cas de la lecture automatique de l'écriture manuscrite.

Pour conclure, nous noterons que les résultats que nous avons obtenus sont déjà très encourageants et semblent donc ouvrir la voie à l'intégration des SVM au sein de systèmes de reconnaissance de formes. Ainsi, concernant l'application à la lecture automatique de l'écriture manuscrite, la prochaine étape vers l'intégration au sein de systèmes industriels de tri automatique de documents consisterait donc à tester l'impact sur un système complet, tel que le système de reconnaissance de noms de ville présenté en introduction.

BIBLIOGRAPHIE

[1] C.-L. Liu, K. Nakashima, H. Sako, and H. Fujisawa. Handwritten digit recognition: benchmarking of state-of-the-art techniques. *Pattern Recognition*, 36(10), 2271-2285, 2003.

[2] N.-E. Ayat. *Sélection de modèle automatique des machines à vecteurs de support : Application à la reconnaissance d'images de chiffres manuscrits*. Thèse de doctorat, École de Technologie Supérieure, Université du Québec, Montréal, Canada, 2003.

[3] F. Grandidier. *Un nouvel algorithme de sélection de caractéristiques - Application à la lecture automatique de l'écriture manuscrite*. Thèse de doctorat, École de Technologie Supérieure, Université du Québec, Montréal, Canada, 2003.

[4] A. L. Koerich. *Large vocabulary off-line handwritten word recognition*. Thèse de doctorat, École de Technologie Supérieure, Université du Québec, Montréal, Canada, 2002.

[5] M. E. Morita. *Automatic recognition of handwritten dates on brazilian bank cheques*. Thèse de doctorat, École de Technologie Supérieure, Université du Québec, Montréal, Canada, 2003.

[6] L. E. S. Oliveira. *Automatic recognition of handwritten numerical strings*. Thèse de doctorat, École de Technologie Supérieure, Université du Québec, Montréal, Canada, 2003.

[7] P. V. W. Radtke. *Classification systems optimization with multi-objective evolutionary algorithms*. Thèse de doctorat, École de Technologie Supérieure, Université du Québec, Montréal, Canada, 2006.

[8] A. L. Koerich, R. Sabourin, and C. Y. Suen. Recognition and Verification of Unconstrained Handwritten Words. *IEEE transaction on Pattern Analysis and Machine Intelligence* 27(10), 1509-1522, 2005.

[9] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Second Edition, Springer, 1999.

[10] C. Cortes and V. N. Vapnik. Support-vector networks. *Machine Learning*, 20(3), 273-297, 1995.

[11] B. E. Boser, I. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifier. *Workshop of Computational Learning Therory*, 144-152, Pittsburg, 1992.

[12] B. Schölkopf and A. Smola. *Learning with Kernels. Support Vector Machines, Regularization, Optimization, and Beyond* MIT Press, 2002.

[13] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, *A Practical Guide to Support Vector Classification*, Department of computer science and information engineering, National Taiwan University, Technical report 2003.

[14] E. Mayoraz and E. Alpaydin. Support vector machines for multi-class classification. *IWANN*, 833-842, 1999.

[15] T. Hamamura, H. Mizutani, and B. Irie. A multiclass classification method based on multiple pairwise classifiers. *International Conference on Document Analysis and Recognition*, 809-813, Edingbugh, Scotland, August 3-6, 2003.

[16] T. Hastie and R. Tibshirani. Classification by pairwise coupling. *The Annals of Statistics*, 26(1), 451-471, 1998.

[17] U. Kreßel, Pairwise classification and support vectors machines, in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge, MA: MIT Press, 1999, 255-268.

[18] M. Moreira and E. Mayoraz. Improved pairwise coupling classification with correcting classifiers. *European Conference on Machine Learning*, 160-171, Chemnitz, Germany, April 21-24, 1998.

[19] D. Price, S. Knerr, L. Personnaz, and G. Dreyfus, Pairwise Neural Network Classifiers with Probabilistic Outputs, in *Neural Information Processing Systems*: MIT Press, 1995, 1109-1116.

[20] P. Savicky and J. Fürnkranz. Combining Pairwise Classifiers with Stacking. *International Symposium on Intelligent Data Analysis* 219-229, Berlin, Germany, 2003.

[21] T.-F. Wu, C.-J. Lin, and R. C. Weng. Probability Estimates for Multi-class Classification by Pairwise Coupling. *Journal of Machine Learning Research*, 5, 975-1005, 2004.

[22] E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers. *Journal of Machine Learning Research*, 1, 113-141, 2000.

[23] J. Fürnkranz. Round robin classification. *Journal of Machine Learning Research*, 2(Mar), 721-747, 2002.

[24] C.-W. Hsu and C.-J. Lin. A comparison of methods for multi-class support vector machines. *IEEE transactions on Neural Networks*, 13(2), 415-425, 2002.

[25] J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large Margin DAGs for Multiclass Classification. *Advances in Neural Information Processing Systems* 547-553, 2000.

[26] F. Schwenker. Hierarchical Support Vector Machines for Multi-class Pattern Recognition. *International Conference on Knowledge-Based Intelligent Engineering Systems & Allied Technologies*, 561-565, Brighton, UK, 30 Aug-01Sept, 2000.

[27] T. G. Dietterich and G. Bakiri. Solving Multiclass Learning Problems via Error-Correcting Output Codes. *Journal of Artificial Intelligence Research*, 2, 263-286, 1995.

[28] K. Crammer and Y. Singer. On the learnability and design of output codes for multiclass problems. *Machine Learning*, 47(2), 201-233, 2002.

[29] V. Vapnik. *Statistical Learning Theory.* New-York, Wiley, 1998.

[30] J. Weston and C. Watkins. Multi-class support vector machines. *European Symposium on Artificial Neural Networks*, 219-224, Bruges, Belgium, April 21-23, 1999.

[31] E. Bredensteiner and K. P. Bennett. Multicategory classification by support vector machines. *Computational Optimizations and Applications*, 12, 53-79, 1999.

[32] Y. Lee, Y. Lin, and G. Wahba. Multicategory Support Vector Machines. *Computing Science and Statistics*, 33, 498-512, 2001.

[33] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2, 265-292, 2001.

[34] Y. Guermeur. Combining Discriminant Models with New Multi-Class SVMs. *Pattern Analysis & Applications*, 5, 168-179, 2002.

[35] R. Rifkin and A. Klautau. In defence of one-vs-all classification. *Journal of Machine Learning Research*, 5(Jan), 101-141, 2004.

[36] P. J. Grother, *NIST Special Database 19-Handprinted Forms and Characters Database*, National Institute of Standards and Technology 1995.

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

[38] J. Milgram, M. Cheriet, and R. Sabourin. "One Against One" or "One Against All": Which One is Better for Handwriting Recognition with SVMs? *International Workshop on Frontiers in Handwriting Recognition*, 181-186, La Baule, France, October 23-26, 2006.

[39] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, Department of computer science and information engineering, National Taiwan University, Technical Report 2001.

[40] G. Wahba, Support vector machines, reproducing kernel Hilbert spaces and the randomized GACV, in *Advances in Kernel Methods Support Vector Learning*, C. B. A. S. B. Schoelkopf, Ed. Cambridge, MA: MIT Press, 1999, 69-88.

[41] W. Chu, S. S. Keerthi, and C. J. Ong. Bayesian Trigonometric Support Vector Classifier. *Neural Computation*, 15, 2227-2254, 2003.

[42] J. C. Platt, Probabilities for SV Machines, in *Advances in Large Margin Classifiers*, A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, Eds.: MIT Press, 1999, 61-74.

[43] H.-T. Lin, C.-J. Lin, and R. C. Weng, *A note on Platt's probabilistic outputs for support vector machines*, Department of computer science and information engineering, National Taiwan University, Technical report 2003.

[44] B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. *International Conference on Knowledge Discovery and Data Mining*, Edmonton, Alberta, Canada, July 23-26, 2002.

[45] J. Bridle, Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition, in *Neurocomputing: algorithm, architectures, and applications*, F. F.-S. J. Herault, Ed. New York: Springer, 1989, 227-236.

[46] K. Duan, S. S. Keerthi, S. K. Shevade, W. Chu, and A. N. Poo. Multi-category classification by soft-max combination of binary classifiers. *International Workshop on Multiple Classifier Systems*, 125-134, Guildford, UK, June11-13, 2003.

[47] C. K. Chow. On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory*, 16(1), 41-46, 1970.

[48] D. d. Ridder, E. Pekalska, and R. P. W. Duin. The economics of classification: error vs. complexity. *International Conference on Pattern Recognition*, Québec, Canada, 11-15 August, 2002.

[49] C. J. C. Burges. Simplified support vector decision rules. *International Conference on Machine Learning*, 71-77, Bari, Italy, 1996.

[50] C. J. C. Burges and B. Schölkopf. Improving speed and accuracy of support vector learning machines. *Neural Information Processing Systems*, 375-381, 1997.

[51] B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K. Muller, G. Ratsch, and A. J. Smola. Input space versus feature space in kernel-based methods. *IEEE Transaction on Neural Networks*, 10, 1000–1017, 1999.

[52] D. Nguyen and T. Ho. An Efficient Method for Simplifying Support Vector Machines. *International Conferences on Machine Learning*, Bonn, Germany, 2005.

[53] B. Tang and D. Mazzoni. Multiclass Reduced-Set Support Vector Machines. *International Conference on Machine Learning*, 921-928, Pittsburg, Pennsylvania, 2006.

[54] T. Downs, K. E. Gates, and A. Masters. Exact Simplification of Support Vector Solutions. *Journal of Machine Learning Research*, 2, 293-297, 2001.

[55] S. Romdhani, P. Torr, B. Schölkopf, and A. Blake. Computationally Efficient Face Detection. *International Conference on Computer Vision*, 695-700, Vancouver, Canada, 2001.

[56] E. Parrado-Hernandez, I. Mora-Jimenez, J. Arenas-Garcia, A.-R. Figuera-Vidal, and A. Navia-Vasquez. Growing support vector classifiers with controlled complexity. *Pattern Recognition*, 36(7), 1479-1488, 2003.

[57] D. DeCoste and D. Mazzoni. Fast Query-Optimized Kernel Machine Classification Via Incremental Approximate Nearest Support Vectors. *International Conference on Machine Learning*, 115-122, Washington DC, 2003.

[58] A. Bellili, M. Gilloux, and P. Gallinari. An MLP-SVM combination architecture for offline handwritten digit recognition: Reduction of recognition errors by

Support Vector Machines rejection mechanisms. *International Journal on Document Analysis and Recognition*, 5(4), 244-252, 2003.

[59] L. E. S. Oliveira, A. S. Brito, and R. Sabourin. Improving Cascading Classifiers with Particle Swarm Optimization. *International Conference on Document Analysis and Recognition*, 570-574, Seoul, South Korea, August 29-September 1st, 2005.

[60] E. Oja and T. Kohonen. The subspace learning algorithm as a formalism for pattern recognition and neural networks. *International Conference on Neural Networks*, 277-284, San Diego, USA, July 24-27, 1988.

[61] H. Schwenk. *Amélioration de classifieurs neuronaux par incorporation de connaissances explicites : Applications à la reconnaissance de caractères manuscrits*. Thèse de doctorat, Université Paris VI, 1996.

[62] A. Biem. Minimum Classification Error Training for Online Handwritten Word Recognition. *International Workshop on Frontiers in Handwriting Recognition*, 61-65, Niagra-on-the-lake, August 6-8, 2002.

[63] C.-L. Liu, H. Sako, and H. Fujisawa. Performance evaluation of pattern classifiers for handwritten character recognition. *International Journal on Document Analysis and Recognition*, 4(3), 191-204, 2002.

[64] F. Kimura, K. Takashina, S. Tsuruoka, and Y. Miyake. Modified quadratic discriminant functions and the application to chinese character recognition. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 9(1), 149-153, 1987.

[65] C.-L. Liu, H. Sako, and H. Fujisawa. Learning quadratic discriminant function for handwritten character classification. *International Conference on Pattern Recognition*, 44-47, Quebec, Canada, 2002.

[66] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[67] E. Francesconi, M. Gori, S. Marinai, and G. Soda. A serial combination of connectionist-based classifiers for OCR. *International Journal on Document Analysis and Recognition*, 3(3), 160-168, 2001.

[68] N. Ragot and E. Anquetil. A generic hybrid classifier based on hierarchical fuzzy modeling: Experiments on on-line handwritten character recognition. *International Conference on Document Analysis and Recognition*, 963-967, Edinburgh, Scotland, august 3-6, 2003.

[69] L. Prevost, L. Oudot, A. Moises, C. Michel-Sendis, and M. Milgram. Hybrid generative/discriminative classifier for unconstrained character recognition. *Pattern Recognition Letters*, 26, 1840–1848, 2005.

[70] C.-H. Chou, C.-C. Lin, Y.-H. Liu, and F. Chang. A prototype classification method and its use in a hybrid solution for multiclass pattern recognition. *Pattern Recognition*, 39, 624-634, 2006.

[71] L. Vuurpijl and L. Schomaker. Two-stage character classification: A combined approach of clustering and support vector classifiers. *International Workshop on Frontiers in Handwriting Recognition*, 423-432, Amsterdam, Netherlands, september 11-13, 2000.

[72] K. T. Abou-Moustafa, M. Cheriet, and C. Y. Suen. Classification of Time-Series data Using a Generative/Discriminative Hybrid. *International Workshop on Frontiers in Handwriting Recognition*, 51-56, Tokyo, Japan, October 26-29, 2004.

[73] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of IEEE*, 86(11), 2278-2324, 1998.

[74] P. Y. Simard, D. Steinkraus, and J. C. Platt. Best practices for convolutional networks applied to visual document analysis. *International Conference on Document Analysis and Recognition*, 958-962, Edinburgh, Scotland, 3-6 August, 2003.

[75] F. Kimura, S. Inoue, T. Wakabayashi, S. Tsuruoka, and Y. Miyake. Handwritten numeral recognition using autoassociative neural networks. *International Conference on Pattern Recognition*, 166-171, Brisbane, Australia, August 16-20, 1998.

[76] R. O. Duda, P. E. Hart, and D. G. Stork, Lack of inherent superiority of any classifier, in *Pattern Classification*, Second Edition ed: Wiley-Interscience, 2001, 454-465.

[77] J. Milgram, R. Sabourin, and M. Cheriet. Two-stage classification system combining model-based and discriminative approaches. *International Conference on Pattern Recognition*, 155-162, Cambridge, U.K., August 23-26, 2004.

[78] J. Milgram, R. Sabourin, and M. Cheriet. Combining Model-based and Discriminative Approaches in a modular Two-stage Classification System: Application to Isolated Handwritten Digit Recognition. *Electronic Letters on Computer Vision and Image Analysis*, 5(2), 1-15, 2005.

[79] J. Milgram, R. Sabourin, and M. Cheriet. Système de classification à deux niveaux de décision combinant approche par modélisation et machines à vecteurs de support. *Colloque International Francophone sur l'Ecrit et le Document*, 25-29, La Rochelle, France, 21-25 juin, 2004.

[80] J. Milgram, R. Sabourin, and M. Cheriet. Système de classification à deux niveaux de décision combinant approche par modélisation et machines à vecteurs de support *Traitement du Signal*, 22(3), 293-304, 2005.

[81] J. Milgram, M. Cheriet, and R. Sabourin. Speeding Up the Decision Making of Support Vector Classifiers. *International Workshop on Frontiers in Handwriting Recognition*, 57-62, Tokyo, Japan, October 26-29, 2004.

[82] J. Milgram, M. Cheriet, and R. Sabourin. Estimating Accurate Multi-class Probabilities with Support Vector Machines. *International Joint Conference on Neural Networks*, 1906-1911, Montreal, Canada, August 1-4, 2005.