République Algérienne Démocratique et Populaire
وزارة التعليـم العالـــي والبحـــث العلمـــي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université des Sciences et des Technologies d'Oran Mohamed BOUDIAF

Faculté des Sciences
Département d'informatique
Laboratoire SIMPA

Spécialité : Informatique

OPTION : Modélisation, Optimisation et Evaluation des Performances des Systèmes (MOEPS)

MÉMOIRE

Présenté par

Mr. MAMOUNI EL MAMOUN

Thème

Sélection expérimentale de modèles SVM multi-classes application à la reconnaissance des caractères arabes

Soutenu le 11/10/ 2012 à 14 heur dans la salle de soutenance

Devant la commission d'examen composée de :

• Président	Mr BENYETTOU Abdelkader	Pr	USTO-MB
• Rapporteur	Mr SADOUNI Kaddour	M.C.A	USTO-MB
• Examinateur	Mr BELKADI Khaled	M.C.A	USTO-MB
• Examinateur	Mr RAHAL Sid-Ahmed	M.C.A	USTO-MB
• Invité	Mr ZENNAKI Mahmoud	M.A.A	USTO-MB

Année universitaire : 2012-2013

Remerciements

Tous nos remerciements s'adressent tous d'abord à tout puissant ALLAH, d'avoir guidé nos pas vers le chemin du savoir.

Nous tenons à remercier vivement notre encadreur ***Mr K. Sadouni*** *pour sa patienté, pour ses conseils précieux et pour toutes les commodités et aisances qu'il nous a apportées durant notre étude et réalisation de ce projet.*

Nos remerciements les plus vifs s'adressent aussi a notre co-encadreur ***Mr M.Zennaki*** *pour leur aide, pour l'intéressante documentation qui a mis à notre disposition, son suivi pendant la réalisation de ce travail, ses discussions qui ont aidé à résoudre des problèmes rencontrés pendant la réalisation de ce mémoir.*

J'aimerai aussi remercier messieurs le président de jury ***Mr A.Benyettou*** *et les membres de jury* ***Mr K.Belkadi*** *et* ***Mr S.Rahal*** *d'avoir accepté d'examiner et d'évaluer notre travail.*

Nous exprimons également notre gratitude à tous les enseignants qui ont collaboré à notre formation durant la première année poste graduation MOEPS. Sans omettre bien sûr de remercier profondément touts les personnes qui ont participé de prés ou de loin à la réalisation de ce travail.

Et enfin, que nos chers parents et familles, trouvent ici l'expression de nos remerciements les plus sincères et les plus profonds en reconnaissance de leurs sacrifices, soutien et encouragement.

El Mamoun Mamouni

Résumé

Nous présentons dans cet mémoire une étude expérimentale qui a pour objectif la sélection optimale du modèle SVM pour la reconnaissance des caractères arabes manuscrits en utilisant la recherche tabou pour le choix et le balayage d'un grand espace de paramètres de modèles SVM. Ces modèles regroupent le type de classification (une-contre-une ou une-contre-reste), le noyau SVM ainsi que les paramètres du noyau. Le choix de ces paramètres a une grande influence sur les performances du classifieur final et aussi sur le temps de calcul.
Ce travail a nécessité la réalisation d'un système complet de reconnaissance des caractères arabes manuscrits hors-ligne, la génération d'une base de données de 4840 caractères dans leurs différentes positions (début, milieu, fin et isolé) et la mise au point d'un algorithme de sélection de modèles SVM basé sur la recherche tabou conduisant aux meilleurs résultats possibles en termes de taux de prédiction et temps CPU. Les résultats préliminaires obtenus sont très encourageants et prometteurs par rapport à la littérature et montrent que l'utilisation de métaheuristiques peut être bénéfique à une meilleure reconnaissance des caractères arabes manuscrits par les SVM.

Mots clés

Reconnaissance, caractères arabes manuscrits, sélection de modèle, Machines à vecteurs de support SVM, SVM multi-classe, paramètres de modèle, recherche tabou.

Abstract

We present in this memorial an experimental study which aims to do an optimal selection of SVM model for the recognition of handwritten Arabic characters using the tabu search for selection and scanning of a large parameter space of SVM models. These models include the type of classification (one-against-one or one-against-rest), the kernel SVM and kernel parameters. The choice of these parameters has a great influence on the final performance of the classifier and also the computation time.
This work required the realization of a complete system of recognition offline of Arabic manuscripts, the generation of database of 4840 characters in their different positions (beginning, middle, end and isolated) and the development a selection algorithm of SVM models based on tabu search leading to best results in terms of prediction rate and CPU time. Preliminary results are very encouraging and promising compared to the literature and show that the use of metaheuristics can be beneficial to a better recognition of Arabic manuscripts by the SVM.

Keywords

Recognition, handwritten Arabic characters, model selection, support vector machines SVM, SVM multiclass, model parameters, tabu search.

Sommaire

Chapitre I : Reconnaissance de caractères arabe manuscrits

Chapitre II : Machines à vecteurs de support multi-classes

Chapitre III : Sélection de modèle machines à vecteurs de support

Chapitre IV : Résultats expérimentaux et discussions

Liste des figures

Introduction

Ces dernières années, des progrès considérables ont été réalisés dans la mise en œuvre de systèmes pour la reconnaissance de l'écriture manuscrite, et cela grâce, d'une part aux nombreux travaux effectués dans ce domaine et d'autre part, à la production évaluée à bas prix des micro-ordinateurs et des systèmes d'acquisition (scanner, tablette à digitaliser... etc.).

Malgré les avancées d'autres moyens de communication tel que l'audio visuel, nombreuses sont les applications dont l'existence commence sur le papier, plus particulièrement dans la bureautique, en publication assistée par ordinateur (pour faciliter la composition à partir d'une sélection de plusieurs documents), dans la poste (lecture des adresses et tri automatique), dans les banques (traitement des chèques, des factures). Cependant malgré les progrès technologiques, le clavier reste encore un moyen obligé de communication avec l'ordinateur.

La reconnaissance de l'écriture manuscrite par ordinateur est un domaine très vaste, les travaux de recherches sur l'écriture arabe sont moins nombreux en comparaison avec d'autres types d'écriture (le latin, le japonais…). En plus la cursivité de l'écriture arabe montre une complexité de la morphologie des caractères. Ce problème engendre une forte inertie à différents niveaux notamment :

- Le choix de primitives pertinentes décrivant la variabilité de la morphologie des caractères, sachant que certaines caractéristiques topologiques sont sensibles à la dégradation, notamment les points diacritiques et les boucles.
- La nécessité d'une modélisation robuste et une méthode d'apprentissage efficace pour prendre en considération toutes les variations morphologiques de l'écriture arabe.

Parmi les techniques utilisées pour la reconnaissance des caractères arabes manuscrits, nous retrouvons les machines à vecteurs de support (SVM) basés sur la théorie de l'apprentissage statistique [52]. Les SVM introduites au début des années 90, Elles réalisent un grand succès dans la théorie de l'apprentissage statistique. Aujourd'hui, nous pouvons dire sans exagérer que ces machines ont supplanté les réseaux de neurones et les autres techniques d'apprentissage. En effet, elles sont largement répandues en apprentissage statistique et ont eu beaucoup de succès dans quasiment tous les domaines où elles ont été appliquées.

Il est nécessaire de distinguer également la reconnaissance en ligne (on-line) de l'écriture manuscrite, qui relève plutôt de l'interfaçage entre l'homme et l'ordinateur (un stylo spécial est connecte a la machine et ne fonctionne que sur une tablette sensible), et la reconnaissance hors ligne (off-line) où l'entrée est une image numérique de l'écriture. Seule la reconnaissance hors ligne sera considérée dans ce travail.

L'objectif le plus important visé par ce travail est la sélection expérimentale du modèle SVM pour la reconnaissance des caractères arabes manuscrits hors-ligne. Nous avons utilisé une approche basée sur la recherche tabou pour le choix et le balayage d'un grand espace de paramètres afin d'avoir un taux de reconnaissance appréciable.

Notre mémoire est structuré en quatre chapitres :

Le premier chapitre de ce mémoire, présente le concept général de reconnaissance des caractères arabes manuscrits, en mettant le point sur les caractéristiques morphologique de l'écriture Arabe et les différents aspects d'un OCR ainsi que les différentes phases de processus de reconnaissance. Enfin, nous avons présenté quelques récents travaux réalisés dans le domaine de la reconnaissance de l'écriture arabe.

Le deuxième chapitre représente un état de l'art sur les machines à vecteur de support en général avec une synthèse des différentes approches des SVM multi-classes présente à travers une analyse minutieuse de ces approches en mettant en lumière leurs atouts et leurs défauts.

Dans le chapitre 3 nous décrivons les techniques et les méthodes de sélection de modèle SVM, ainsi nous avons présenté dans ce chapitre la stratégie utilisée pour la recherche et la sélection des paramètres du modèle SVM.

Nous présentons dans le chapitre 4 une description détaillée de notre système de reconnaissance de caractères Arabes manuscrits hors-ligne. Ainsi que nous avons présenté les différents résultats d'expérimentations par l'utilisation des différents noyaux du SVM et l'implémentation de notre stratégie de recherche pour varier et sélectionner les hyper-paramètres qui donnent le bon taux de reconnaissance. Finalement, nous présentons une discussion et évaluation des résultats obtenus.

Chapitre I

Reconnaissance de caractères arabes manuscrits

1.1 Introduction

Les recherches sur la reconnaissance des caractères Arabes exposent un domaine qui s'étend rapidement et indéfiniment évoquées par une place aussi importante dans les deux dernières décennies. C'est ainsi que la reconnaissance des caractères Arabes constitue aujourd'hui une préoccupation dont la pertinence est incontestée par la communauté de chercheurs qui ont dévoués leurs efforts à réduire les contraintes et à élargir le royaume de la reconnaissance des caractères Arabes.

Un système de reconnaissance de l'écriture doit idéalement, localiser, reconnaître et interpréter n'importe quel texte ou nombre écrit sur un support de qualité arbitrairement variable tel que des cartes, des formulaires, des agendas, des vieux manuscrits, etc.

Parmi les domaines d'application, on trouve le domaine postal pour la reconnaissance du code, de l'adresse postale et la lecture automatique des chèques bancaires; le domaine administratif pour la gestion électronique des flux de documents; les bibliothèques numériques pour l'indexation de documents et la recherche d'informations; la biométrie pour l'identification du scripteur…etc.

1.2 Caractéristiques de l'écriture Arabe

L'arabe est une écriture consonantique qui utilise un alphabet de 28 lettres (Tableau 1.1), auquel il faut ajouter la Hamza « ء », qui est le plus souvent considérée comme signe complémentaire [4]. La hamza « ء » a une orthographe spéciale qui dépend de règles grammaticales, ce qui multiplie les formes nécessaires à sa représentation, puisqu'elle peut s'écrire seule ou sur le support de trois voyelles (alif, waw et ya) dont elle suit le code (Tableau 1.1).

De plus l'alphabet arabe comprend d'autres caractères additionnels tels que « ة » et « لا ». De ce fait, certains auteurs considèrent que l'alphabet arabe comprend plutôt 31 au lieu de 29 lettres. La considération du symbole « ~» qui s'écrit uniquement sur le support du caractère « ا », fait apparaître d'autres graphismes (Tableaux 1.1) [80].

N°	Lettre isolé	Au début	Au milieu	A la fin	Prononciation
01	آ / أ / ا	آ / أ / ا	آ / أ / ا	آ / أ / ا	Alif
02	ب	بـ	بـ	ـب	Ba
03	ت	تـ	تـ	ـت	Ta
04	ث	ثـ	ثـ	ـث	Tha
05	ج	جـ	جـ	ـج	Jim
06	ح	حـ	حـ	ـح	Ha
07	خ	خـ	خـ	ـخ	Kha
08	د	د	د	ـد	Dal
09	ذ	ذ	ذ	ـذ	Dhal
10	ر	ر	ر	ـر	Ra
11	ز	ز	ز	ـز	Zay
12	س	سـ	سـ	ـس	Sin
13	ش	شـ	شـ	ـش	Chin
14	ص	صـ	صـ	ـص	Sad
15	ض	ضـ	ضـ	ـض	Dhad
16	ط	طـ	طـ	ـط	TTa
17	ظ	ظـ	ظـ	ـظ	Dha
18	ع	عـ	عـ	ـع	Ayn
19	غ	غـ	غـ	ـغ	Ghayn
20	ف	فـ	فـ	ـف	Fa
21	ق	قـ	قـ	ـق	Qaf
22	ك	كـ	كـ	ـك	Kaf
23	ل	لـ	لـ	ـل	Lam
24	م	مـ	مـ	ـم	Mim
25	ن	نـ	نـ	ـن	Noun
26	ه	هـ	هـ	ـه	He
27	و	و	و	ـو	Waw
28	ي	يـ	يـ	ـي	Ya
Autre formes de caractères (complémentaires, combinaisons)					
29	ء	ئـ / ـئـ	ؤ	ـئ	Hamza
30	لآ	لآ	لآ / ألآ	ـلآ	Madda + alif
31	لإ / لأ	لإ / لأ	ملإ / ملأ	ـلإ / ـلأ	Lam alif + hamza
32	ة			ـة	Ta marbouta
33	لا	لا	لا	ـلا / لا	Lam alif

Tableau 1.1 Alphabet arabe dans ses différentes formes

Un trait caractéristique de l'écriture arabe est la présence d'une ligne de base horizontale dite encore ligne de référence ou d'écriture. C'est le lieu des caractères d'une même chaîne (Figure 1.1).

Figure 1.1 Exemple d'écriture arabe montrant la ligne de base

Certains caractères ne peuvent être rattachés à leur gauche, ainsi ils ne peuvent se trouver qu'en position isolée ou finale; ce qui donnent quand ils existent, des mots composés d'une ou de plusieurs parties appelés généralement PAW (Peace of Arabic Word) ou encore pseudo-mot [80]. Un PAW correspond donc à une chaîne d'un ou de plusieurs caractères (Tableau 1.2).

3 PAW/mot	2 PAW/mot	1 PAW/mot
		

Tableau 1.2 Exemple de mots composés de la droite vers la gauche de 1, 2, 3 PAWs

Pour des raisons de justification de texte et/ou d'esthétique, les ligatures horizontales peuvent être allongées en insérant entre les caractères d'une même chaîne une ou plusieurs élongations « madda » (ou tatwil), correspondant au symbole « - ». L'élongation se situe toujours à gauche du caractère coûtant. Si le trait d'allongement est associé à un caractère en position début ou finale, le caractère prend sa forme du milieu et voit sa chasse augmenter du nombre de « madda » insérées (Tableau 1.3) [5].

Au niveau du PAW, l'insertion de traits d'allongement affecte uniquement sa largeur, la morphologie reste la même [6]. Les éditeurs de texte tels que Word de Microsoft, insèrent dans les lignes de texte, le nombre approprié de « Madda », pour la justification gauche-droite d'un texte arabe.

Avec 6 maddas	Avec 3 maddas	Avec 1 madda	Sans madda
قـــــ	قـــ	قـ	ق
تـــــ	تـــ	تـ	ت

Tableau 1.3 Exemple de caractères avec et sans madda

L'écriture arabe est semi-cursive dans sa forme imprimée ainsi que manuscrite. Les caractères d'une même chaîne (ou pseudo-mots) sont ligaturés horizontalement et parfois verticalement (dans certaines fontes deux, trois et même quatre caractères peuvent être ligaturés verticalement), rendant difficile la segmentation en caractères.

قـ{ جـ , حـ , خـ }	فـ{ جـ , حـ , خـ }	لـ{ جـ , حـ , خـ }	مـ{ جـ , حـ , خـ }

Tableau 1.4 Caractères susceptibles d'être ligaturés verticalement

Ligatures obligatoires des lettres :{ ة ح م ل }: **لمحة** :

Ligature esthétique entre les 2 premières lettres : لمـــحة

Ligature esthétique entre les 3 premières lettres : لمحة

De plus, la forme d'un caractère diffère selon sa position dans les pseudo-mots et même dans certains cas; selon le contexte phonétique. En outre, plus de la moitié des caractères arabes incluent dans leur forme des points diacritiques. Ces points peuvent se situer au-dessus ou au-dessous du caractère, mais jamais en haut et en bas simultanément. Plusieurs caractères peuvent avoir le même corps mais un nombre et /ou une position de points diacritiques différents.

Les caractères arabes peuvent être voyellés. Les voyelles appelées aussi diacritiques dans certains documents et courtes voyelles dans d'autres, tels que [7], peuvent se placer au dessus ou en dessous du caractère. Les voyelles sont d'une invention postérieure aux consonnes. Dans l'arabe contemporain ordinaire, on écrit seulement les consonnes et les voyelles longues. Un même mot avec différentes voyelles courtes peut être compris comme verbe, nom ou adjectif. Par d'exemple « **علم** » peut signifier « drapeau :**عَلَمُ** » ou « savoir : **عِلْمُ** » ou encore « Enseigner : **عَلَمَ** », selon sa voyellation.

Il existe 8 signes de voyellation qui peuvent se placer au dessus de la ligne d'écriture, tels que fatha (ـَ) dammah (ـُ), soukoun (ـْ) et chaddah (ـّ) qui doit être accompagnée de l'une des

voyellations fatha, Dammah ou kasrah, et ceux qui peuvent se placer en dessous de la ligne d'écriture tels que Kasrah (ـِ). De plus trois «tanwin» peuvent être formés à partir d'un double fatha (ـً), d'un double dammah (ـٌ) ou d'un double kasrah (ـٍ).

L'écriture arabe contient beaucoup de polices et modèles d'écriture, ainsi il est difficile parfois de séparer un mot d'un autre, particulièrement quand les gens écrivent avec la calligraphie.

Figure 1.2 Différentes phrases arabes dans différents modèles [8]

Par ailleurs, la cursivité de l'écriture arabe montre une complexité de la morphologie des caractères, les élongations des ligatures horizontales ainsi que les combinaisons verticales de certains caractères, constituent les problèmes majeurs liés au traitement de cette écriture surtout pour les pseudo-mots.

En effet, ces problèmes engendrent une forte inertie à différents niveaux notamment :

- Le choix de primitives pertinentes décrivant la variabilité de la morphologie des caractères, sachant que certaines caractéristiques topologiques sont sensibles à la dégradation, notamment les points diacritiques et les boucles.

- La méthode de segmentation en caractères ou même en pseudo-mots (qui peuvent se chevaucher surtout dans le cas du manuscrit).

Tous ces problèmes et bien d'autres, se trouvent accentués dans le cas du manuscrit où d'autres facteurs interviennent (conditions de l'écriture, fusion de points diacritiques, chevauchement de pseudo-mots, graphismes inégalement proportionnés...).

Face à ces problèmes, la nécessité d'une modélisation robuste une méthode d'apprentissage efficace pour prendre en considération toutes les variations morphologiques de l'écriture arabe.

1.3 Différents aspects de l'OCR (Optical Character Recognition)

Il n'existe pas de système universel d'OCR qui permet de reconnaître n'importe quel caractère dans n'importe quelle fonte. Cela dépend du type de données traitées et bien évidemment de l'application visée [9]. Il existe plusieurs modes de classification des systèmes OCR parmi lesquels on peut citer:

- Les systèmes qualifiés de « en-ligne » ou « hors-ligne » suivant le mode d'acquisition.
- Les approches globales ou analytiques selon que l'analyse s'opère sur la totalité du mot, ou par segmentation en caractères.
- Les approches statistiques, structurelles ou stochastiques relatives aux traits caractéristiques extraits des formes considérées.

La Figure 1.3 représente les différentes dichotomies de type d'écriture

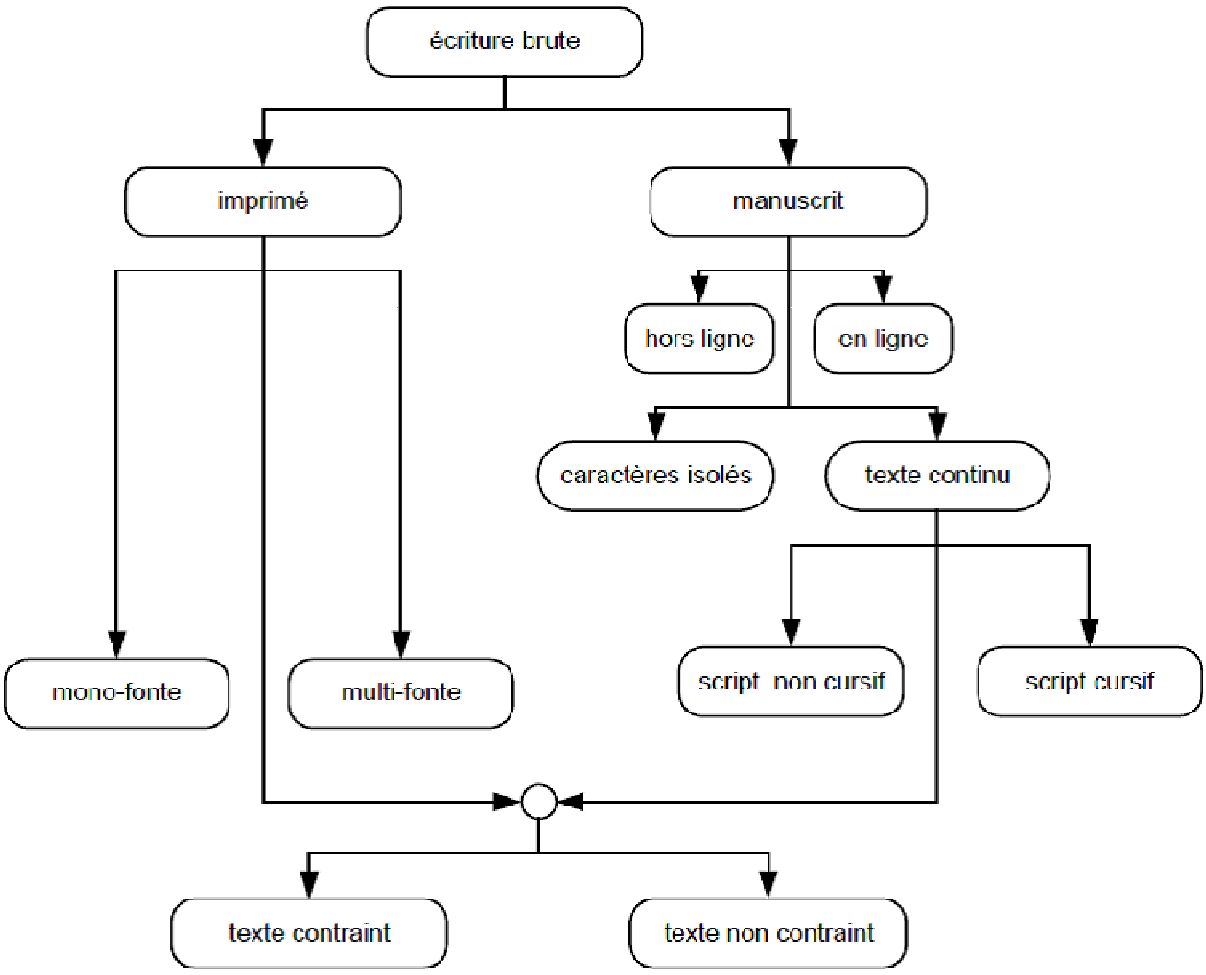


Figure 1.3 Dichotomie des types d'écriture [3]

1.3.1 Reconnaissance (En-Ligne et Hors-Ligne)

Ce sont deux modes différents d'OCR, ayant chacun ses propres outils d'acquisition et ses algorithmes correspondants de reconnaissance.

1.3.1.1 Reconnaissance En-Ligne (on-line)

Ce mode de reconnaissance s'opère en temps réel (pendant l'écriture). Les symboles sont reconnus au fur et à mesure qu'ils sont écrits à la main.

Ce mode est réservé généralement à l'écriture manuscrite, c'est une approche «signal» où la reconnaissance est effectuée sur des données à une dimension. L'écriture est représentée comme un ensemble de points dont les coordonnées sont en fonction du temps [10].

1.3.1.2 Reconnaissance hors-ligne (off-line)

Elle démarre après l'acquisition, elle convient aux documents imprimés et les manuscrits déjà rédigés. Ce mode peut être considéré comme le cas le plus général de la reconnaissance de l'écriture. Il se rapproche du mode de la reconnaissance visuelle. L'interprétation de l'information est indépendante de la source de génération [11].

La reconnaissance hors-ligne peut être classée en plusieurs types :

- Reconnaissance de texte ou analyse de documents : Dans le premier cas il s'agit de reconnaître un texte de structure limitée à quelques lignes ou mots. La recherche consiste en un simple repérage des mots dans les lignes, puis à un découpage de chaque mot en caractères [9].

  Dans le second cas (analyse de document), il s'agit de données bien structurées dont la lecture nécessite la connaissance de la typographie et de la mise en page du document. La démarche n'est plus un simple prétraitement, mais une démarche experte d'analyse de document : il y a localisation des régions, séparation des régions graphiques et photographique, étiquetage sémantique des zones textuelles à partir de modèles, détermination de l'ordre de lecture et de la structure du document.
- Reconnaissance de l'imprimé ou du manuscrit : Les approches diffèrent selon qu'il s'agisse de reconnaissance de caractères imprimés ou manuscrits. Les caractères imprimés sont dans le cas général alignés horizontalement et séparés verticalement, c'est la phase de lecture [9]. La forme des caractères est définie par un style calligraphique (fonte) qui constitue un modèle pour l'identification.

  Dans le cas du manuscrit, les caractères sont souvent ligaturés et leur graphisme est inégalement proportionné provenant de la variabilité intra et interscripteurs. Cela nécessite généralement l'emploi de techniques de délimitation spécifiques et souvent des connaissances contextuelles pour guider la lecture [12].

Dans le cas de l'imprimé, la reconnaissance peut être monofonte, multifonte ou omnifonte :

- Un système est dit monofonte s'il ne peut reconnaître qu'une seule fonte à la fois, c'est à dire qu'il ne connaît de graphisme que d'une fonte unique. C'est le cas le plus simple de reconnaissance de caractères imprimés [13].

- Un système est dit multifonte s'il est capable de reconnaître divers types de fontes parmi un ensemble de fontes préalablement apprises [9].
- Un système omnifonte est capable de reconnaître n'importe quelle fonte, généralement sans apprentissage préalable. Cependant ceci est quasiment impossible car il existe des milliers de fontes dont certaines illisibles par l'homme (à l'exception de son concepteur) et avec un logiciel de création de fonte n'importe qui peut concevoir des fontes à sa guise.

Dans le cas du manuscrit, la reconnaissance peut être mono-scripteur, multi-scripteur ou omni-scripteur. L'écriture manuscrite hors-ligne peut être classée en deux catégories d'écritures : écriture cursive et écriture semi-cursive.

- Un système est dit Mono-scripteur (propre au scripteur) : c'est le fait que le système ne peut reconnaître qu'une seule écriture. Tous ces éléments influent sur la forme des lettres (écriture penchée, bouclée, arrondie, linéaire, etc.) et bien sûr sur la forme des ligatures, compromettant parfois le repérage des limites entre lettres.
- Un système est dit Multi-scripteur (propre à l'écriture manuscrite): c'est que le système peut identifier et reconnaître l'écriture pour un certain nombre de scripteurs.
- Un système est dit Omni-scripteur (propre à n'importe quelle écriture manuscrite): c'est le fait de réduire l'information contenue dans l'image au minimum nécessaire pour modéliser précisément la structure des caractères [14].

1.4 Approches de reconnaissance

Deux approches s'opposent en reconnaissance des mots : globale et analytique.

1.4.1 Approche globale

L'approche globale se base sur une description unique de l'image du mot, vue comme une entité indivisible. Disposant de beaucoup d'informations, en effet, la discrimination de mots proches est très difficile, et l'apprentissage des modèles nécessite une grande quantité d'échantillons qui est souvent difficile à réunir.

1.4.2 Approche analytique

L'approche analytique basée sur un découpage (segmentation) du mot. La difficulté d'une telle approche a été clairement évoquée par Sayre en 1973 et peut être résumée par le dilemme suivant : "pour reconnaître les lettres**,** il faut segmenter le tracé et pour segmenter le tracé**,** il faut reconnaître les lettres". Il s'ensuit qu'un processus de reconnaissance selon cette approche doit nécessairement se concevoir comme un processus de relaxation alternant les phases de segmentation et d'identification des segments. Cette approche est la seule applicable dans le cas de grands vocabulaires [15].

1.5 Processus de reconnaissance

Un système de reconnaissance fait appel généralement aux étapes suivantes : acquisition, prétraitement, segmentation, extraction des caractéristiques, classification, suivies éventuellement d'une phase de post-traitement.

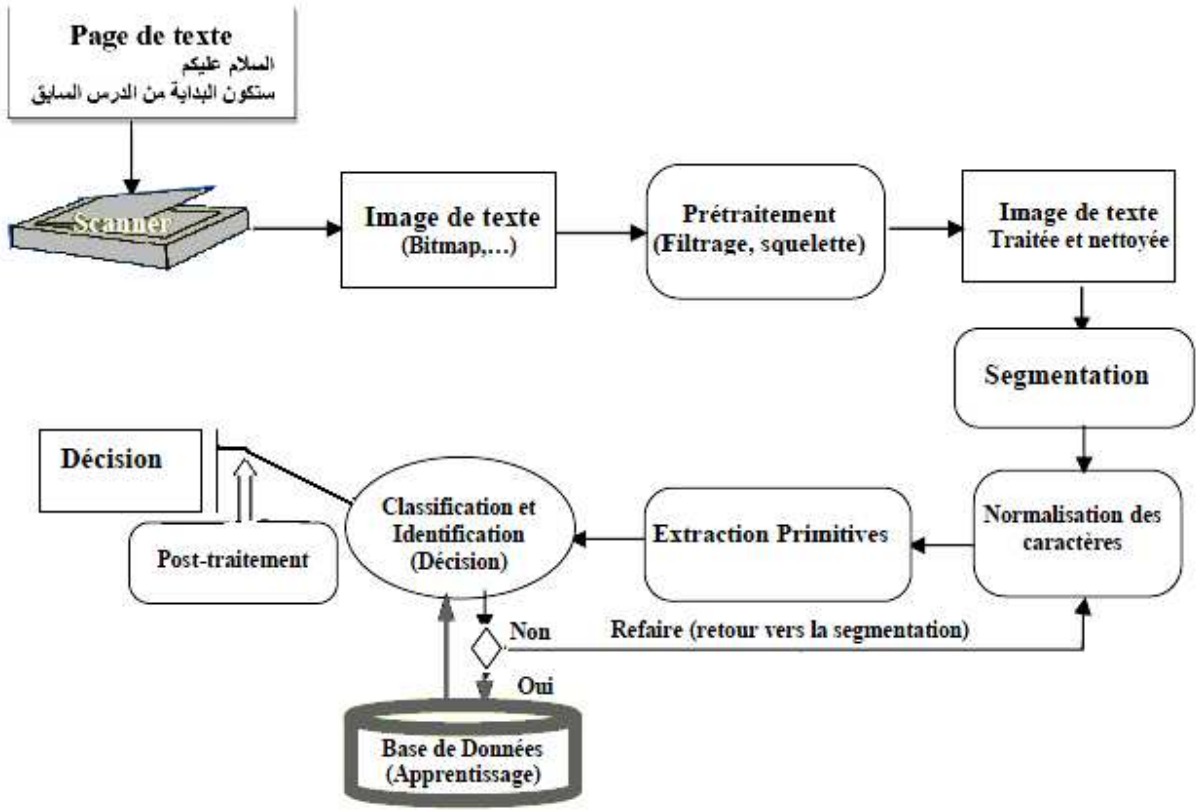


Figure 1.4 Schéma général du système de reconnaissance des caractères.

1.5.1 Phase d'acquisition

Cette phase consiste à capter l'image d'un texte au moyen des capteurs physiques (scanner, caméra,...) et de la convertir en grandeurs numériques adaptées au système de traitement, avec un minimum de dégradation possible.

1.5.1.1 Etape d'acquisition

Elle consiste en deux phases :

- **Echantillonnage (numerisation)** d'une image est spatial, par découpage en pixels.
- **Quantification (codage)** : est une valeur numérique donnée à l'intensité lumineuse, c'est un niveau de gris, appelé la dynamique de l'image.

Cette dynamique est donnée comme suit : 2^m, où m est le nombre de bits. Par exemple : le niveau de gris 256 est codé sur 8 bits, l'image couleur est codée sur 24 bits (1 octet pour chaque couleur(R, V, B)) [16].

1.5.1.2 Notion de voisinage

Dans le domaine de traitement des images en distingue deux types de voisinage :

- Voisinage à 4 : ensemble de pixels 'p' qui ont un coté en commun avec le pixel considéré.
- Voisinage à 8 : ensemble de pixels 'p' qui ont au moins un point de liaison avec le pixel considéré.

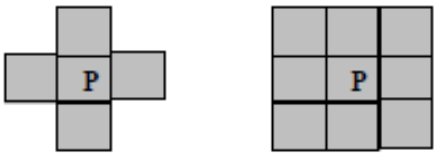


Figure 1.5 Voisinage à 4 et Voisinage à 8

1.5.2 Phase de prétraitement

Le prétraitement consiste à préparer les données issues du capteur à la phase suivante. Il s'agit essentiellement de réduire le bruit superposé aux données et essayer de ne garder que l'information significative de la forme représentée. Le bruit peut être dû aux conditions

d'acquisition (éclairage, mise incorrecte du document, ...) ou encore à la qualité du document d'origine.

Parmi les opérations de prétraitement généralement utilisées on peut citer : la binarisation, la dilatation, l'érosion, la squelettisation et la normalisation (figure 1.6).

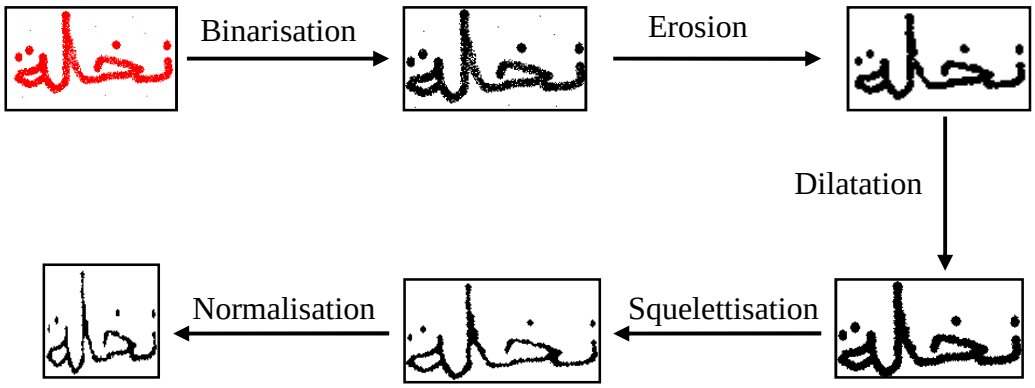


Figure 1.6 Effets de certaines opérations de prétraitement.

1.5.2.1 Binarisation

La binarisation c'est le passage d'une image en couleur, définie par plusieurs niveaux de gris en image bitonale (composée de deux valeurs 0 et 1) qui permet une classification entre le fond (image du support papier en blanc) et la forme (traits des gravures et des caractères en noir).

Plusieurs techniques ont été développeés dans le but de transformer une image à niveaux de gris ou en couleur, en image binaire. Toutes ces techniques sont basées sur le principe de seuillage comme le montre l'équation suivante :

$$I_b(x,y) = \begin{cases} 0 & \text{si } I_n(x,y) < T \\ 1 & \text{si } I_n(x,y) >= T \end{cases}$$

$I_n(x, y)$ décrit l'intensité à "n" niveaux de gris à chaque point de l'image, $I_b(x, y)$ représente l'intensité à deux niveaux et T est le seuil de binarisation. Si $I_n(x,y)$ est supérieur à la valeur de seuil alors on attribut le point image correspondant à la valeur d'intensité maximale (le blanc). Dans le cas contraire, le point est considéré comme noir et on lui attribut la valeur d'intensité minimale [17].

Pour des images de niveaux de gris, on peut trouver dans [18] une liste des méthodes de binarisation, proposant des seuils adaptatifs (ex. s'adaptant à la différence de distribution des niveaux de gris). Les auteurs dans [19] proposent une solution pour les images d'adresses postales.

La recherche du seuil passe par plusieurs étapes : binarisation préliminaire basée sur une distribution de mixture multimodale, analyse de la texture à l'aide d'histogrammes de longueurs de traits, et sélection du seuil à partir d'un arbre de décision.

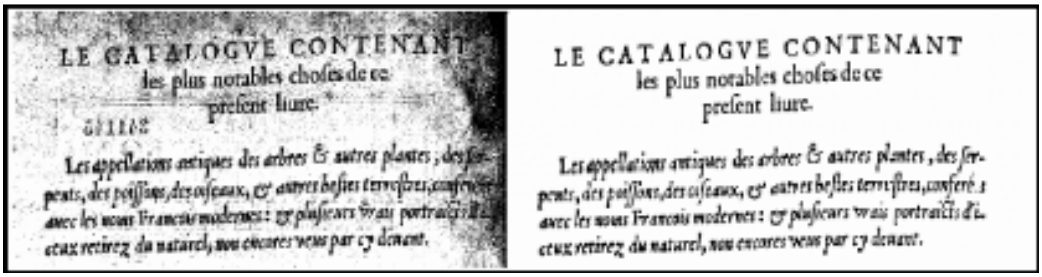


Figure 1.7 Exemple de Binarisation adaptative [20]

1.5.2.2 Transformation par érosion

Si un pixel 'p' est noir (p(x,y)=0), et il y a au moins 3 pixels, ou 7 pixels dans les 4-voisins, ou les 8-voisins respectivement, qui sont blancs ($P_{voisin}(x,y)=1$), alors on affecte à ce pixel la couleur blanche (p(x,y)=1), c-à-d. on efface ce pixel, (C'est un lissage d'un ou deux pixels d'une forme connexe) [17].

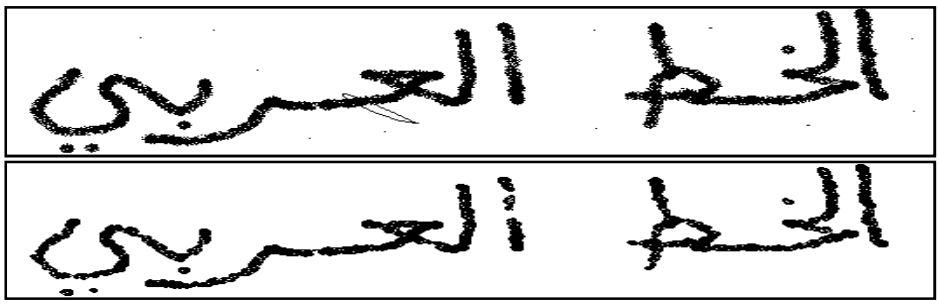


Image avant « érosion »

Image après « érosion »

Figure 1.8 Image traitée par érosion

1.5.2.3 Transformation par dilatation

C'est l'inverse de la transformation par érosion, si un pixel 'p' est blanc (p(x,y)=1), et il y a au moins 3 pixels, ou 7 pixels dans les 4-voisins, ou les 8-voisins respectivement, qui sont noir (Pvoisin(x,y)=0), alors on affecte à ce pixel la couleur noir (p(x,y)=0).

Image avant «Dilatation»

Image après «dilatation»

Figure 1.9 Image traitée par dilatation

1.5.2.4 Ouverture morphologique

C'est une combinaison d'opérations : érosion suivie d'une dilatation d'une image par le même élément structurant. Servant à adoucir les contours, simplifier les formes en lissant les bosses tout en conservant l'allure globale [17].

Figure 1.10 Image traitée par Ouverture

1.5.2.5 Fermeture morphologique

C'est le contraire de l'ouverture, elle consiste en une combinaison d'opérations : une dilatation suivie d'érosion d'une image par le même élément structurant. Elle permet de simplifier les formes, en comblant les creux.

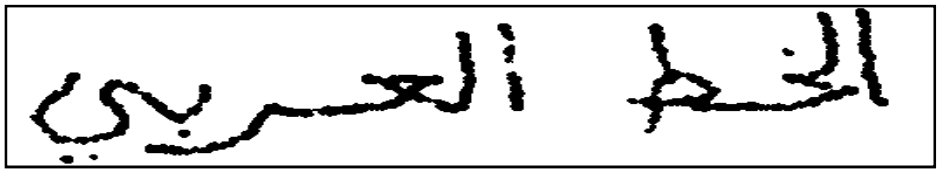


Figure 1.11 Image traitée par Fermeture

1.5.2.6 Squelettisation d'une image

Très utilisée dans le domaine de la ‘reconnaissance des formes’, cette opération consiste à transformer une image binaire en un ‘squelette’. Le squelette est un ensemble de lignes d’épaisseurs infiniment petites. La squelettisation doit préserver la connexité de l’image. En d’autres termes, cette opération ne doit ni séparer les éléments connexes, ni raccorder les éléments non connexes. Le but est de simplifier l'image du caractère en une image à « ligne » plus facile à traiter en la réduisant au tracé du caractère [21].

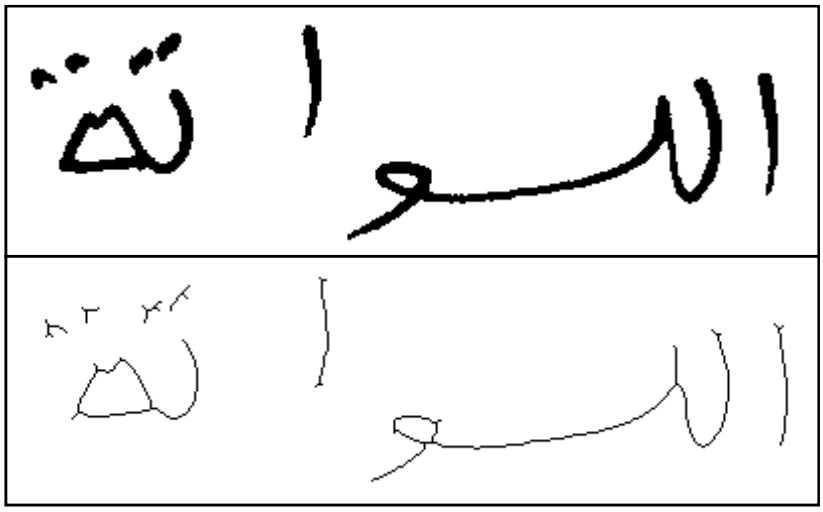


Figure 1.12 La squelettisation d'une Image [1]

Il existe plusieurs techniques de squelettisation parmi celle-ci nous citons une technique de suivi appelée semi-squelettisation. Cette technique est basée sur la détection pour chaque colonne de pixels: les points de début, milieu et fin du tracé de l'écriture (figure 1.13).

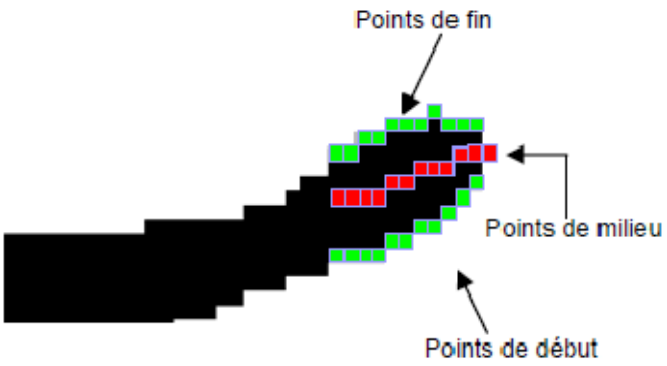


Figure 1.13 Technique de semi-squelettisation [2].

Les points de milieu extraits sont utilisés pour calculer au cours du balayage la longueur et l'angle de déviation du segment en cours de traitement tandis que les points d'extrémités sont utilisés pour extraire des informations sur la forme du segment **[2]**.

1.5.2.7 Normalisation

L'étude de discernement nécessite l'élimination des conditions qui peuvent fausser les résultats, comme la différence de taille. Il faut donc aboutir à la normalisation de la taille des caractères. Après cette opération, les images de tous les caractères se retrouvent définies dans une matrice de même taille, pour faciliter les traitements ultérieurs.

Cette opération introduit généralement de légères déformations sur les images. Cependant certains traits caractéristiques tels que la hampe dans des caractères (ا ل ظ ط par exemple) peuvent être éliminées à la suite de la normalisation, ce qui peut entraîner à des confusions entre certains caractères [22].

Une application doit être définie pour effectuer cette tache. Cette application doit être réversible pour pouvoir aller d'une taille à une autre et revenir par la suite.

$$G : R^4 \longrightarrow \text{ensemble des couleurs}$$

$$(x, y) \longrightarrow \text{couleur (noir/blanc)}$$

G est défini de la façon suivante :

$$\begin{cases} \mathbf{G(x,y) = noir} & \mathbf{si\ pixel\ (\mu 1\ x, \mu 2\ y) = noir} \\ \mathbf{G(x,y) = blanc} & \mathbf{sinon} \end{cases}$$

Tel que : $\mu 1$ = (largeur du cadre du caractère)/ (largeur du cadre de la normalisation)

$\mu 2$ = (hauteur du cadre du caractère)/ (hauteur du cadre de la normalisation)

1.5.3 Phase de segmentation

La segmentation est une opération appliquée à l'image qui consiste à subdiviser une scène réelle, en parties constituantes ou objets, en projetant une scène réelle sur un plan. Elle est la première opération à réaliser dans « la reconnaissance des formes ». Il faut donc, disposer d'un certains nombres d'attributs, représentatifs des régions que l'on cherche à extraire, pour procéder à la classification individuelle des points [6].

1.5.3.1 Techniques de la segmentation

Il existe deux techniques permettant de mettre en œuvre la segmentation. La première, connue sous le nom de segmentation implicite, et la deuxième c'est la segmentation explicite.

a- Segmentation implicite

Les méthodes de segmentation implicite s'inspirent des approches utilisées dans le domaine de la parole, où le signal est divisé en intervalles de temps réguliers, et procèdent à une sur-segmentation importante de l'image du mot à pas fixe (un ou quelques pixels). Cela permet d'assurer un taux de présence important des points de liaison entre lettres considérées.

La segmentation s'effectue pendant la reconnaissance qui assure son guide. Le système recherche dans l'image, des composantes ou des groupements de graphèmes[1] qui correspondent à ses classes de lettres [23]. Classiquement, il peut le faire de deux manières :

- soit par fenêtrage : le principe est d'utiliser une fenêtre mobile de largeur variable (qui n'est pas facile à déterminer) pour trouver des séquences de points de segmentation potentiels qui seront confirmés ou non par la reconnaissance de caractères. Elle nécessite deux étapes : la génération d'hypothèses de segmentation (séquences de points obtenus par le fenêtrage) ; la deuxième est le choix de la meilleure hypothèse de la reconnaissance (validation).
- soit par recherche de primitives : il s'agit de détecter les combinaisons de primitives qui donneront la meilleure reconnaissance.

b- Segmentation explicite

Cette approche, souvent appelée dissection, est antérieure à la reconnaissance et n'est pas remise en cause pendant la phase de reconnaissance. Les hypothèses des caractères sont déterminées à partir des informations de bas niveau présentes sur l'image. Ces hypothèses sont définitives, et doivent être d'une grande fiabilité car la moindre erreur de segmentation remet en cause la totalité des traitements ultérieurs.

Les approches de segmentation explicite, s'appuient sur une analyse morphologique du mot manuscrit pour localiser des points de segmentation potentiels. Elles sont

[1] La plus petite unité distinctive de l'écriture.

particulièrement adaptées à l'analyse de la représentation bidimensionnelle et donc plus souvent utilisées dans les systèmes de reconnaissance hors-ligne de mots. Certaines méthodes de segmentation explicite sont basées sur une analyse par morphologiques mathématiques, exploitent les concepts de régularité et singularité du tracé, analyse des contours supérieurs/inferieurs du mot. Les points de segmentation potentiels détectés sont confirmés à l'aide de diverses heuristiques [23].

1.5.3.2 Etapes de segmentation

a- Segmentation du texte en lignes

Les méthodes de traitement de l'arabe utilisent souvent la projection horizontale pour extraire les lignes. Cependant la présence des points diacritiques complique cette extraction et conduit parfois à la confusion des lignes [24]. Ce problème a lieu quand l'interligne est calculée par une simple moyenne des différentes interlignes. Pour remédier à ce problème, certains auteurs tels que [25] identifient d'abord les différentes lignes d'écriture, ensuite regroupent les blocs de texte d'après leur proximité par rapport aux lignes d'écriture déjà localisées.

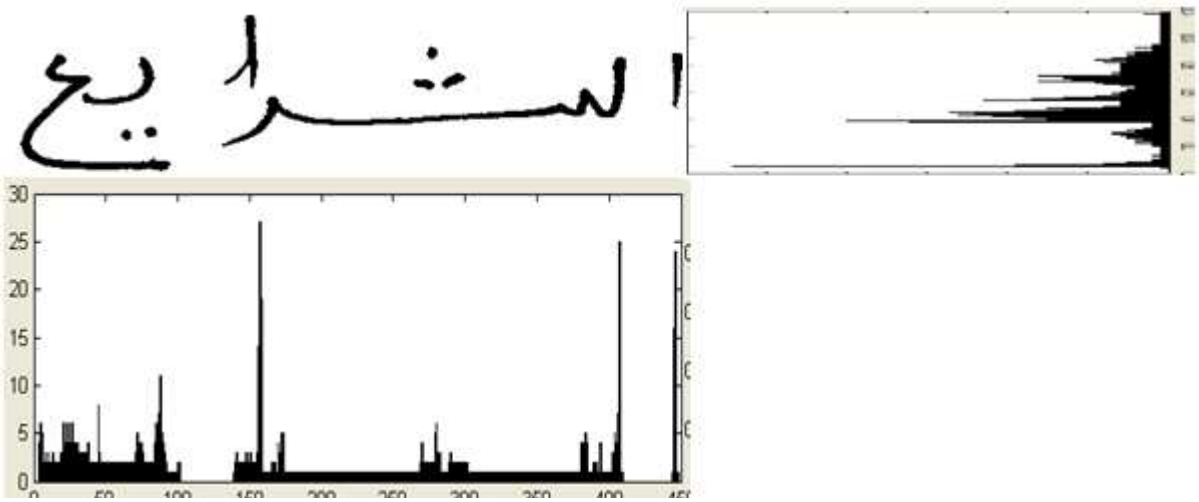


Figure 1.14 Exemple d'histogramme horizontal et vertical d'une ligne de texte [1].

Comme pour le latin, une fusion des lignes est aussi possible à cause des hampes et des jambages. En cas de fusion, une méthode empirique de correction consiste à localiser d'abord la ligne qui contient le maximum de pixels noirs [23]. Les parties au dessus et en dessous de

cette ligne sont ensuite analysées en se basant sur les densités de pixels noirs des différentes lignes.

Si la fusion a eu lieu par exemple dans la partie supérieure, la ligne ayant le minimum de densité de pixels dans cette partie, correspond à la frontière entre les lignes fusionnées.

b- Segmentation en Pseudo mot (PAW : Peace of Arabic Word)

Elle est réalisée en déterminant l'histogramme des projections verticales des différentes lignes de texte [5]. Cependant, cette méthode n'est pas efficace dans le cas où les PAWs se chevauchent verticalement. Dans ce cas, d'autres techniques sont utilisées telles que la détermination du contour [26], du squelette [23], ou encore des composantes connexes [27]. Le choix de la technique est souvent guidé par la méthode d'analyse [23][5].

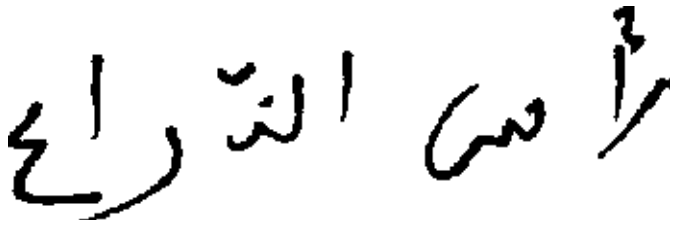


Figure 1.15 Exemple de chevauchement de PAWs respectivement de droite à gauche entre «رأ» et «راع».

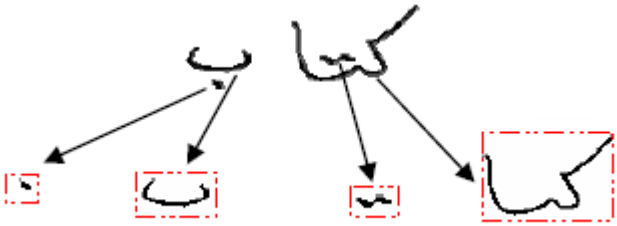


Figure 1.16 Exemple de segmentation [2]

c- Segmentation de la ligne de texte en mots

En arabe OCR, la segmentation est souvent réservée à l'extraction des PAWs, le mot est plutôt considéré dans la phase de post-traitement (si elle est prévue) pour valider les résultats trouvés ou corriger les erreurs de reconnaissance. Par ailleurs afin d'éviter le problème d'une segmentation erronée en mots, certains auteurs introduisent dans leurs systèmes, un seul mot à la fois [23],[5].

d- Segmentation du mot en caractères

La segmentation en caractères (ou en graphèmes) constitue le problème le plus ardu lié à la reconnaissance de l'écriture arabe. Les difficultés à ce niveau sont du même type que celles affrontées lors de la reconnaissance du latin manuscrit (cursif), mais souvent plus complexes à cause de la diversité des formes du caractère arabe, de la courte liaison qui existe entre les caractères successifs, de l'allongement des ligatures horizontales et de la présence des ligatures verticales [5],[28].

1.5.4 Phase d'extraction des caractéristiques

C'est l'une des étapes les plus délicates et les plus importantes en OCR. Les types des caractéristiques peuvent être classés en quatre groupes principaux : caractéristiques structurelles, caractéristiques statistiques, transformations globales, et superposition des modèles et corrélation [29] [10].

1.5.4.1 Caractéristiques structurelles

Les caractéristiques structurelles décrivent une forme en termes de sa topologie et sa géométrie en donnant ses propriétés globales et locales. Parmi ces caractéristiques on peut citer :

- Les traits et les anses dans les différentes directions ainsi que leurs tailles.
- Les points terminaux.
- Les points d'intersections.
- Les boucles.
- Le nombre de points diacritiques et leur position par rapport à la ligne de base.
- Les voyellations et les zigzags (hamza).
- La hauteur et la largeur du caractère.
- La catégorie de la forme (partie primaire ou point diacritique, etc).

Plusieurs autres caractéristiques peuvent être tirées, suivant qu'ils soient extraites d'une courbe, un trait ou d'un segment de contour.

1.5.4.2 Caractéristiques statistiques

Les caractéristiques statistiques décrivent une forme en termes d'un ensemble de mesures extraites à partir de cette forme. Les caractéristiques utilisées pour la reconnaissance de textes arabes sont: le zonage (zonning), les caractéristiques de lieu géométrique (Loci) et les moments [29].

Le zonage consiste à superposer une grille (n×m) sur l'image du caractère et calculer pour chacune des régions résultantes, la moyenne ou le pourcentage de points en niveaux de gris, donnant ainsi un vecteur de taille (n×m) de caractéristiques.

La méthode Loci est basée sur le calcul du nombre de segments blancs et de segments noirs le long d'une ligne verticale traversant la forme, ainsi que leurs longueurs [10].

1.5.4.3 Transformations globales

Elles sont naturellement basées sur une transformation globale de l'image. La transformation consiste à convertir la représentation en pixels en une représentation plus abstraite pour réduire la dimension des caractères, tout en conservant le maximum d'informations sur la forme à reconnaître. Par exemple : la transformée de Hough, la transformée de Fourier, et les moments de Zernike [23].

Moments de Zernike

Les polynômes de Zernike ont été proposés la première fois en 1934. La formulation de ces moments semble être une des plus populaires, surpassant les solutions de rechange en termes de résilience de bruit, redondance de l'information et possibilités de reconstruction. Plusieurs études montrent également la supériorité des moments de Zernike par rapport à d'autres approches [30],[31].

Les moments de Zernike sont construits en utilisant un ensemble de polynômes complexes qui forment un ensemble orthogonal complet défini sur le disque unité avec : $(x^2 + y^2) \leq 1$

$$A_{mn} = \frac{m+1}{\pi} \sum_x \sum_y I(x,y)[V_{mn}(x,y)]\, dx\, dy \qquad (1.1)$$

Où m et n définissent l'ordre du moment et $I(x,y)$ le niveau de gris d'un pixel de l'image I sur laquelle on calcule le moment. Les polynômes de Zernike $V_{mn}(x,y)$ sont exprimés en coordonnées polaires :

$$V_{mn}(r,\theta) = R_{mn}(r)e^{-jn\theta} \tag{1.2}$$

Où $R_{mn}(r)$ est le polynôme radial orthogonal :

$$R_{mn}(r) = \sum_{s=0}^{\frac{m-|n|}{2}} (-1)^s \frac{(m-s)!}{s!\left(\frac{m+|n|}{2}-s\right)!\left(\frac{m-|n|}{2}-s\right)!} r^{m-2s} \tag{1.3}$$

Les moments A_{mn} sont invariants par rotation, translation et changement d'échelle (après normalisation de la taille de la forme).

Cette représentation est inversible, l'image peut être reconstruite de la manière suivante :

$$\hat{I}(x,y) = \lim_{N\to+\infty} \sum_{n=0}^{N} \sum_{m} A_{mn} V_{mn}(x,y) \tag{1.4}$$

L'ordre des moments possède une grande influence sur la conservation de l'information angulaire. Plus l'ordre est élevé et plus les variations angulaires décrites sont fines. La Figure 1.17 en donne une illustration.

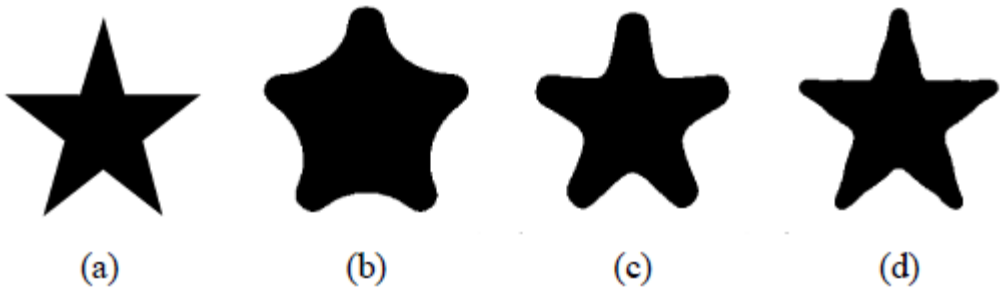


Figure 1.17 Exemple de reconstructions à partir des descripteurs de Zernike, (a) image d'origine, (b) reconstruction d'ordre 10, (c) reconstruction d'ordre 20, (d) reconstruction d'ordre 40 [32].

1.5.4.4 Superposition des modèles (template matching) et corrélation

La méthode de 'template matching' appliquée à une image binaire (en niveaux de gris ou squelettes), consiste à utiliser l'image de la forme comme vecteur de caractéristiques pour être comparé à un modèle (template) pixel par pixel dans la phase de reconnaissance, et une mesure de similarité est calculée [29].

1.5.5 Phase de classification

La classification dans un système OCR regroupe deux tâches : l'apprentissage et la reconnaissance (décision). A cette étape les caractéristiques de l'étape précédente sont utilisées pour identifier un caractère et l'attribuer à un modèle de référence [4].

1.5.5.1 Apprentissage

II s'agit lors de cette étape d'apprendre au système les propriétés pertinentes du vocabulaire utilisé et de l'organiser en modèles de références. L'idéal serait d'apprendre au système autant d'échantillons que de formes d'écritures différentes, mais cela est impossible à cause de la grande variabilité de l'écriture qui conduirait à une explosion combinatoire de modèles de représentation. La tendance consiste alors à remplacer le nombre par une meilleure qualité des traits caractéristiques [5],[4].

L'apprentissage consiste en deux concepts différents; l'entraînement et l'adaptation. L'entraînement consiste à enseigner au système la description des caractères tandis que l'adaptation sert à améliorer les performances du système en profitant des expériences précédentes. Certains systèmes permettent à l'utilisateur d'identifier un caractère lorsqu'ils échouent à le reconnaître et ils utilisent l'entrée de l'utilisateur à chaque fois que le caractère est rencontré [34] [35].

Les processus d'apprentissage sont différents selon qu'il s'agisse de reconnaissance de caractères imprimés ou manuscrits ou de reconnaître des textes monofonte ou multifonte. D'une manière générale, on distingue deux types de techniques d'apprentissage : supervisé et non supervisé.

- L'apprentissage est dit supervisé s'il est guidé par un superviseur appelé professeur. Il est réalisé lors d'une étape préliminaire de reconnaissance en introduisant un grand nombre d'échantillons de référence. Le professeur indique dans ce cas le nom de

chaque échantillon. Le choix des caractères de référence est fait à la main en fonction de l'application. Le nombre d'échantillons peut varier de quelques unités à quelques dizaines.

- L'apprentissage non supervisé ou sans professeur consiste à doter le système d'un mécanisme automatique qui s'appuie sur des règles précises de regroupement pour trouver les classes de référence avec une assistance minimale. Dans ce cas les échantillons sont introduits en un grand nombre par l'utilisateur sans indiquer leur classe [36].

1.5.5.2 Reconnaissance et décision

La décision est l'ultime étape de reconnaissance. A partir de la description en paramètres du caractère traité, le module de reconnaissance cherche parmi les modèles de référence en présence, ceux qui lui sont les plus proches.

Dans ce qui suit, nous présentons une brève revue de littérature de quelques modèles discriminants parmi les plus connus pour la reconnaissance de caractères manuscrits.

1.5.5.3 Décision Bayesienne

Dans une approche par modélisation, on cherche un modèle de production $p(x|c_i)$, qui donne pour chaque classe de formes c_i , la distribution des données qui lui sont associées. Dans une approche par discrimination, on cherche à approximer la distribution $p(c_i|x)$, qui représente la probabilité à posteriori des données étant donné la classe c_i . En pratique, cette information n'est pas toujours fournie [3].

La règle de décision bayesienne est une théorie clé en classification qui permet d'estimer la probabilité à posteriori à partir de la probabilité conditionnelle et d'émettre un vote d'appartenance de la forme traitée. Elle s'écrit :

$$P(c_i|x) = \frac{P(x|c_i)P(c_i)}{P(x)} \tag{1.5}$$

Pour k classes, la décision bayesienne cherche la classe c_i qui maximise la probabilité à posteriori. Elle s'écrit :

$$c(x) = arg\max P(c_i|x) \tag{1.6}$$

Le terme $P(x)$ représente la probabilité a priori de la classe c_i . Il est particulièrement utile si les classes ne sont pas balancées dans l'échantillon de données considéré. La vraisemblance de l'observation $P(x)$, est une quantité constante qui peut être omise du processus de décision.

1.5.5.4 Méthode du plus proche voisin (KPPV)

L'algorithme KPPV affecte une forme inconnue à la classe de son plus proche voisin en la comparant aux formes stockées dans une classe de références nommée prototypes. Il renvoie les K formes les plus proches de la forme à reconnaître suivant un critère de similarité. Une stratégie de décision permet d'affecter des valeurs de confiance à chacune des classes en compétition et d'attribuer la classe la plus vraisemblable (selon la métrique choisie) à la forme inconnue [5].

Cette méthode présente l'avantage d'être facile à mettre en œuvre et fournit de bons résultats. Son principal inconvénient est lié à la faible vitesse de classification due au nombre important de distances à calculer.

1.5.5.5 Réseaux de neurones

Les réseaux de neurones ont connu un essor important grâce à l'algorithme de rétropropagation du gradient de l'erreur attribué à Werbos [37], Rumelhart [38] et Lecun [39]. Ce classifieur a trouvé application dans beaucoup de domaines tels que la reconnaissance de caractères, la reconnaissance de visages, la classification d'expressions de gènes, etc. Il est capable d'inférer n'importe quelle fonction de décision non linéaire moyennant une seule couche de neurones cachées et des fonctions d'activation sigmoïdales [40].

En OCR, les primitives extraites sur une image d'un caractère (ou de l'entité choisie) constituent les entrées du réseau. La sortie activée du réseau correspond au caractère reconnu. Le choix de l'architecture du réseau est un compromis entre la complexité des calculs et le taux de reconnaissance [36].

1.5.6 Phase de post traitement

L'objectif du post-traitement est l'amélioration du taux de reconnaissance des mots (par opposition au taux de reconnaissance du caractère). Cette phase est souvent implémentée comme un ensemble d'outils relatifs à la fréquence d'apparition des caractères dans une chaîne, aux lexiques et à d'autres informations contextuelles.

Comme la classification peut aboutir à plusieurs candidats possibles, le post-traitement a pour objet d'opérer une sélection de la solution en utilisant des niveaux d'informations plus élevés (syntaxiques, lexicales, sémantiques...) [5]. Le post-traitement se charge également de vérifier si la réponse est correcte (même si elle est unique) en se basant sur d'autres informations non disponibles au classifieur.

1.6 Quelques travaux réalisés dans ce domaine

La reconnaissance de l'écriture arabe remonte aux années 70. Depuis, plusieurs solutions ont été proposées. Nous présentons ici quelques récents travaux réalisés dans ce Domaine.

Dans [41] Somaya Alma’adeed et al ont proposé un système de reconnaissance qui se base sur les MMC (modèles de Markov cachés), et qui a permis d'avoir un taux de reconnaissance de 45.0% sans post-traitement.

Mustapha Kadri dans [1] présente un système complet de reconnaissance de l’écriture arabe manuscrite hors-ligne, en utilisant le réseau de neurones SPIKE (SNN) (a impulsion) et les Séparateur à Vaste Marge (SVM). Les taux de reconnaissance qu’il obtient sont 76% pour la méthode SVM, et de 69% pour la méthode SNN.

Un autre système AOCR basé sur une compression ondelette, a donné un taux de 80.0% au détriment du temps d'analyse [42].

Le système décrit en [43], basé en DWT a permis d'atteindre un taux de reconnaissance de l'ordre de 90.0%, avec un taux relativement faible pour les caractères au milieu. Ces résultats sont toujours au détriment de la vitesse du système due à l'utilisation des ondelettes.

On trouve dans [44] un système appelé ASCA combiné avec un système déjà existant RECAM (ASCA-RECAM), basé sur une analyse morphologique du contour du mot. Les caractéristiques topologiques du texte sont exploitées pour extraire des règles morphologiques.

Dans [2] une approche de segmentation appliquée aux caractères manuscrits arabes, qui permet de reconstruire en hors-ligne un chemin de traçage similaire à celle dans le cas d'en-ligne. Avec l'utilisation d'une technique de semi-squelettisation pour le suivi des tracés et le calcul des caractéristiques des caractères. Avec l'application d'un classifieur SVM dans la phase de classification ils ont atteint des taux de reconnaissance intéressants en des temps réduits.

1.7 Conclusion

Nous avons présenté dans ce chapitre le concept de reconnaissance des caractères de façon générale, en mettant le point sur les caractéristiques morphologique de l'écriture Arabe et les différents aspects d'un OCR ainsi que les différentes phases de processus de reconnaissance. En fin, nous avons présenté quelques récents travaux réalisés dans le domaine de la reconnaissance de l'écriture arabe.

Nous avons vu que la cursivité de l'écriture arabe montre une complexité de la morphologie des caractères**.** Face à ce problème, la nécessité d'une modélisation robuste et une méthode d'apprentissage efficace pour prendre en considération toutes les variations morphologiques de l'écriture arabe.

Pour cette raison notre choix s'est orienté vers le modèle SVM (machines à vecteurs de support) multi-classes qui a beaucoup de succès pour la résolution de problème d'apprentissage en général, et de l'OCR en particulaire. Tel est l'objectif du chapitre suivant.

Chapitre II

Machines à vecteurs de support multi-classes

2.1 Introduction

Les machines à vecteurs supports (SVM) introduites au début des années 90 et basé sur la théorie de l'apprentissage statistique, ont eu un grand succès dans différents domaines de l'apprentissage, notamment dans les OCR. En effet, elles sont largement répandues en apprentissage statistique et ont eu beaucoup de succès dans quasiment tous les domaines où elles ont été appliquées.

L'origine des machines à vecteurs de support (SVM) remonte à 1975 lorsque Vapnik et Chervonenkis proposèrent le principe du risque structurel et la dimension VC pour caractériser la capacité d'une machine d'apprentissage. A cette époque, ce principe n'a pas trouvé place et il n'existait pas encore un modèle de classification solidement appréhendé pour être utilisable. Il a fallu attendre jusqu'à l'an 1982 pour que Vapnik propose un premier classificateur basé sur la minimisation du risque structurel baptisé SVM [45]. Ce modèle était toutefois linéaire et l'on ne connaissait pas encore le moyen d'induire des frontières de décision non linéaires. En 1992, Boser et al. proposent d'introduire des noyaux non-linéaires pour étendre le SVM au cas non-linéaire [46]. En 1995, Cortes et al. proposent une version régularisée du SVM qui tolère les erreurs d'apprentissage avec pénalités [47] [3].

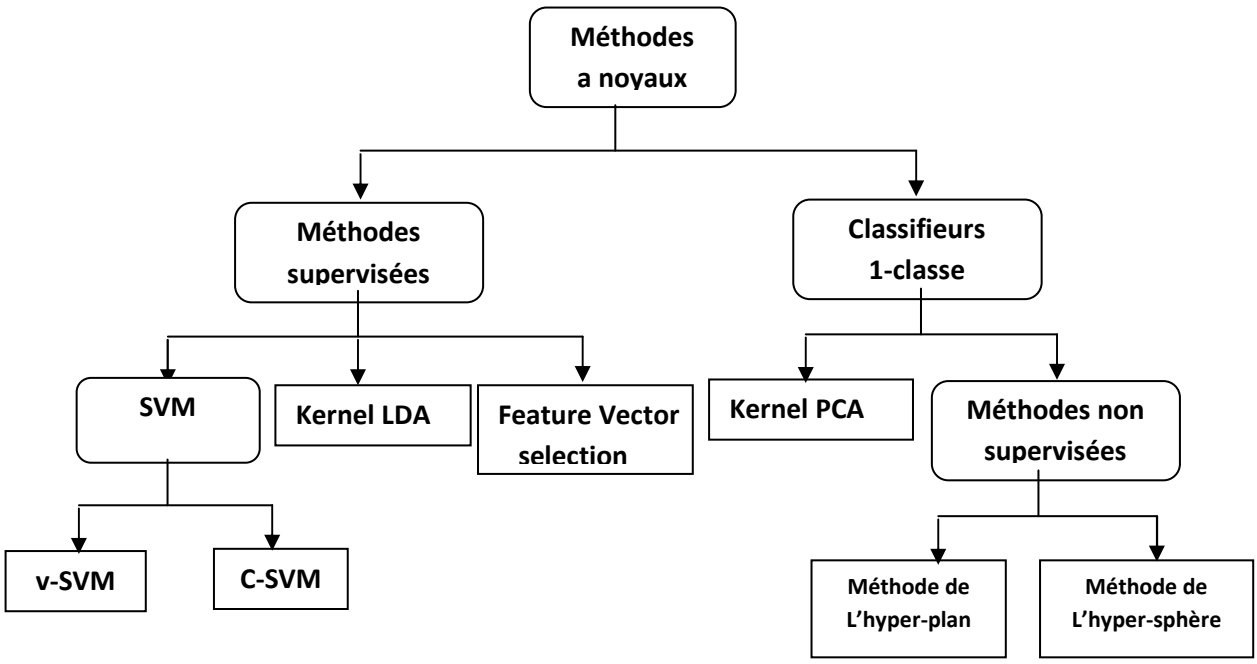


Figure 2.1 Arbre de classification des méthodes d'apprentissage à base de noyaux [3]

Les SVM multi-classes sont des modèles de l'apprentissage de conception relativement récente, dont l'étude est actuellement en plein essor. Ceci résulte en premier lieu du fait que la communauté des théoriciens, qui avait jusque dans un passé récent consacré l'essentiel de ses forces au développement de la théorie statistique du calcul des dichotomies, exprime à présent un intérêt de plus en plus marqué pour le cas multi-classe, dont elle perçoit mieux les spécificités. Cette situation nouvelle fait naître un besoin, celui de disposer d'une étude synthétique sur les SVM multi-classes, ou plus généralement l'utilisation de SVM pour la discrimination à catégories multiples.

2.2 Notion d'apprentissage

L'apprentissage c'est l'acquisition de connaissances et compétences permettant la synthèse d'information [48]. Un algorithme d'apprentissage va permettre de passer d'un espace des exemples X à un espace dit des hypothèses H. L'algorithme SVM va explorer l'espace H pour obtenir le meilleur hyperplan séparateur. L'apprentissage permet, à partir d'un ensemble de paramètres en entrée, d'obtenir un ensemble de résultats en sortie. On va donc préparer un jeu de données à apprendre, constitué de couples paramètres/résultats. Le but recherché est d'apprendre une fonction permettant de prédire de nouveaux résultats pour de nouvelles entrées.

L'apprentissage est dit supervisé ou non supervisé. Les SVM se situent dans le groupe des algorithmes d'apprentissage supervisés puisque que l'on utilise une base d'exemples pour obtenir la règle de classification.

2.3 Principe de fonctionnement général

Pour deux classes d'exemples données, le but des SVM est de trouver un classificateur qui va séparer les données et maximiser la distance entre ces deux classes. Avec SVM, ce classificateur est linéaire appelé « hyperplan » [49]. Dans le schéma qui suit, on détermine un hyperplan qui sépare deux ensembles de points.

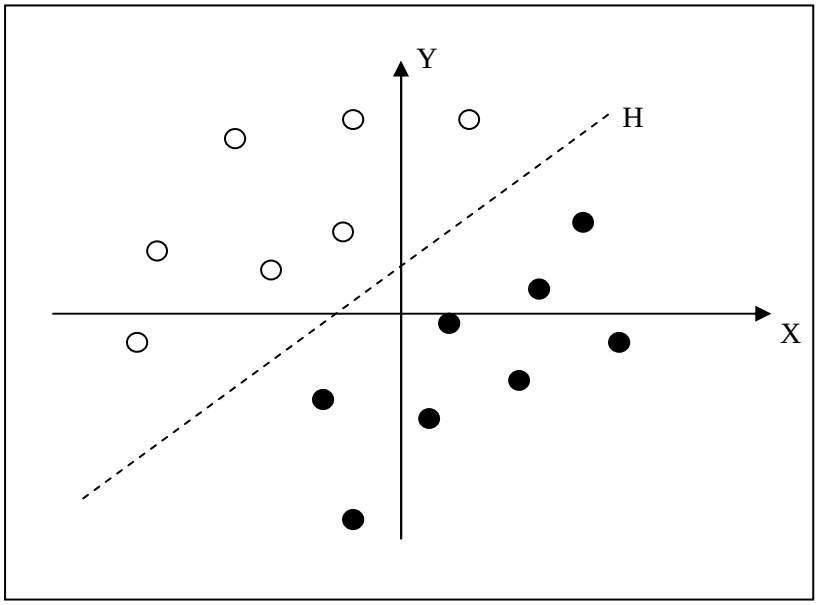


Figure 2.2 Séparation de deux ensembles de points par un Hyperplan H

Les points les plus proches, qui seuls sont utilisés pour la détermination de l'hyperplan, sont appelés vecteurs de support.

Il est évident qu'il existe une multitude d'hyperplan valides mais la propriété remarquable des SVM est que hyperplan doit être optimal. Nous allons donc en plus chercher parmi les hyperplans valides, celui qui passe « au milieu » des points des deux classes d'exemples.

Cela revient à chercher un hyperplan dont la distance minimale aux exemples d'apprentissage est maximale. On appelle cette distance « marge » entre l'hyperplan et les exemples.

L'hyperplan séparateur optimal est celui qui maximise la marge. Comme on cherche à maximiser cette marge, on parlera de séparateurs à vaste marge [50].

Dans le schéma qui suit, on détermine un hyperplan qui sépare les deux ensembles de points.

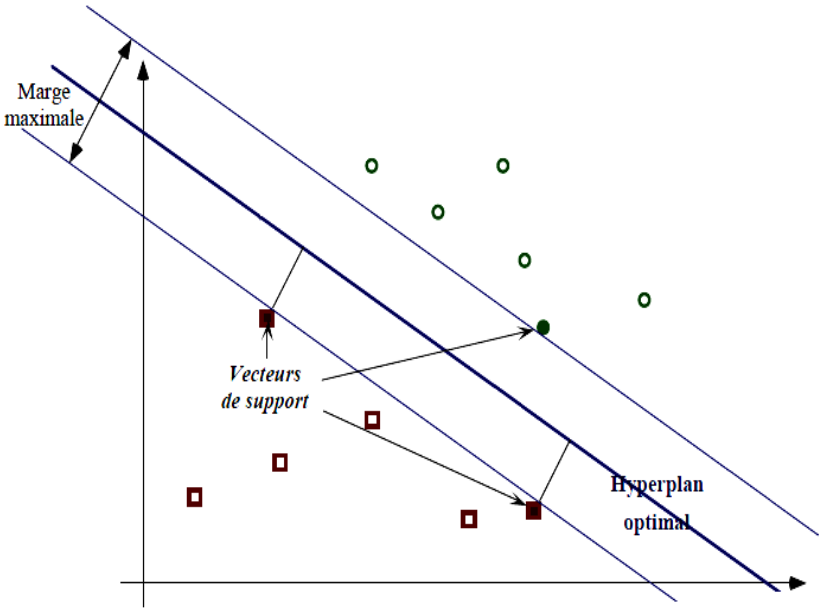


Figure 2.3 Hyperplan optimal, marge et vecteurs de support

Intuitivement, le fait d'avoir une marge plus large procure plus de sécurité lorsque l'on classe un nouvel exemple. De plus, si l'on trouve le classificateur qui se comporte le mieux vis-à-vis des données d'apprentissage, il est clair qu'il sera aussi celui qui permettra au mieux de classer les nouveaux exemples.

2.3.1 Linéarité et non linéarité

Dans les SVM, on constate deux cas de séparabilité :

2.3.1.1 Cas linéairement séparable

Dans un problème linéairement séparable les SVM trouvent facilement un séparateur (classificateur linéaire) qui maximise la marge.

2.3.1.2 Cas non linéairement séparable

Dans la plupart des problèmes réels il n'y a pas de séparation linéaire possible entre les données, le classificateur de marge maximale ne peut pas être utilisé car il fonctionne seulement si les classes de données d'apprentissage sont linéairement séparables [51] (Figure2.4).

Le principe consiste à projeter les données de l'espace d'entrée (appartenant à deux classes différentes) non linéairement séparables dans un espace de plus grande dimension appelé «espace de caractéristiques ou espace de redéscription» de façon à ce que les données deviennent linéairement séparables.

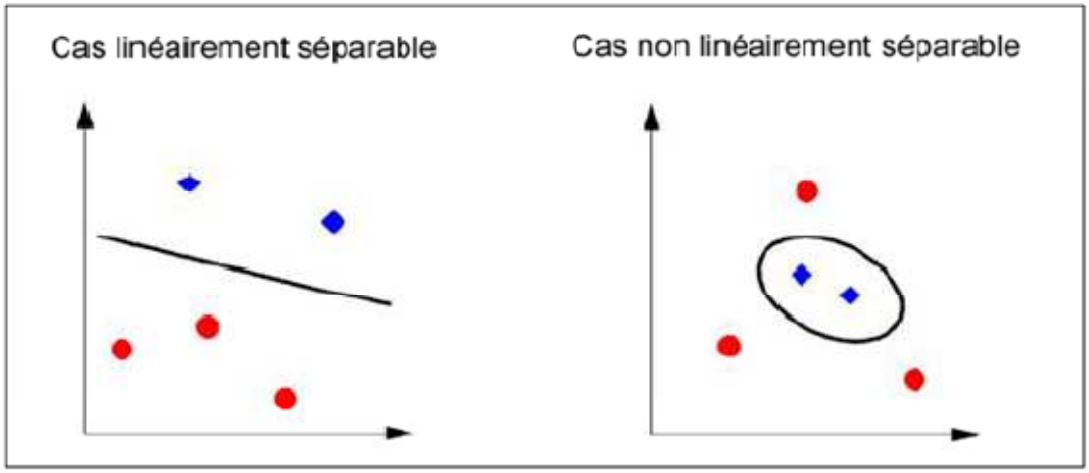


Figure 2.4 linéarité et non linéarité

On a donc une transformation d'un problème de séparation non linéaire dans un espace de représentation en un problème de séparation linéaire dans un espace de redéscription de plus grande dimension. Cette transformation est réalisée via une fonction noyau (Figure 2.5).

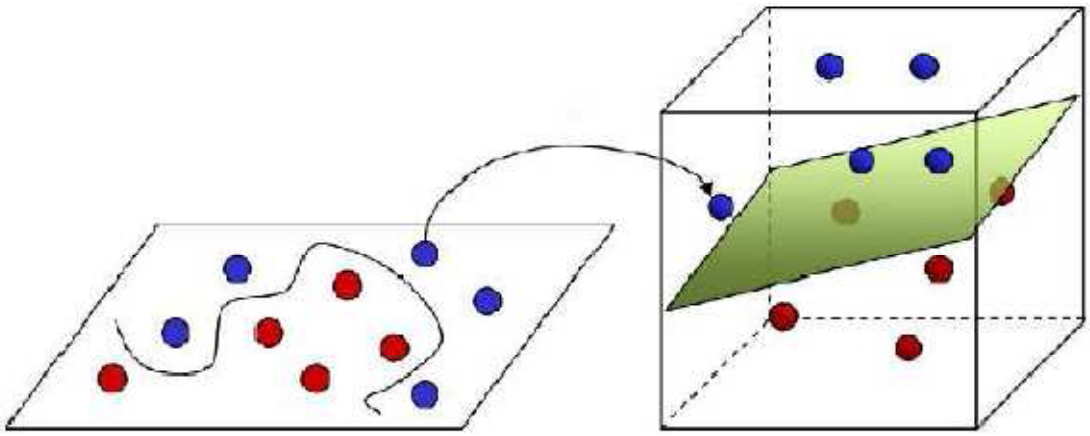


Figure 2.5 Transformation de l'espace de représentation et l'hyperplan séparateur dans le cas non linéairement séparables.

2.4 Fondements mathématiques

2.4.1 Principe

Les machines à vecteurs support forment une classe d'algorithmes d'apprentissage supervisés. Nous nous intéressons à une fonction notée f qui à toute entrée x fait correspondre une sortie $y = f(x)$. Le but est d'essayer d'apprendre f à partir d'un ensemble de couple (x_i, y_i). Dans ce problème les machines à vecteurs support vont être utilisées pour classifier une nouvelle observation x en se limitant à deux classes $y \in \{-1, 1\}$.

Nous allons donc construire une fonction f qui à chaque valeur d'entrée dans un ensemble R^d va faire correspondre une valeur de sortie $y \in \{-1, 1\}$:

$$f : R^d \rightarrow \{-1, 1\}, \qquad f(x) = y$$

Dans le cas linéaire, une fonction discriminante h est obtenue par combinaison linéaire d'un vecteur d'entrée $x = (x_1, \ldots, x_d)$ et s'écrit :

$$h(x) = w.x + b \tag{2.1}$$

La classe est donnée par le signe de $h(x)$: $f(x) = sign(h(x))$. Si $h(x) \geq 0$ alors x est de classe 1 sinon x est de classe -1. Le séparateur est alors un hyperplan d'équation : $w.x + b = 0$.

Si (x_i, y_i) est un des p éléments de la base d'apprentissage notée A_p , on veut trouver le classifieur h tel que :

$$y_i(w.x_i + b) \geq 0 \, , i \in [1, p] \tag{2.2}$$

Dans le cas simple linéairement séparable il existe de nombreux hyperplans séparateurs. Selon la théorie de Vapnik [52], l'hyperplan optimal est celui qui maximise la marge. Cette dernière étant définie comme la distance entre un hyperplan et les points échantillons les plus proches. Ces points particuliers sont les vecteurs supports. La distance entre un point x quelconque et l'hyperplan est donnée par l'équation 2.3.

$$d(x) = \frac{w.x + b}{\|w\|} \tag{2.3}$$

Donc maximiser la marge va revenir à minimiser $\|w\|$.

2.4.2 Forme primale

Les paramètres w et b étant définis à un coefficient multiplicatif près, on choisit de les normaliser pour que les échantillons les plus proches (x_s) vérifient l'égalité suivante :

$$y_s(w.x_s + b) = 1$$

Donc quelque soit l'échantillon x_i on obtient :

$$y_i(w.x_i + b) \geq 1 \tag{2.4}$$

La distance entre l'hyperplan et un point support est donc définie par $\frac{1}{\|w\|}$. La marge géométrique entre deux classes est égale à $\frac{2}{\|w\|}$. La forme primale (qui dépend seulement de *w* et *b*) des SVM est donc un problème de minimisation sous contrainte qui s'écrit :

$$\begin{cases} min\left(\frac{1}{2}\|w\|^2\right) \\ \forall(x_i, y_i) \in A_P, \;\; y_i(w.x_i + b) \geq 1 \end{cases} \tag{2.5}$$

2.4.3 Forme duale

La formulation primale peut être transformée en formulation duale en utilisant les multiplicateurs de Lagrange. L'équation 2.5 s'écrit alors sous la forme suivante :

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{P} \alpha_i(y_i(w.x_i + b) - 1) \tag{2.6}$$

La formulation de Lagrange permet de trouver les extremums en annulant les dérivées partielles de la fonction $L(w, b, a)$. Le lagrangien L doit être minimisé par rapport à w et b et maximisé par rapport à α. On résout ce nouveau problème en calculant les dérivées partielles :

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{P} \alpha_i y_i x_i = 0 \tag{2.7}$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{P} \alpha_i y_i = 0 \tag{2.8}$$

En réinjectant les deux premières dérivées partielles 2.7 et 2.8 dans l'équation 2.6 nous obtenons :

$$L(\alpha) = \frac{1}{2}\sum_{i=1}^{P} \alpha_i y_i \sum_{j=1}^{P} \alpha_j y_j x_i . x_j - \sum_{i=1}^{P} \alpha_i y_i \sum_{j=1}^{P} \alpha_j y_j x_i . x_j - \sum_{i=1}^{P} \alpha_i y_i b + \sum_{i=1}^{P} \alpha_i$$

On en extrait la formulation duale (dépendant des α_i) suivante :

$$L(\alpha) = \sum_{i=1}^{P} \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j x_i . x_j \tag{2.9}$$

On cherche donc à maximiser $L(\alpha)$ sous les contraintes $\alpha_i \geq 0$ et $\sum_i \alpha_i y_i = 0$. A l'optimal α^* les conditions de Karush Kuhn Tucker (conditions KTT) sont satisfaites et permettent d'écrire l'égalité suivante :

$$\alpha_i [y_i (w . x_i + b) - 1] = 0, \forall i \in [1, P] \tag{2.10}$$

Cela nous donne $a_i = 0$ ou $(w . x_i + b) - 1 = 0$. Ces deux possibilités impliquent que seuls les α_i associés à des exemples situés sur la marge peuvent être non nuls. Autrement dit, ces exemples sur la marge constituent les vecteurs supports, qui seuls contribuent à définir l'hyperplan optimal.

Cette maximisation est un problème de programmation quadratique de dimension égale au nombre d'exemple. L'équation 2.7 nous donne la valeur optimale pour w noté $w^* : w = \sum_{i=1}^{P} \alpha_i^* y_i x_i$, avec α_i^* les coefficients de Lagrange optimaux. En utilisant l'équation de l'hyperplan 2.1 nous obtenons l'hyperplan de marge maximale :

$$h(x) = \sum_{i=1}^{P} \alpha_i^* y_i x . x_i + b \tag{2.11}$$

2.5 Principe du noyau

Le cas linéairement séparable est peu intéressant, car les problèmes de classification sont souvent non linéaires. Pour le résoudre ce problème il suffit de projeter les données dans un espace de dimension supérieur appelé espace de redéscription noté H. L'idée étant qu'en augmentant la dimensionnalité du problème on se retrouve dans le cas linéaire vu précédemment. Nous allons donc appliquer une transformation non linéaire $\Phi(\cdot)$ aux vecteurs d'entrée x_i tel que $x_i \in R^d$ et $\Phi(x_i) \in R^e$, $(e > d)$. Ce changement va conduire à passer d'un produit scalaire dans l'espace d'origine $x_i . x_j$ à un produit scalaire $\Phi(x_i).\Phi(x_j)$ dans l'espace de redéscription (voir la figure 2.7).

Le principe est d'utiliser une fonction noyau notée K qui évite le calcul explicite du produit scalaire dans l'espace de redéscription.

Nous avons alors l'égalité suivante :

$$K(x_i, x_j) = \Phi(x_i).\Phi(x_j)$$

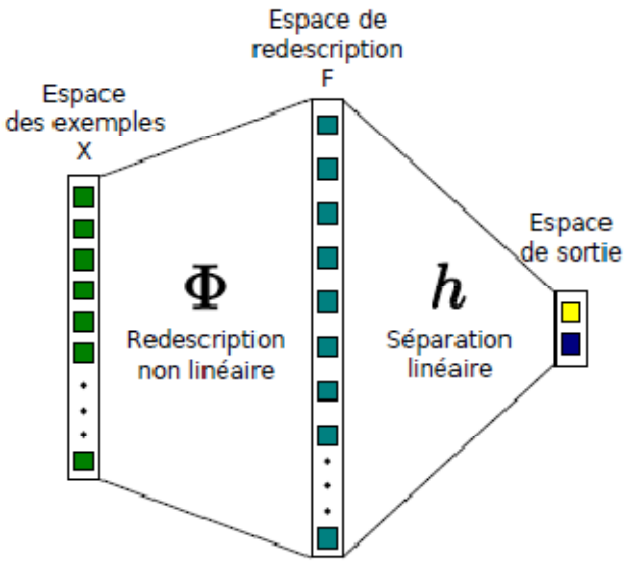


Figure 2.6 Transformation linéaire des données en une séparation linéaire dans un nouvel espace.

2.5.1 Exemple

Prenons le cas trivial où $x = (x_1, x_2)$ dans R^2 et $\Phi(x) = \left(x_1^2, \sqrt{2\, x_1 x_2}, x_2^2 \right)$ est explicite. Dans ce cas, H est de dimension 3 et le produit scalaire s'écrit :

$$\langle \Phi(x), \Phi(x') \rangle = x_1^2 x_1'^2 + 2\, x_1 x_2 x_1' x_2' + x_2^2 x_2'^2$$

$$= \left(x_1 x_1' + x_2 x_2'\right)^2$$

$$= \langle x, x' \rangle^2 = k(x, x')$$

Le calcul du produit scalaire dans H ne nécessite pas l'évaluation explicite de Φ. La fonction noyaux doit satisfaire la condition de Mercer présenté ci-dessous.

2.5.2 Condition de Mercer

Une fonction $k(.,.)$ symétrique est un noyau si, pour tous les x_i possibles, la matrice de terme général $k(x_i, x_j)$ est une matrice définie positive c'est-à-dire quelle définit une matrice de produit scalaire [36]. Dans ce cas, on montre qu'il existe un espace H et une fonction Φ tels que :

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle$$

Malheureusement, cette condition théorique d'existence est difficile à vérifier et, de plus, elle ne donne aucune indication sur la construction de la fonction noyau ni sur la transformation Φ. La pratique consiste à combiner des noyaux simples pour en obtenir des plus complexes (multidimensionnels) associés à la situation rencontrée [36].

2.5.3 Exemples de noyaux

- Linéaire : $K(x, x') = x.x'$
- Polynomial : $K(x, x') = (x.x')^d \; ou \; (c + x.x')^d$
- Gaussien : $K(x, x') = exp^{-\|x-x'\|^2/\sigma}$
- Laplacien : $K(x, x') = exp^{-\|x-x'\|/\sigma}$
- Sigmoïde : $K(x, x') = tanh(\alpha_0(x.x') + \beta_0)$,

Les noyaux gaussiens sont des noyaux dits de type radial (fonction à base radiale abrégée RBF [53]), indiquant qu'ils dépendent de la distance entre deux exemples. L'hyperplan séparateur se réécrit avec la fonction noyau sous la forme suivante :

$$h(x) = \sum_{i=1}^{P} \alpha_i^* y_i k(x.x_i) + b \qquad (2.12)$$

Beaucoup d'articles sont consacrés à la construction d'un noyau plus ou moins exotique et adapté à une problématique posée : reconnaissance de séquences, de caractères, l'analyse de textes... La grande flexibilité dans la définition des noyaux, permettant de définir une notion adaptée de similitude, confère beaucoup d'efficacité à cette approche à condition bien sur de construire et tester le noyau le plus approprié.

2.6 Marges souples

Les machines à vecteurs support sont efficaces quand le problème est séparable. Nous avons vu que l'utilisation d'une méthode noyau permettait de traiter les cas non linéaires mais cela n'est pas utilisable dans le cas de données non séparables par exemple pour des données bruitées. En effet, on peut avoir des éléments à classer du mauvais coté de l'hyperplan comme le montre la Figure 2.6. Cortes et Vapnik en 1995 ont donc introduit le concept de marge souple [47]. Certains exemples d'apprentissage peuvent violer la contrainte 2.4 que l'on retrouve dans l'équation de la forme primale (2.5).

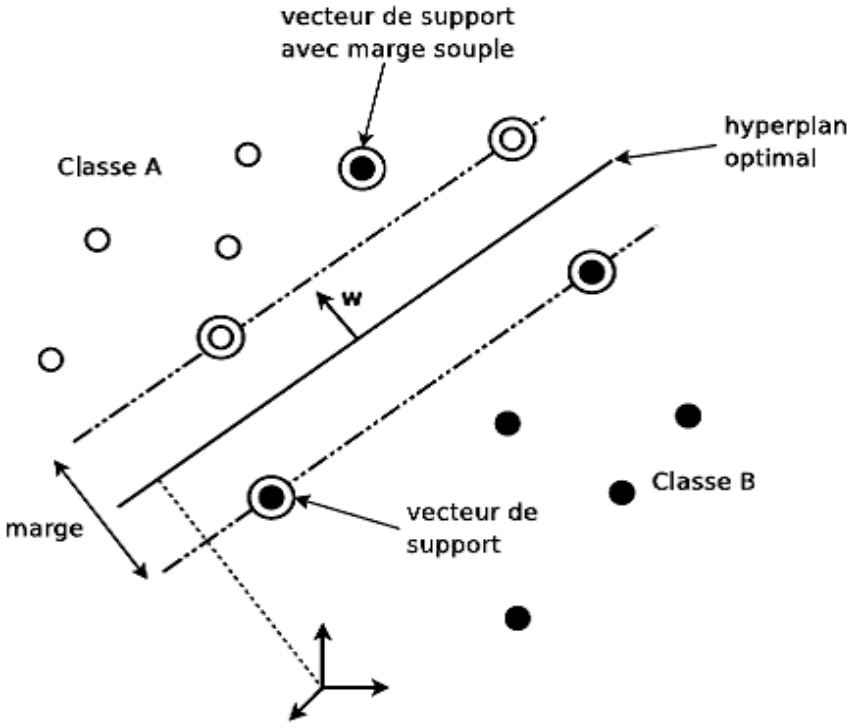


Figure 2.7 Hyperplan de séparation optimale avec marge souple dans un cas non linéairement séparable.

On introduit par conséquent des variables dites « ressorts » $\xi = \xi_1, \dots, \xi_P$ qui permettent d'assouplir la contrainte pour chaque exemple. Un paramètre supplémentaire de régularisation C est ajouté pour contrôler la pénalité associée aux exemples. La nouvelle forme primale décrite en 2.5 devient alors :

$$\begin{cases} \min \left(\frac{1}{2}\|w\|^2 + C \sum_{i=1}^{P} \xi_i \right) \\ \forall\, (x_i, y_i) \in A_P, \qquad y_i(w.x_i + b) \geq 1 - \xi_i \end{cases} \tag{2.13}$$

De même que pour la forme primale nous obtenons une nouvelle formulation duale qui est alors similaire à celle décrite précédemment. Si on utilise en plus la fonction noyau K dans la formulation duale 2.9 en appliquant la méthode des multiplicateurs de Lagrange on cherche alors à maximiser la nouvelle fonction $L(\alpha)$.

$$L(w, b, \xi, \alpha) = \frac{1}{2}w^2 + C \sum_{i=1}^{P} \xi_i - \sum_{i=1}^{P} \alpha_i [y_i(w.x_i + b) - 1 + \xi_i]$$

En appliquant la même méthode on obtient $L(\alpha)$ à partir de l'expression du Lagrangien précédent.

$$L(\alpha) = \sum_{i=1}^{P} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{P} \alpha_i \alpha_j y_i y_j K(x_i x_j) \tag{2.14}$$

$$\forall (x_i, y_i) \in A_P, 0 \leq \alpha_i \leq C \; et \; \sum_i y_i \alpha_i = 0$$

Le seul changement est la contrainte supplémentaire sur les coefficients α_i, qui se traduit par une borne supérieure C. La solution de l'équation précédente 2.14 est de la forme :

$$h(x) = \sum_{i=1}^{P} \alpha_i^* y_i K(x, x_i) + b \tag{2.15}$$

Remarque :

Le calcul des α_i^* du Lagrangien vu précédemment est obtenu par programmation quadratique. C'est un problème bien connu pour lequel il existe de nombreux algorithmes d'optimisation. Ce type de problème a un optimum global unique, élément important pour

éviter d'atteindre un optimum local [54]. Il s'ensuit un seul α^* qui maximise l'équation (2.14).

Une fois que l'on a obtenu les coefficients α^*, le coefficient b peut être obtenu en utilisant n'importe quel vecteur x_{i} avec l'égalité (2.10).

Des paragraphes précédents, on soulignera que la solution dépend uniquement d'un nombre restreint d'exemple de l'ensemble d'apprentissage et non pas de la dimension d de l'espace des exemples.

2.7 SVM multi-classes

A l'origine, les SVM ont été conçus essentiellement pour les problèmes à deux classes cependant plusieurs approches permettant d'étendre cet algorithme aux cas à N classes ont été proposées. Pour pouvoir traiter plus de deux classes, il convient d'apporter les modifications nécessaires.

Nous cherchons à modéliser une fonction $G : \Omega \rightarrow 1, \ldots, m$, qui définit m partitions dans l'espace des caractéristiques. Connaissant G, les partitions des classes sont définies par G^{-1}(k), où $k \in [1..m]$.

Pour un problème à deux classes, l'hyper-plan (w, b) du SVM délimite les deux partitions selon $signf(x)$ où $f(x) = w.x + b$. Par ailleurs, bien que le SVM soit un classifieur binaire, il peut facilement être étendu pour décider de l'appartenance de données multi-classes. En particulier, on trouve deux schémas de classification :

2.7.1 Approche Une-contre-Tous

Cette méthode est simple d'usage et donne des résultats raisonnables. Elle consiste à entraîner m SVM différents, séparant chaque classe des $m - 1$ restantes. Ainsi, pour chaque exemple de test, m valeurs de sortie $f_k(x)$ sont disponibles. Une façon simple de classification consiste à attribuer l'exemple à la sortie de plus grande amplitude.

L'extension au cas multi-classe originellement proposée par Vapnik [52] peut être vue aussi comme une généralisation du cas binaire. À toute classe k est associé un hyperplan $H_{(w_k, b_k)}$ défini par la fonction de décision $f_k(x) = \langle w_k \cdot x \rangle + b_k$ dont le rôle est de discriminer entre les observations de la classe k et de l'ensemble des autres classes.

Une observation x sera donc affectée à la classe k^* selon la règle de décision discrète,

$$k^* = \underset{1 \le k \le m}{\text{Arg max}}\ h_k(x) \tag{2.16}$$

Avec

$$h_k(x) = sign(f_k(x))$$

Afin de bien comprendre cette généralisation, considérons le cas binaire où $Y = \{-1, +1\}$. À chaque classe est associé un hyperplan défini par les fonctions de décision $f_k(x) = \langle w_k \cdot x \rangle + b_k, k = -1, +1$.

$$H_{+1} = \{x \in \mathbb{R}^p; f_{+1}(x) = 0\}$$

$$H_{-1} = \{x \in \mathbb{R}^p; f_{-1}(x) = 0\}$$

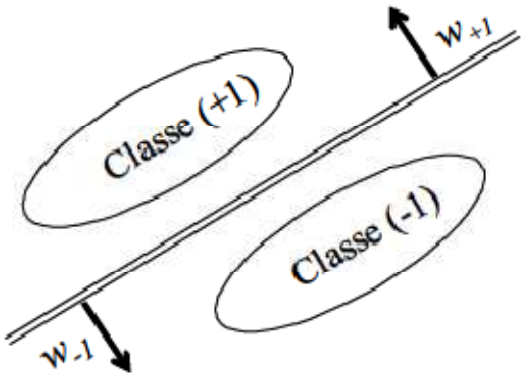


Figure 2.8 Association de classes aux hyperplans.

H_{+1} est associé à la classe (+1) et H_{-1} est associé à la classe (−1). Une illustration graphique est représentée par la figure 2.8.

Géométriquement, les deux hyperplans sont confondus, en revanche $f_{+1}(x) = -f_{-1}(x)$. Ainsi, si nous posons $w = w_{+1} - w_{-1}$ et $b = b_{+1} - b_{-1}$ nous pouvons réduire le problème en recherchant d'un seul hyperplan et c'est ce qu'on fait dans le cas binaire.

L'architecture la plus ancienne, utilisée pour les machines à vecteurs supports multi-classes, est probablement la méthode une-contre-reste. Elle construit m classifieurs binaires à vecteurs supports où m est le nombre total des classes. L'apprentissage du $k^{ème}$ classifieur à

vecteurs supports s'effectue en considérant tous les exemples de la $k^{ème}$ classe dans la région positive et tous les autres exemples dans la région négative. Le $k^{ème}$ classifieur à vecteurs supports s'obtient en résolvant le problème,

$$\begin{aligned} Minimiser_{w^k,\xi^k,b} \quad & \frac{\|w_k\|^2}{2} + C\sum_{j=1}^{l}\xi_j^k, \\ sous \quad & \langle w_k.\Phi(x_j)\rangle + b_k \geq 1 - \xi_j^k, \quad si\ y_j = k, \\ & \langle w_k.\Phi(x_j)\rangle + b_k \leq -1 + \xi_j^k, \quad si\ y_j \neq k, \\ & \xi_j^k \geq 0, \quad j = 1,2,\dots,l. \end{aligned} \tag{2.17}$$

Où les $\Phi(x_s)$ sont les transformées des x_s dans l'espace induit par la fonction Φ et C est le paramètre de pénalité.

La résolution du problème (2.17) pour chaque valeur de $k \in \{1,2,\dots,m\}$ donne lieu à m fonctions de décision :

$$f_k(x) = \langle w_k.\Phi(x)\rangle + b_k\,, \quad k \in \{1,2,\dots,m\} \tag{2.18}$$

Pratiquement, nous résolvons le problème dual correspondant au problème (2.17) ayant exactement l variables duales. En total, nous aurons à résoudre m problèmes quadratiques chacun à l variables. Ainsi, le temps d'apprentissage de cette méthode croît linéairement en fonction de m [55].

Une nouvelle observation x sera donc affectée à la classe k^* selon la règle de décision discrète (2.16). Dans le cas multi-classes ($m > 2$), cette égalité peut être satisfaite pour plus qu'une classe. Dans ce cas, l'observation x est dite non-classifiable. Toutes les observations x non-classifiables forment la région d'ambiguïté dite aussi région non-classifiable. Cette région est schématisée dans la figure 2.9.

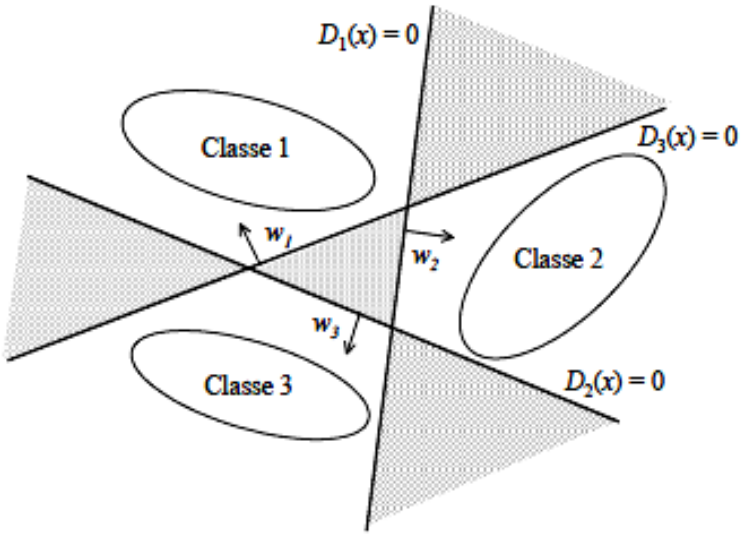


Figure 2.9 Représentation de la région d'ambiguïté (espace hachurée).

Afin de pouvoir classer une observation x tombant dans la région d'ambiguïté, la règle de décision continue a été utilisée. Cette règle est donnée par :

$$k^* = \underset{1 \le k \le m}{\text{Arg max}}\ f_k(x) \tag{2.19}$$

Géométriquement interprétée, tout nouvel exemple x est affecté à la classe qui correspond à l'hyperplan le plus éloigné. Ainsi, l'espace des variables explicatives X est subdivisé en m régions convexes, chacune correspondant à une classe. La figure 2.10 donne un exemple de subdivision de l'espace X.

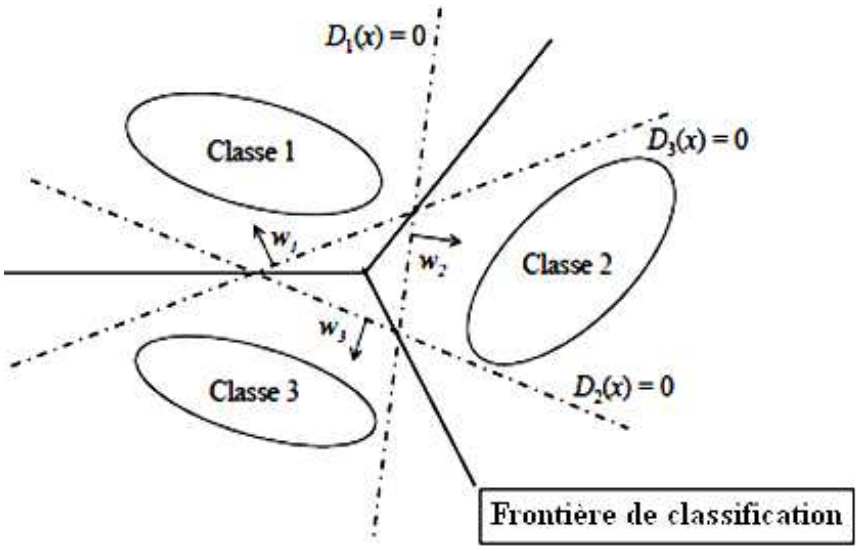


Figure 2.10 Règle de décision continue : bissectrices des secteurs d'ambiguïté formant la nouvelle frontière de classification.

Cette approche est nommée "*le gagnant emporte le tout*"[1]. L'inconvénient majeur de cette heuristique est qu'elle ne conserve pas les m frontières de séparation. La figure 2.10 montre ce propos. Il est clair que cette heuristique a amélioré la règle de décision discrète, en revanche, elle perd totalement les capacités de généralisation des m hyperplans construits. Malheureusement, on ne dispose pas de borne pour l'erreur de généralisation de l'approche une-contre-reste.

Des machines à vecteurs supports floues[2] (FSVM) ont été proposées par S. Abe dans [56] le but de fournir une règle de décision au niveau de la région d'ambiguïté. Le même auteur a prouvé dans [57] l'équivalence des FSVM avec la méthode standard "*le gagnant emporte le tout*".

Une autre heuristique utilisant les hyperplans séparateurs construits pour chaque paire de classes a été proposée afin de conserver le maximum des propriétés des classifications binaires et de réduire la région d'ambiguïté [55].

[1] The winner-takes-all.
[2] Fuzzy Support Vector Machines

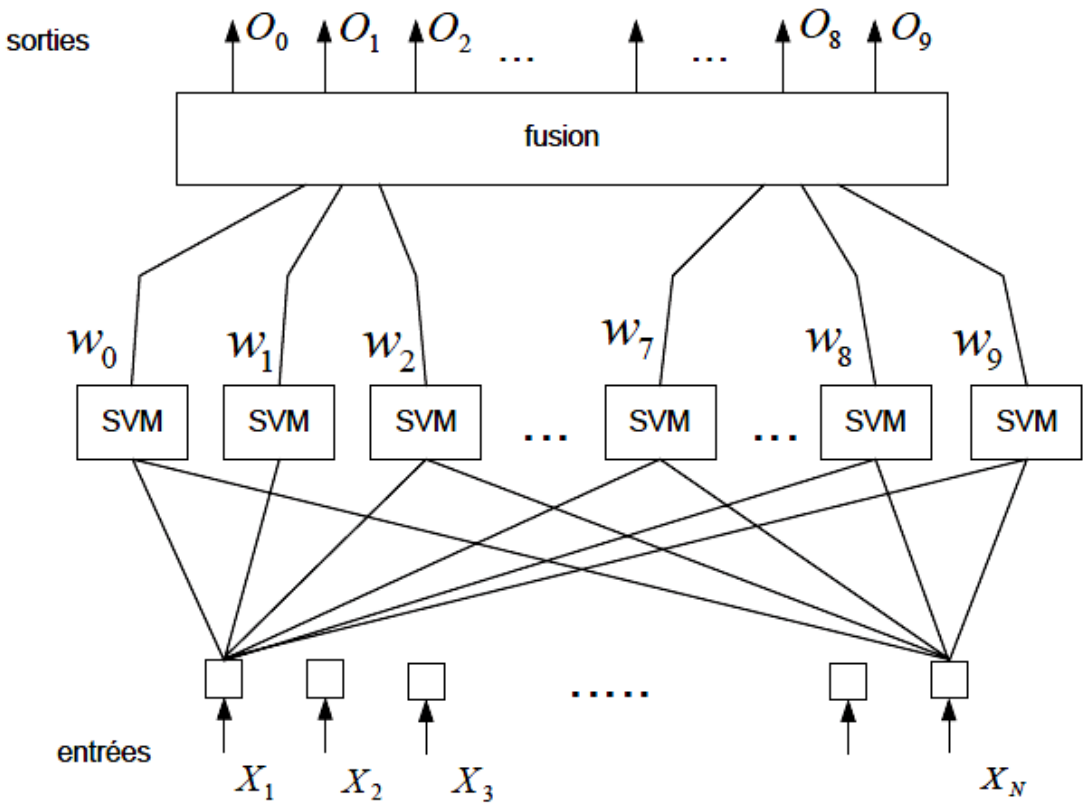


Figure 2.11 Architecture du système en stratégie Un-contre-Tous

La Figure 2.11 montre l'architecture du système en stratégie Un-Contre-Tous. Notons que l'étage étiqueté « fusion » désigne le schéma de vote utilisé et aussi toute sorte d'expert capable de fusionner les sorties pour décider de la classe d'appartenance.

2.7.2 Approche Une-contre-Une

Une-contre-Une. Cette méthode requiert l'apprentissage de $\frac{1}{2}m(m-1)$ classifieurs pour tous les couples de classes possibles. Durant le test, la méthode requiert la combinaison de toutes les sorties de classifieurs pour qu'une décision soit émise.

Ce schéma de décomposition a été adopté par S. Knerr et al. [58] et utilisé pour la première fois dans le contexte des machines à vecteurs supports par Kreßel [59]. Il consiste à construire $\frac{m(m-1)}{2}$ prédicateurs binaires chacun séparant uniquement deux classes tout en ignorant les autres.

L'hyperplan séparateur des classes k et s est la solution du problème d'optimisation suivant :

$$Minimiser_{w^{ks},\xi^{ks},b} \qquad \frac{\|w^{ks}\|^2}{2} + C\sum_{i=1}^{l_{ks}} \xi_i^{ks},$$

$$\begin{aligned} sous \qquad & \langle w^{ks}.\Phi(x_i)\rangle + b^{ks} \geq 1 - \xi_i^{ks}, \qquad si\ \ y_i = k, \\ & \langle w^{ks}.\Phi(x_i)\rangle + b^{ks} \leq -1 + \xi_i^{ks}, \qquad si\ \ y_i = s, \end{aligned} \tag{2.20}$$

$$\xi_i^{ks} \geq 0, \qquad \forall\, i = 1,2,\dots,l_{ks}.$$

Où l_{ks} est le nombre des observations issues des classes k et s.

Pratiquement, nous résolvons le problème dual correspondant au problème (2.20) ayant l_{ks} variables duales. Si chaque classe contient en moyenne $\frac{l}{m}$ exemples, nous aurons à résoudre dans la phase d'apprentissage $\frac{m(m-1)}{2}$ problèmes quadratiques chacun dépendant de $\frac{2l}{m}$ variables.

L'approche une-contre-une consiste donc à construire un classifieur pour chaque paire de classes (k, s) définissant ainsi des fonctions de décision binaire $h_{ks} : X \subseteq \mathbb{R}^p \rightarrow \{-1, +1\}$,

$$h_{ks}(x) = sign\big(f_{ks}(x)\big) = \begin{cases} +1 & si\ x \in \text{à } la\ classe\ k \\ -1 & si\ x \in \text{à } la\ classe\ s \end{cases} \tag{2.21}$$

Pour des raisons de symétrie $h_{ks} \equiv -h_{sk}$ et on convient que $h_{kk} \equiv 0$ pour tout $k \in \{1, 2, \dots, m\}$. Sur la base des $\frac{m(m-1)}{2}$ fonctions de décision binaires h_{ks}, nous définissons m autres fonctions de décision de la façon suivante :

$$h_k(x) = \sum_{s=1}^{m} h_{ks}\,(x)\ , \qquad k = 1,2,\dots,m \tag{2.22}$$

Et la règle de classification d'une nouvelle observation x est donnée par :

$$k^* = \underset{1\leq k\leq m}{\text{Arg max}}\ h_k(x) \tag{2.23}$$

Cette règle proposée par Friedman [60] est connue sous le nom de vote majoritaire, et elle a été appliquée pour la première fois avec les SVM par Kreßel [59].

Une nouvelle observation x est dite non-classifiable si elle appartient à la région d'ambiguïté. Cette région est présentée par la figure 2.12. Toute observation située dans la région d'ambiguïté est classée arbitrairement dans l'une des classes vérifiant la règle (2.23).

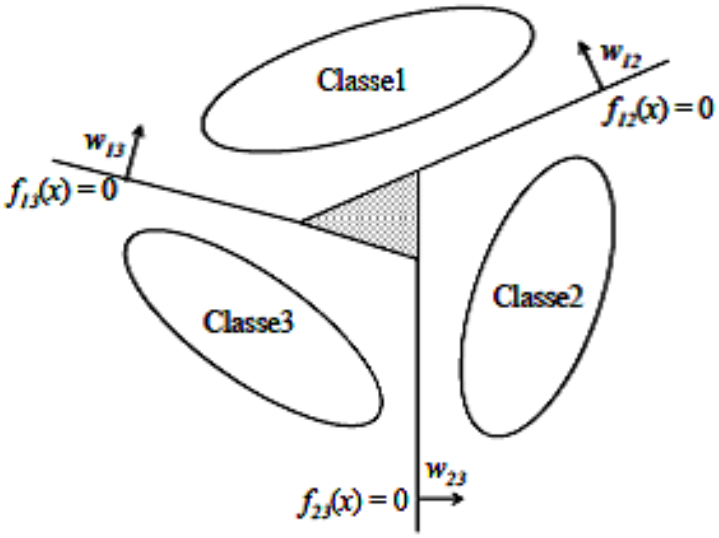


Figure 2.12 Réduction de région d'ambiguïté hachurée pour l'approche une-contre-une.

Les avantages majeurs de cette combinaison sont : la conservation de bonnes parties des $\frac{m(m-1)}{2}$ hyperplans préalablement construits et la diminution de la région d'ambiguïté relativement à l'approche une-contre-reste. En revanche, son erreur de généralisation n'a pas encore de majorant.

Plusieurs méthodes ont été proposées pour combiner les différents classifieurs issus de toutes les paires de classes. Chaque architecture vise à réduire le temps d'apprentissage et le temps de classification d'une nouvelle observation tout en améliorant les capacités de généralisation de la machine. Dans ce qui suit, nous présentons, dans l'ordre chronologique de leur apparition, les différentes combinaisons proposées. Ces combinaisons diffèrent au niveau de la prise de décision au niveau de la région d'ambiguïté.

La Figure 2.13 présente l'architecture simplifiée du système en stratégie Un-contre- Un.

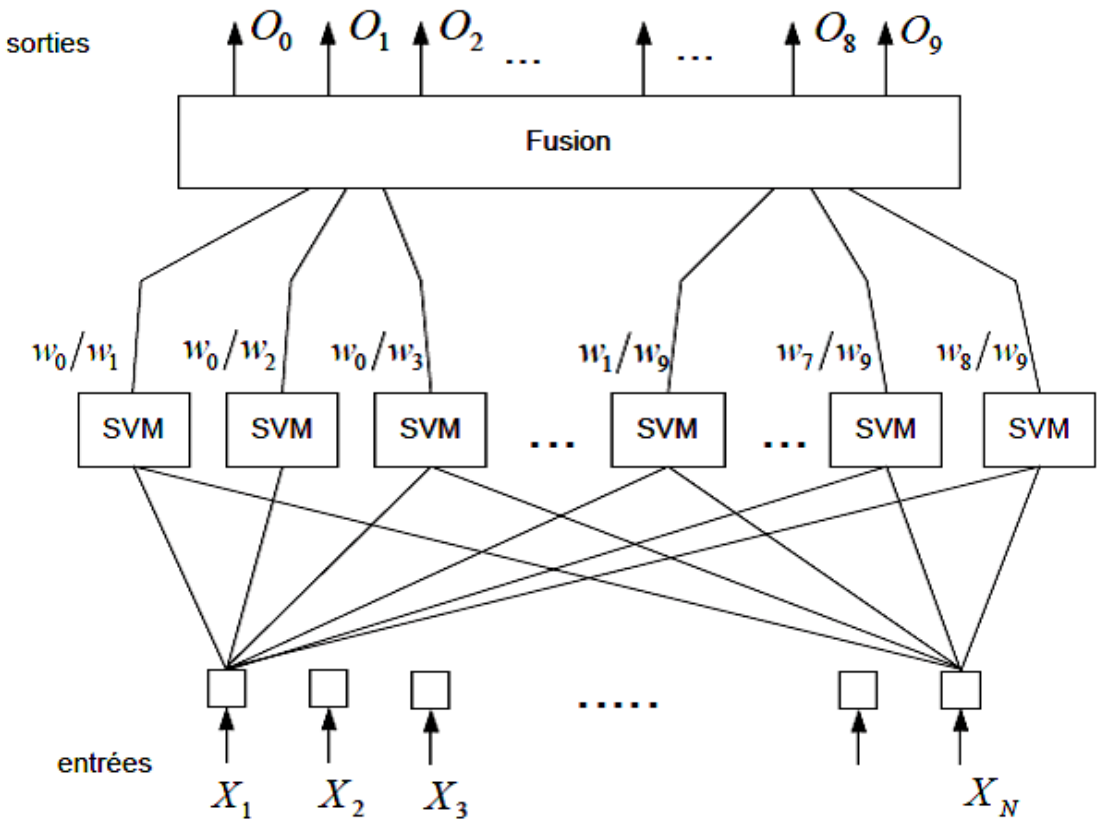


Figure 2.13 Architecture du système en stratégie Une-contre-Une

2.7.3 SVM floue

D'une façon similaire aux FSVM proposées pour le schéma de décomposition en une-contre-reste, S. Abe [61] a introduit pour la décomposition une-contre-une les fonctions d'adhésion[3] définies pour chaque hyperplan séparateur $H_{ks} = \{x \in X ; f_{ks}(x) = 0\}$, $(k \neq s)$, de la façon suivante :

$$M_{ks}(x) = \begin{cases} 1 & pour\ f_{ks}(x) \geq 1 \\ f_{ks}(x) & sinon \end{cases}$$

Les hyperplans H_{ks} sont obtenus suite à la résolution des problèmes (2.20) pour toute paire de classes k et s. Il est à noter que, selon la relation (2.21), l'observation x est bien classée par H_{ks} si est seulement si $f_{ks}(x) > 0$. Dans le cas où l'observation x viole la marge on a $M_{ks}(x) = f_{ks}(x)$. En d'autres termes, la valeur de la fonction d'adhésion M_{ks} mesure la difficulté que posent les observations critiques pour être classées par l'hyperplan H_{ks} ; plus $M_{ks}(x)$ est petite plus la classification de x est difficile.

[3] Membership functions

En utilisant les fonctions $M_{ks}(x), (s \neq k,\ s = 1, 2, \ldots, m)$, on définit la fonction d'adhésion à la classe k comme étant :

$$M_k(x) = \min_{\substack{s=1,2,\ldots,m \\ s \neq k}} M_{ks}(x)$$

Cette dernière équation est équivalente à :

$$M_k(x) = \min\left(1, \min_{\substack{s=1,2,\ldots,m \\ s \neq k}} f_{ks}(x)\right) \tag{2.24}$$

La fonction d'adhésion à la classe k définit dans l'espace X des formes polyédriques tronquées. Une représentation de ces formes dans le plan est donnée par la Figure 2.14 : plus la valeur de la fonction d'adhésion à la classe k est élevée plus l'observation x est proche de la classe k.

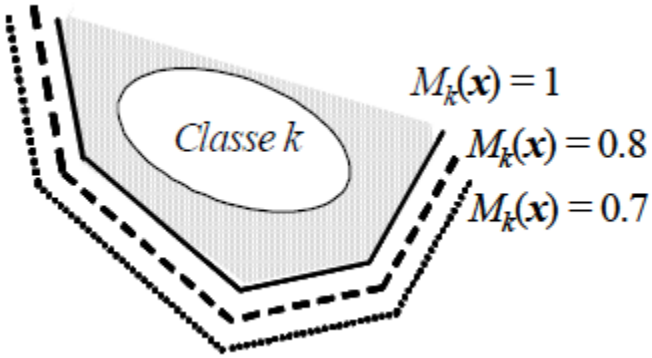


Figure 2.14 Chaque frontière définit une courbe de niveau pour la fonction $M_k(x)$ d'adhésion à la classe k. Cette fonction vaut 1 sur tout point de la zone hachurée.

On vérifie aisément que l'égalité ne peut être vérifiée que pour une seule classe. En effet d'après (2.24) $M_k(x) = 1$ si et seulement si $f_{ks}(x) \geq 1\ \forall s$ d'où l'observation x est classée dans k par toutes les fonctions binaires. Par conséquent la relation (2.24) peut être réduite à :

$$M_k(x) = \min_{\substack{s=1,2,\ldots,m \\ s \neq k}} f_{ks}(x)$$

Une nouvelle observation x sera donc classée suivant la règle :

$$\operatorname{Arg}\max_{k=1,2,\ldots,m} M_k(x)$$

Cette règle donne la même décision que celle donnée par (2.23) pour les observations qui n'appartiennent pas à la région d'ambiguïté montrée dans la Figure 2.12. Les FSVM partagent la région d'ambiguïté équitablement sur les classes selon leur proximité. Ce partage est illustré dans la Figure 2.15.

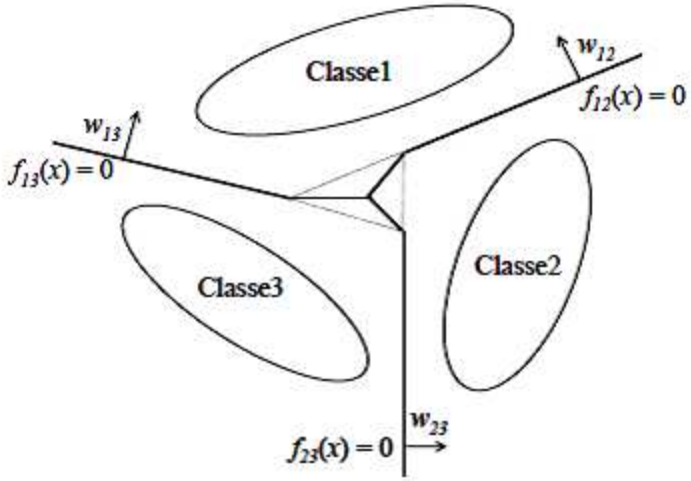


Figure 2.15 Résolution de la région d'ambiguïté par les FSVMs.

2.7.4 Graphe de décision acyclique orienté (DDAG[4])

Platt et al. [62] ont proposé une structure d'arbre de décision pour combiner les $\frac{m(m-1)}{2}$ classifieurs binaires construites selon la décomposition une-contre-une. La phase d'apprentissage du DDAG est exactement la même que celle pour le vote majoritaire. Elle consiste à construire touts les $\frac{m(m-1)}{2}$ classifieurs binaires. Par contre, l'étape test utilise un graphe binaire, enraciné, orienté et acyclique ayant $\frac{m(m-1)}{2}$ nœuds intérieurs répartis sur $(m-1)$ couches et m feuilles formant la dernière couche. Chaque nœud correspond à un classifieur binaire de la $k^{ème}$ et la $s^{ème}$ classes et chaque feuille désigne une classe.

Une nouvelle observation x, partant du noeud racine, circule d'un nœud à un autre jusqu'à atteindre une feuille qui indiquera sa classe d'appartenance. Au niveau de chaque nœud, l'observation x se retrouve devant un choix binaire : passer à gauche ou à droite. Ce choix dépend de la décision de classification binaire prise au niveau de ce nœud. Une illustration graphique du DDAG pour $m = 3$ est donnée par la Figure 2.16.

[4] Decision Directed Acyclic Graph.

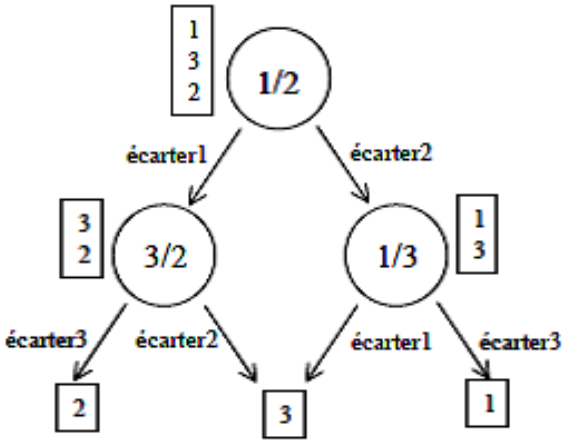


Figure 2.16 Graphe de décision acyclique orienté à trois classes.

Cette architecture peut être vue sous forme d'une liste à m classes, dans laquelle chaque nœud élimine une classe. Cette liste est initialisée avec toutes les classes. Une nouvelle observation x sera évaluée par le nœud de décision binaire correspondant au premier et au dernier élément de la liste. Lorsque ce nœud préfère l'une des deux classes confrontées, l'autre sera éliminée de la liste et l'algorithme se poursuit pour la nouvelle liste. Cet algorithme s'arrête quand la liste est réduite à une seule classe, celle-ci sera attribuée à x. Ainsi, pour un problème à m classes, $(m - 1)$ nœuds de décision binaire sont évalués dans le but de classer toute nouvelle observation.

L'avantage du DDAG par rapport aux autres approches multi-classes est que, grâce à sa structure particulière, son erreur de généralisation est bornée. En outre, son temps de classification est réduit comparativement au vote majoritaire et aux FSVM. En revanche les capacités de généralisation du DDAG dépendent de l'ordre de la liste initiale sur laquelle il agit.

Pour un même problème à m classes, il y a $\frac{m!}{2}$ structures différentes du DDAG. L'ordre de la liste initiale du haut vers le bas est le même que celui qu'on retrouve sur les feuilles de droite à gauche. Ainsi, pour chaque DDAG la région d'ambiguïté est partagée sur les feuilles internes. Une illustration graphique du cas $m = 3$ est donnée par la Figure 2.17.

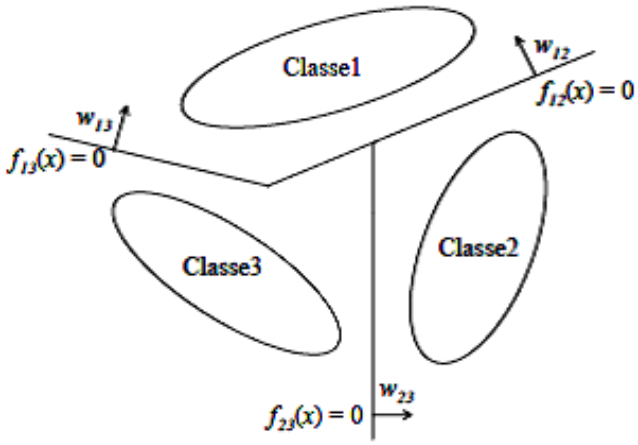


Figure 2.17 Favorisation de la feuille du milieu pour DDAG pour l'affectant la région d'ambiguïté.

2.7.5 Graphe acyclique orienté adaptatif (ADAG[5])

Pontil et al. [63] ont proposé une combinaison du type tournoi de tennis entre les classes afin d'optimiser les confrontations bi-classes. Sans s'en rendre compte, Kijsirikul et al. [64] ont employé la même méthode, qu'ils ont nommée graphe orienté acyclique adaptatif, dans le but d'améliorer les performances du DDAG.

Un ADAG est une structure triangulaire renversée du DDAG. Pour un problème à m classes, sa phase d'apprentissage est identique à celle du DDAG. Par contre, pour la phase de classification, les $\frac{m(m-1)}{2}$ nœuds sont arrangés sous forme d'un triangle renversé. Sa première couche contient $\frac{m}{2}$ nœuds, sa deuxième couche sera réduite à $\frac{m}{2^2}$ nœuds et ainsi de suite jusqu'à la dernière couche formée d'un seul nœud. Un ADAG contient exactement $(m - 1)$ nœuds de décision binaire répartis sur $log_2(m)$ couches. La structure d'un ADAG pour $m = 8$ est représentée dans la Figure 2.18.

Une nouvelle observation x sera testée au niveau de chacun des $\frac{\mathrm{m}}{2}$ nœuds de la première couche. Au cours de cette première manche, chaque décision nœudale éliminera une classe. Par conséquent, le nombre des classes candidates sera réduit de moitié. Ce processus éliminatoire continue jusqu'à ce que ce tournoi atteigne sa finale. Ainsi, la dernière couche d'ADAG est réduite à un nœud unique de décision finale pour l'observation x.

[5] Adaptive Directed Acyclic Graph.

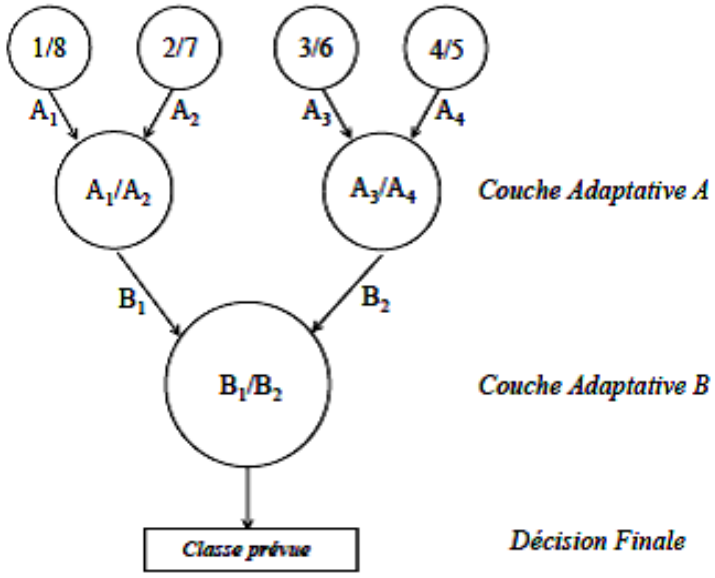


Figure 2.18 ADAG à huit classes.

Notons que pour un ADAG, le nombre maximal de confrontations que peut avoir la classe correcte avec les autres classes est $log_2(m)$. Ce nombre est considérablement inférieur à celui lié à la structure DDAG qui croît linéairement en fonction de m. Par conséquent, l'architecture ADAG réduit bien l'erreur cumulée commise par DDAG.

Un ADAG peut être mis en œuvre en utilisant une liste, de laquelle chaque nœud élimine une classe. La liste initiale est constituée de toutes les m classes du problème. Une nouvelle observation x est évaluée au niveau du nœud de séparation binaire qui confronte la première et la dernière classe de la liste. La classe préférée par ce nœud est gardée dans la position extrême gauche de la liste de la seconde manche, tandis que l'autre classe est rejetée. Ensuite, cette observation x est testée au niveau du nœud qui correspond à la deuxième et l'avant dernière classes de la liste initiale. Le processus d'évaluation pour la première manche se termine quand au plus une classe reste non confrontée aux autres classes de la première liste. Dans le cas où une seule classe reste non confrontée aux autres, elle occupera la position extrême droite de la liste de la seconde manche. À la fin de cette première manche, la liste initiale à m classes est réduite en une liste à $\frac{m}{2}$ éléments si m est pair et en une liste à $\frac{m+1}{2}$ éléments si m est impair. Le processus de manches continue jusqu'à aboutir à une liste réduite à une seule classe à laquelle l'observation x est affectée. Une illustration graphique est donnée par la Figure 2.19 pour $m = 7$.

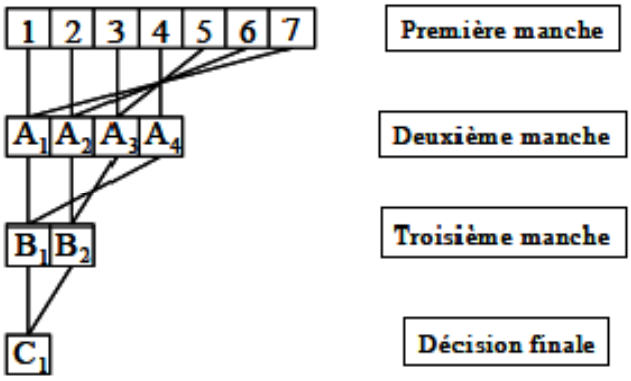


Figure 2.19 Illustration par listes d'un ADAG à sept classes.

Les structures DDAG et ADAG sont fortement liées. ADAG est une structure renversée du DDAG. Par exemple, pour $m = 3$, à chaque ADAG est associé un DDAG équivalent. Ce résultat n'est pas généralisable pour un nombre de classes m quelconque.

Il est clair que l'ADAG est aussi instable vu qu'il dépend de l'ordre de la liste de chaque manche. En effet, pour un problème à m classes il y a $\frac{m!}{\left[\frac{m}{2}\right]2^{\left\lceil\frac{m}{2}\right\rceil}}$ cas de figures possibles pour la première couche de nœuds selon l'ordre de la liste initiale [55].

2.7.6 Graphe acyclique orienté adaptatif réordonné (RADAG[6])

Dans le but de trouver une architecture optimale de l'ADAG, Phetkaew et al. [65] [66] ont proposé une version réordonnée de l'ADAG qu'ils ont nommé RADAG.

Cette approche consiste à optimiser l'architecture d'un ADAG en introduisant une étape de mise en ordre de la liste qui précède la formation de chaque couche de nœuds. Cette étape est accomplie par un algorithme d'optimisation qui groupe tous les éléments de la liste par paires de classes en minimisant la somme des erreurs de généralisation de toutes les paires. Les paires trouvées forment la couche des nœuds associés à la manche courante. La Figure 2.20 résume les différentes étapes du RADAG.

[6] Reordering Adaptive Directed Acyclic Graph.

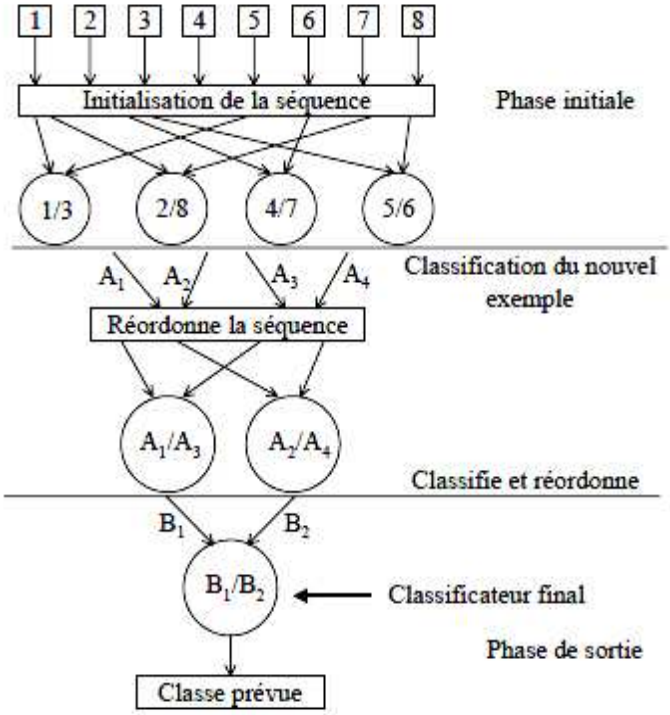


Figure 2.20 Différentes étapes du RADAG.

2.8 Conclusion

Nous avons commencé ce chapitre par la présentation du principe de fonctionnement générale des machines à vecteurs de support et leur fondement mathématique. On a exposé les deux cas des données linéairement séparable et non linéairement séparables ce dernier cas nécessite l'utilisation d'une marge souple et une fonction noyau pour trouver un hyperplan séparateur optimal.

Une synthèse des différentes approches des SVM multi-classes a été présentée à travers une analyse minutieuse de ces approches, nous avons pu évoquer leurs atouts et leurs défauts. Des similitudes entre ces approches ont été aussi établies.

Nous avons vu que l'approche une-contre-une est plus adaptée que l'approche une-contre-reste pour les machines à vecteurs supports multi-classes. Plusieurs méthodes ont été introduites pour les deux approches dont chacune présente des atouts et des défauts, la prise de décision au niveau de la région d'ambiguïté est à l'origine des différences entre ces approches.

Chapitre III

Sélection de modèles de machines à vecteurs de support

3.1 Introduction

Parmi les difficultés majeures liées à l'utilisation de classifieurs figure la nécessité d'adapter les variables conditionnant de processus d'apprentissage. Ces variables sont appelées hyper-paramètres. La résolution de la méthode des machines à vecteurs support implique la sélection de plusieurs paramètres : le type de noyau, le ou les paramètres du noyau (σ,d...) et le paramètre de régularisation C, en plus du type d'apprentissage multi-classe (une-contre-une et une-contre-reste).

Plusieurs méthodes de sélection de modèles sont utilisables pour le choix des valeurs des hyper-paramètres. Si le modèle utilise un seul hyper-paramètre, il est possible d'essayer un nombre fini de valeurs et choisir celle qui maximise le taux de reconnaissance. Cette technique devient toutefois difficile à implémenter pour deux hyper-paramètres ou plus. Afin de choisir les valeurs des hyper-paramètres, il est nécessaire de spécifier une méthode permettant de sélectionner un modèle parmi plusieurs possibles.

3.2 Mesure d'erreur en apprentissage

En utilisant des algorithmes d'apprentissage nous devons considérer deux critères : l'erreur d'apprentissage E_a et l'erreur de généralisation E_g [48]. L'erreur d'apprentissage correspond au taux de mauvais classement sur l'ensemble d'apprentissage. L'erreur de généralisation correspond au taux de mauvais classement sur l'ensemble de test. Pour comparer les modèles d'apprentissage il est intéressant de calculer une table de contingence (table 3.1).

		classe réelle	
		classe +	classe -
classe prédite	classe +	A vrais positifs	B faux positifs
	classe -	C faux négatifs	D vrais négatifs

Table 3.1 Table de contingence

A partir de celle-ci on peut alors obtenir différents critères tels que le taux de reconnaissance donné par la formule 3.1 :

$$\boldsymbol{T_{rec} = \frac{A + D}{A + B + C + D}} \tag{3.1}$$

Ou encore la précision et le rappel du modèle donné respectivement par 3.2 et 3.3.

$$\boldsymbol{T_{prec} = \frac{A}{A + B}} \tag{3.2}$$

$$\boldsymbol{T_{rappel} = \frac{A}{A + C}} \tag{3.3}$$

3.3 Compromis biais-variance

Un choix non judicieux des hyper-paramètres du modèle SVM peut accentuer l'erreur. Une capacité exagérée cause un sur-apprentissage des données et la solution devient sensible au bruit. On parle alors d'erreur de variance. Une capacité non suffisante cause un sous-apprentissage des données. La solution, dans ce cas, approxime mal les exemples d'apprentissage. On parle alors d'erreur de biais. Une erreur de biais importante correspond à une erreur de variance réduite et vice versa. C'est le dilemme «Biais-Variance» bien connu en théorie d'apprentissage.

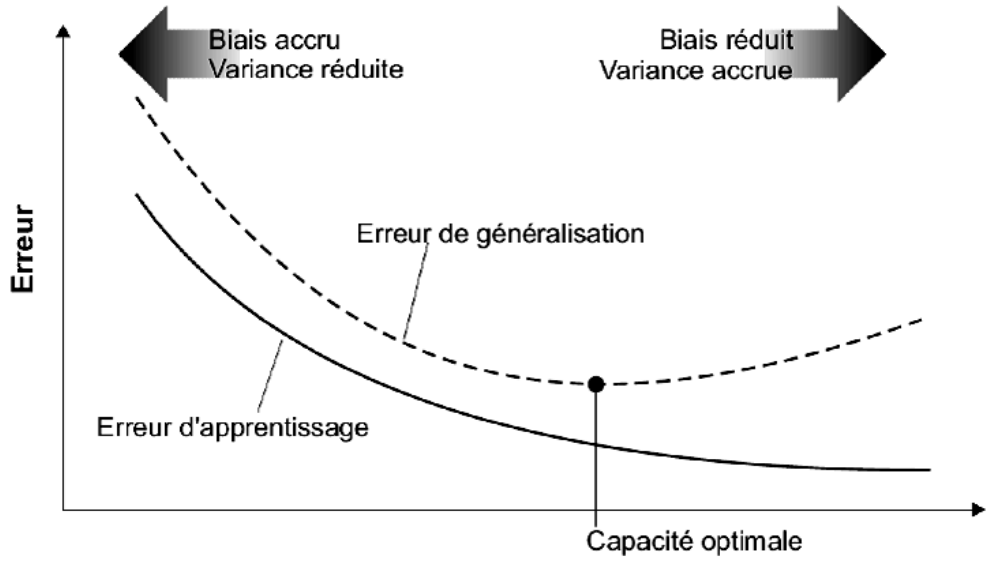


Figure 3.1 Compromis biais-variance

Dans les grandes lignes, le compromis biais-variance peut être formulé de la façon suivante : L'erreur de généralisation peut être décomposée en deux termes : un de biais et un de variance (Figure 3.1).

3.4 Les techniques de sélection

3.4.1 Validation croisée

La validation croisée est une technique qui permet de tester un modèle d'apprentissage. La validation croisée se décline en plusieurs sous-méthodes. La plus répandue est la méthode « k-Fold » avec $k \in [4,10]$. Si l'on a une base d'apprentissage Ap contenant p éléments : $Ap = \{X_1, \dots, X_p\}$ la validation croisée consiste à appliquer les cinq étapes suivantes :

1. Découper l'ensemble des exemples en k sous-ensembles disjoints de taille $\frac{p}{k}$.
$$A_p = B_1, B_2, \dots, B_k$$
2. Apprendre sur les $(k-1)$ sous-ensembles.
3. Calculer l'erreur sur la k^{ime} partie.
4. Réitérer le processus k fois.
5. Obtenir l'erreur finale en calculant la moyenne des k erreurs précédentes.

La validation croisée est simple à mettre en œuvre et utilise toutes les données. Elle permet d'obtenir une estimation de l'erreur de généralisation. Cela permet d'éviter le sur-apprentissage [67].

3.4.2 Leave-one-out

La procédure de leave-one-out est un cas particulier de la procédure de validation croisée avec $k = m$. Il y a donc m phases d'entraînement : une pour chaque exemple de la base A_p. Chaque base d'entraînement B_i^E correspond à la base d'exemples initiale à laquelle un exemple a été enlevé $B_i^E = A_p - \{\mathrm{B}_i\}$. La base de test B_i^T ne contient qu'un exemple $B_i^T = \{\mathrm{B}_i\}$, celui enlevé à A_p. L'erreur finale est obtenue en calculant la moyenne des k erreurs de chaque partie. Cette estimation de l'erreur réelle est considérée comme l'estimateur le moins biaisé. L'avantage de cette méthode est que chaque phase d'entraînement a un maximum d'exemples, mais le nombre important de phases d'apprentissage devient prohibitif dès que le nombre d'exemples est de l'ordre de quelques centaines.

3.4.3 Grille de recherche

C'est une méthode classique qui discrétise l'espace des modèles [68]. Leur limite est la sélection de modèles qui exploitent peu de paramètres (≤ 2). Malgré ce défaut, elle permet de tracer des surfaces correspondant à l'évolution des capacités de généralisation d'un SVM en fonction des valeurs de ces deux hyper-paramètres. Cela permet de mieux appréhender ces évolutions, en plus d'une classique sélection de modèle.

Un grand nombre d'expérimentations dans [69] montrent que le paysage de l'erreur en généralisation, à travers l'utilisation de cette "grille", comporte des minima locaux, ce qui illustre que la seule sélection des hyper-paramètres des SVM est un problème difficile en soi.

Ces surfaces montrent également que l'évolution de l'erreur en généralisation a des variations peu prononcées lorsque les variations des hyper-paramètres sont faibles. L'optimisation des capacités en généralisation ne nécessite donc pas la recherche de valeurs qui soient très précises pour ces hyper-paramètres, mais d'un ordre de grandeur qui ne soit pas trop grossier. Cela permet de réaliser une recherche efficace avec des pas de discrétisation relativement grands.

Par exemple pour un noyau gaussien, nous cherchons le meilleur couple (C, σ). Nous allons donc utiliser des séquences exponentielles [67] : $C = 2^{-5}, 2^{-4}, \dots, 2^{15}$ et $\sigma = 2^{-15}, 2^{-14}, \dots, 2^{3}$. En déterminant une heuristique sur la qualité des résultats on pourrait éviter une recherche exhaustive de toutes les combinaisons de paramètres. Une recherche exhaustive est quand même envisageable car les paramètres sont indépendants et nous pouvons donc facilement paralléliser la recherche.

3.5 Optimisation avec méta-heuristiques

Le choix des valeurs optimales de σ et C (pour un noyau gaussien par exemple) correspond à la recherche d'un modèle θ optimal (θ ≡ (C, σ)). Le problème de la recherche du modèle θ correspond à un problème qui n'est plus convexe, il y a donc en général plusieurs minimas locaux. Parmi les méthodes proposées pour résoudre ce type de problème, un sous-ensemble d'entre elles utilise des méthodes désignées sous le terme de méta-heuristiques.

Les méta-heuristiques sont appliquées à de nombreux domaines dans l'apprentissage artificiel. La problématique qui correspond à la sélection d'un sous-ensemble réduit

d'attributs pertinents ou d'exemples pertinents est une des applications majeures des méta-heurisques dans le cadre de l'apprentissage artificiel [70].

L'avantage des méthodes méta-heuristiques est qu'elles sont suffisamment génériques pour permettre d'optimiser une large gamme de problèmes différents, sans faire de changements profonds dans le principe algorithmique employé. Pour un problème particulier, l'adéquation d'un algorithme de type méta-heuristique consiste à définir des briques élémentaires adaptées à ce problème. Ces briques élémentaires sont alors directement utilisées par l'ossature générale de la méthode méta-heuristique. Un des avantages de ces méthodes est que lorsque deux problèmes sont de nature similaire, l'effort d'adaptation est réduit pour passer de l'optimisation de l'un à l'autre, car certaines briques de base peuvent être réutilisées ou nécessiter des modifications mineures [71].

3.5.1 Algorithmes génétiques

Les algorithmes génétiques ont été conçus par J. Holland [72]. Ce sont des algorithmes d'optimisation. On trouvera une présentation détaillée de ces processus dans de nombreux ouvrages [73]. Les Algorithmes génétiques (AG) reposent sur plusieurs hypothèses :

- un codage homogène des informations,
- une utilisation d'une population de modèles et non un seul modèle,
- une mesure d'adaptation des modèles à l'environnement,
- une reproduction des modèles avec modification et échange d'informations non déterministes.

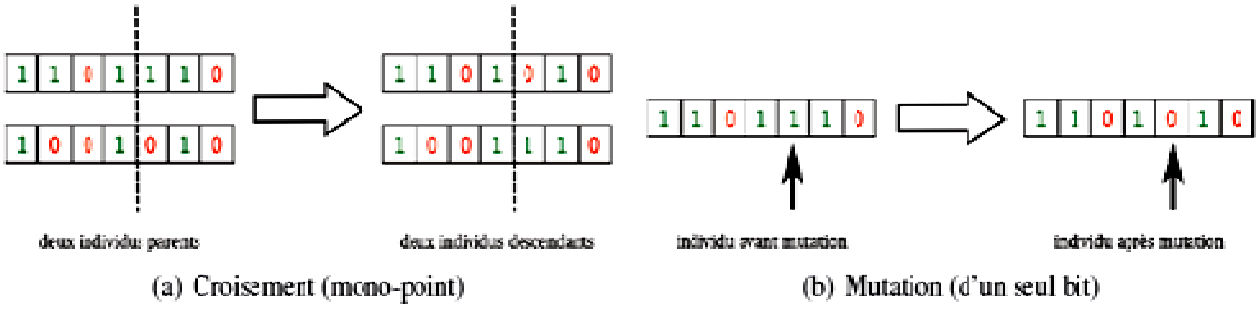
La recherche de la meilleure solution se fait donc par l'évolution d'une population d'individu qui sont sous mis à deux mécanismes tirés de la génétique : la mutation et le croisement. D'une façon générale, les étapes d'un algorithme génétique sont décrites dans l'algorithme ci-après. Des variantes existent mais les points communs sont les hypothèses précédemment listées.

Plusieurs remarques à cette étape s'imposent :

– le codage des individus ou chromosomes se fait sous la forme d'une chaîne de bits,

– le croisement produit deux descendants à l'aide de deux parents à partir d'un point de coupure choisit aléatoirement dans le chromosome des parents,

– la mutation produit un nouveau individu à partir d'un seul individu ; on opère un complément à 1 d'un certain bit.



(a) Croisement (mono-point) (b) Mutation (d'un seul bit)

Figure 3.2 Principe des opérateurs génétiques dans le cas d'individu codé sur 7 bits.

Du point de vue algorithmique, les AG sont intéressants comparativement à d'autres algorithmes tel que le recuit-simulé car ils traitent et font évoluer plusieurs solutions et non une seule solution. De plus les mécanismes d'évolution 3.1 font que les individus de la population courante interragissent entre eux au lieu d'évoluer chacun indépendamment.

Algorithme :

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

3.5.2 Recherche avec tabous

La recherche avec tabous est une méthode méta-heuristique pour l'optimisation difficile qui a été développée dans les années 80 par Fred Glover [74][75]. Elle s'est révélée particulièrement efficace et a été appliquée avec succès à de nombreux problèmes. Elle est basée sur une recherche itérative qui choisit dans un voisinage restreint la meilleure solution même si elle est plus mauvaise que celle de l'itération précédente. Une mémoire à court terme est utilisée pour éviter tout cycle visitant périodiquement le même optimum local. A partir des

derniers mouvements mémorisés, un ensemble de solutions est considéré comme tabou et un mouvement n'est réalisable que vers l'une des solutions voisines qui n'est pas taboue.

Le choix des solutions taboues peut être tout simplement les dernières solutions visitées, mais en général le critère qui détermine les solutions taboues à un moment précis de la recherche dépend du problème à optimiser et de la représentation d'une solution.

Lorsque des caractéristiques de modification définissent l'appartenance d'une solution à l'ensemble tabou, l'ensemble des solutions interdites à chaque itération peut contenir des solutions meilleures que toutes celles déjà visitées. Un mécanisme particulier, appelé l'aspiration, permet de pallier l'effet de ne pas pouvoir les sélectionner et de lever ainsi leur statut tabou.

Des mécanismes nommés diversification et intensification permettent de doter cette méthode de comportements correspondant à des effets caractéristiques d'une mémoire à long terme, la recherche taboue oscillant entre ces deux états pendant son processus de recherche.

L'objectif est de permettre à la fois d'atteindre des régions prometteuses par un parcours rapide de l'ensemble de l'espace de recherche et de pouvoir localiser le plus précisément possible une solution proche de l'optimal par un parcours agressif de la région prometteuse la contenant. L'ensemble de ces propriétés fait que la méthode de recherche avec tabous peut être adaptée à une grande classe de problèmes d'optimisation difficiles.

3.5.2.1 Liste Tabou

La liste Tabou représente la mémoire à court terme, elle contient les attributs des mouvements les plus réalisés. Cette liste est maintenue dans le but d'orienter la recherche.

3.5.2.2 Critère d'aspiration

Ce critère est souvent utilisé pour enlever les restrictions Tabou d'un mouvement de haute qualité.

Il permet de passer outre certains cas interdits. Son utilisation principale consiste à outrepasser l'interdiction d'un mouvement s'il permet d'obtenir un élément meilleur que la solution trouvée jusqu'à présent (consiste à révoquer le statut Tabou d'un mouvement si ce dernier permet d'atteindre une solution de qualité supérieure à celle de la meilleure solution trouvée jusqu'alors).

3.5.2.3 Stratégie d'intensification

L'intensification consiste à retourner à l'une des meilleures solutions trouvées jusqu'à présent, puis de reprendre la recherche à partir de cette solution. C'est une exploration plus poussée d'une région prometteuse.

La stratégie d'intensification est matérialisée dans l'algorithme suivant par renforcement de la recherche dans la liste des meilleurs mouvements.

3.5.2.4 Stratégie de diversification

Elle consiste à générer une nouvelle solution, différente de celles déjà explorées, dans le but de partir dans une nouvelle direction, pour explorer une autre région.

3.5.2.5 Algorithme général de la recherche Tabou

1. Obtenir une solution initiale.
2. Créer une liste des mouvements candidats.
3. Choisir le meilleur candidat. Ce choix est basé sur les restrictions Tabou et le critère d'aspiration.
   - On obtient ainsi une autre solution, mais qui ne sera enregistrer que si elle est meilleur que touts les solutions déjà visitées.
4. Appliquer le critère d'arrêt.
   - Continue : changer les conditions d'admissibilité (restriction Tabou et critère d'aspiration). Aller à 2.
   - Arrêt : passer aux stratégies d'intensification et de diversification.

3.6 Adaptation de la recherche taboue à notre problème

Les quelques expérimentations réalisées dans [76] montrent que la sélection de modèle avec la recherche tabou est aussi efficace et plus rapide que la procédure grille de recherche. Un des avantages de cette méthode est qu'elle est facile à étendre à des noyaux comportant plusieurs paramètres libres.

Dans notre système de reconnaissance nous avons utilisé le principe de la recherche tabou pour sélectionner les paramètres des noyaux, par exemple pour un noyau RBF les mouvements possibles correspondent à ajouter ou soustraire ΔC et $\Delta\sigma$ respectivement à la constante de régularisation et à la largeur du noyau. Donc les valeurs des paramètres C ou σ

sont diminuées ou augmentées suivant que le mouvement choisi a permis d'augmenter ou de diminuer les capacités en généralisation. La procédure s'arrête lorsque le nombre d'itération maximal est atteint.

3.6.1 Procédure du travail pour le noyau RBF

1- Choisir les paramètres initiale (C et σ).
2- Ajouter les paramètres à la liste Tabou.
3- Entrainer et classifier, T* = T (Taux de reconnaissance).
4- Enregistrer les paramètres et le taux correspondent dans un fichier F.
5- Choisir le meilleur mouvement M qui n'existe pas dans la liste Tabou parmi :
   - M1 : $C + \Delta C$ et σ
   - M2 : $C - \Delta C$ et σ
   - M3 : C et $\sigma + \Delta\sigma$
   - M4 : C et $\sigma - \Delta\sigma$
6- Ajouter les paramètres à la liste Tabou.
7- Entrainer et classifier
   Si T > T* alors
   - T* = T
   - Ajouter les paramètres et T* au fichier F.
8- Si le nombre maximal d'itérations n'est pas atteint aller à 5.

Nous devons mentionner ici que la liste Tabou est gérée comme une liste FIFO, le plus ancien mouvement est retiré de la liste et devient non tabou.

Pour la diversification de la recherche nous augmentons la valeur des pas ($\Delta\sigma, \Delta C$ ...). Et on fait l'inverse pour l'intensification (la diminution des valeurs des pas).

Le nombre de voisinages (NV) d'une solution dépend le nombre de paramètres (NP) a sélectionnés, NV = 2 x NP.

3.7 Conclusion

Nous avons présenté dans ce chapitre les différentes méthodes de sélection de modèle SVM. La sélection des hyper-paramètres des classifeurs y est présentée comme un problème d'optimisation qui doit satisfaire le compromis «biais-variance». A travers quelques références, nous montrons que des méthodes à base de méta-heuristiques ont déjà été appliquées avec succès à des problèmes d'apprentissage, finalement nous avons présenté notre algorithme de sélection basé sur la recherche tabou qui permet de guider la recherche d'une solution optimale.

Chapitre IV

Résultats expérimentaux et Discussions

4.1 Introduction

Nous présentons dans ce chapitre une description détaillée de notre système de reconnaissance de caractères Arabe manuscrits hors-ligne, avec l'évaluation de performance de chaque phase de processus de reconnaissance. Ainsi que nous présentons les différents résultats d'expérimentations par l'utilisation des différents noyaux du SVM et l'implémentation de notre stratégie de recherche pour varier et sélectionner les hyper-paramètres qui donnent le meilleur taux de reconnaissance possible.

4.2 Description de notre base de données

Lors de l'évaluation des performances d'un système de reconnaissance il est nécessaire de disposer d'une base de données. Nous avons construit au niveau de notre laboratoire SIMPA une base de données contenant 4840 images de caractères arabes manuscrits. Les lettres se trouvent dans les différentes positions (isolée, début, milieu et fin), pour les lettres isolées on a 100 x 28 = 2800 images et pour les autres il y a 30 x 68 = 2040 images, donc le nombre de classes est 28+68=96 (Tableau 4.1).

Cette base est divisée en deux parties une pour l'apprentissage contenant 3840 images est l'autre pour le test contenant 1000 images.

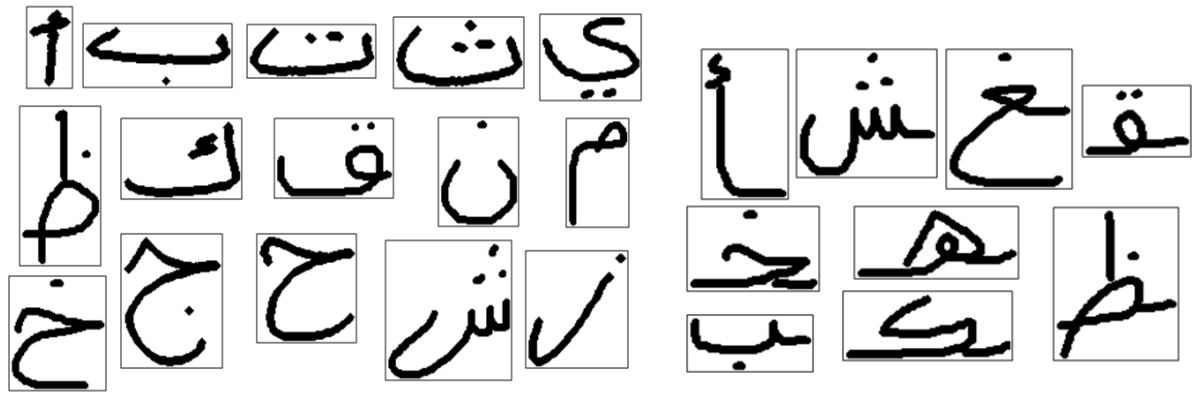


Figure 4.1: Des échantillons de notre base de données

N°	Lettre isolé	Au début	Au milieu	A la fin
1	ا / أ	/	ـا/ـأ	/
2	ب	بـ	ـبـ	ـب
3	ت	تـ	ـتـ	ـت
4	ث	ثـ	ـثـ	ـث
5	ج	جـ	ـجـ	ـج
6	ح	حـ	ـحـ	ـح
7	خ	خـ	ـخـ	ـخ
8	د	/	ـد	/
9	ذ	/	ـذ	/
10	ر	/	ـر	/
11	ز	/	ـز	/
12	س	سـ	ـسـ	ـس
13	ش	شـ	ـشـ	ـش
14	ص	صـ	ـصـ	ـص
15	ض	ضـ	ـضـ	ـض
16	ط	/	ـطـ	/
17	ظ	/	ـظـ	/
18	ع	عـ	ـعـ	ـع
19	غ	غـ	ـغـ	ـغ
20	ف	فـ	ـفـ	ـف
21	ق	قـ	ـقـ	ـق
22	ك	كـ	ـكـ	ـك
23	ل	لـ	ـلـ	ـل
24	م	مـ	ـمـ	ـم
25	ن	نـ	ـنـ	ـن
26	ه	هـ	ـهـ	ـه
27	و	/	ـو	/
28	ي	يـ	ـيـ	ـي

Tableau 4.1 Les lettres arabes utilisées avec ses différentes formes (96 classes).

4.3 Ressources matérielles et logicielles :

4.3.1 Ressources matérielles

Nous avons utilisé dans ce travail un PC portable (CORE 2 Duo CPU 2.10 Ghz, avec 3 Go de RAM), utilisé pour réaliser notre système de reconnaissance et aussi la création de notre base de données (les corpus).

4.3.2 Ressources logicielles

4.3.2.1 Environnement de développement

Nous avons développé notre système à l'aide du langage C++ Builder. Qui est un environnement de programmation visuel orienté objet pour le développement rapide d'applications (RAD). En utilisant C++Builder, vous pouvez, avec un minimum de codage manuel, créer de performantes applications.

C++Builder fournit tous les outils nécessaires pour développer, tester et déployer des applications, notamment une importante bibliothèque de composants réutilisables, une suite d'outils de conception, des modèles d'applications et de fiches, ainsi que des experts de programmation.

4.3.2.2 *SVM*^{multiclass}

Nous avons utilisé l'implémentation *SVM*^{multiclass} qui est un moteur SVM développé par Thorsten Joachims en 2008.

SVM^{multiclass} est une implémentation open-source de référence pour les machines à support vectoriel.
Elle est disponible sur http://www.cs.cornell.edu/People/tj/svm%5Flight/svm_multiclass.html avec documentation, exemples, et références bibliographiques.

Le code est implémenté en C et devenu un des moteurs les plus utilisés actuellement. Il possède des nombreuses fonctionnalités et des caractéristiques très attractives pour l'utilisateur :

- Il permet d'utiliser des nombreux kernels prédéfinis, et même d'utiliser des kernels définis par l'utilisateur.
- Il implémente des algorithmes rapides pour l'optimisation en termes de temps de calcul et gestion de mémoire [77].
- Il fourni tout un ensemble de paramètres qui permettent d'évaluer les performances des SVM après chaque processus d'apprentissage, et plus concrètement :
    - Il montre une estimation de la dimension de Vapnik-Chervonenkis.
    - Il calcule la précision sur la base d'apprentissage selon une procédure « leave-one out » [78].

- Il permet d'aller au-delà de la classification pour aborder des problèmes de régression, et il admet une variante appelée SVM-Struct pour les problèmes multi variés et structurés.

4.4 Description de notre système de reconnaissance

4.4.1 Acquisition de l'image

Cette étape est importante car elle se préoccupe de la préparation des documents à saisir, du choix et du paramétrage du matériel de saisie (scanner), ainsi que du format de stockage des images.

Dans notre base de données**,** pour accélérer cette phase nous avons utilisé plusieurs média de saisie (scanner, tablette graphique, et souris) Les images sont de format bitmap avec deux couleurs noire et blanche (binaire).

4.4.2. Prétraitement

Dans cette étape on procède à une série d'opérations :

- Binarisation (pour le cas général), et des techniques pour éliminer le bruit par seuillage
- Dilatation, cette opération est effectuée parfois pour agrandir l'épaisseur du tracé car l'écriture par la tablette graphique est avec une épaisseur très petite.
- Erosion, utilisé pour éliminer le bruit (les pixels éloignés) pour pouvoir extraire le cadre qui inclut le caractère.
- Des opérations de redressement de l'image par des rotations pour remédier aux problèmes d'inclinaisons verticales et horizontales des caractères.

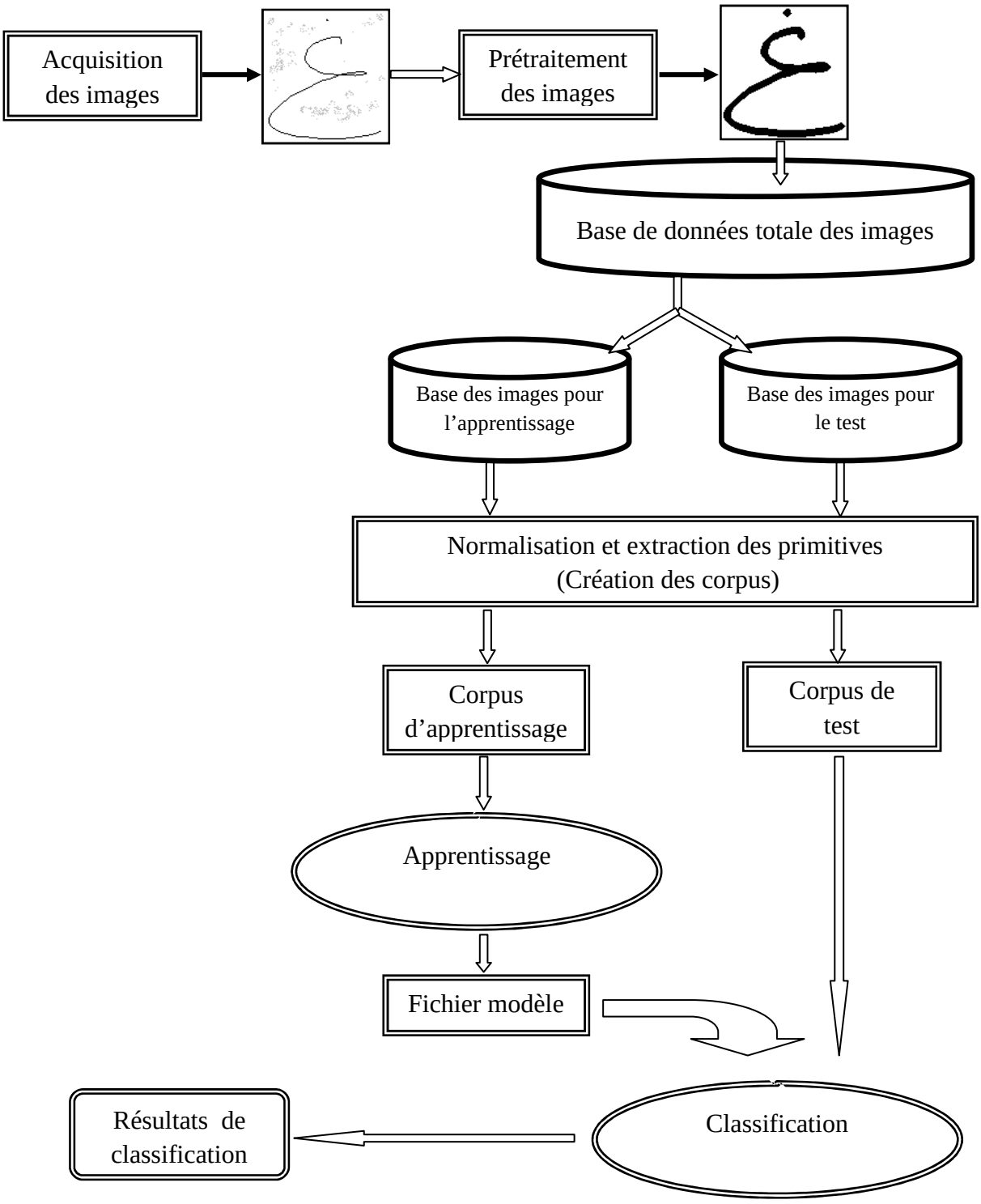


Figure 4.2 Schéma général de notre système de reconnaissance

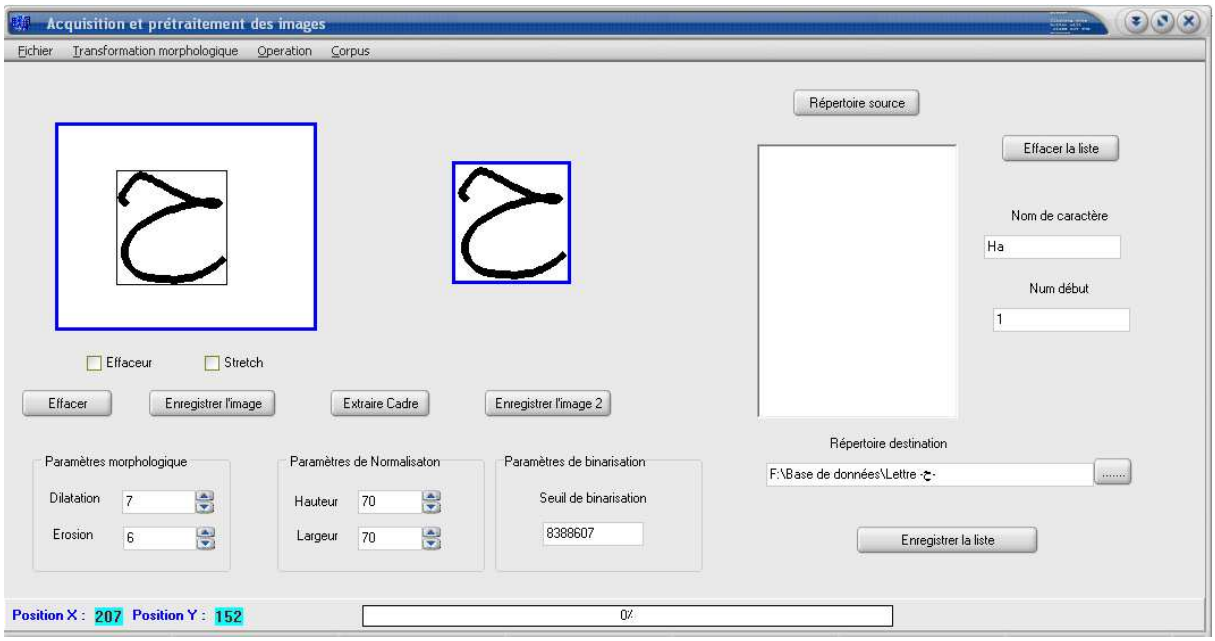


Figure 4.3 Interface d'acquisition et prétraitement des images de notre application

4.4.3 Normalisation

Nous avons effectué cette opération pour éliminer les conditions qui peuvent fausser les résultats, comme la différence de taille. Après la normalisation de la taille, les images de tous les caractères se retrouvent définies dans une matrice de même taille.

Le cadre proposé dans notre système pour la normalisation des caractères, est d'une taille : Hauteur 70 / Largeur 70. Les dimensions de normalisation doivent être suffisamment élevées, pour éviter la perte d'information discriminante.

4.4.4 L'extraction des primitives

C'est l'une des étapes les plus délicates et les plus importantes en OCR. La reconnaissance d'un caractère passe d'abord par l'analyse de sa forme et l'extraction de ses traits caractéristiques (primitives) qui seront exploités pour son identification.

Le choix d'une technique d'extraction des primitives est la principale difficulté rencontrée pendant la réalisation de ce projet, au premier lieu nous avons utilisé les moments de Hough [31] et les moments de Zernik. Malgré que ces moments sont invariant par apport a la translation et le changement d'échelle, mais les résultats obtenus ne sont pas très satisfaisants, en suite nous avons utilisé une technique qui donne un vecteur de caractéristique binaire (1 si le pixel de l'image est noire et 0 si non) avec cette technique nous avons obtenu des bon

résultats mais avec une durée d'apprentissage élevé, vu la taille importante de vecteur de caractéristique (une image 70x70 implique un vecteur de taille 4900). Pour palier à ce problème, nous avons opté la technique de matrice de distribution pour que nous présenterons dans la section suivante.

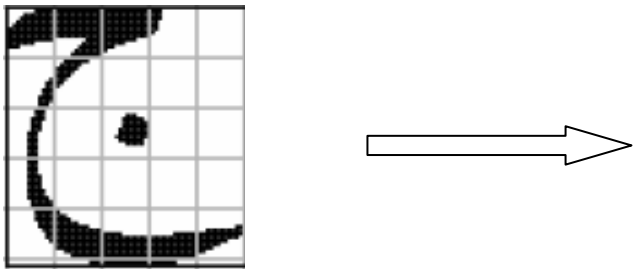
Donc, dans notre système, l'extraction des primitives passe par deux étapes : Construction de la matrice de distribution et détection du point diacritique.

4.4.4.1 Construction de la matrice de distribution

La construction de la matrice de distribution est l'une des phases importantes dans notre système.

Pour une matrice de distribution de taille N x N, le principe consiste à superposer une grille (N×N) sur l'image du caractère et calculer pour chacune des cellules[i,j] résultantes le nombre de pixels noir, puis attribuer ce nombre à la case[i,j] de la matrice de distribution.

Nous considérons par exemple la représentation de la lettre 'jim', qui s'écrit en arabe 'ج' dans sa forme isolée, sur une matrice de distribution de 5x5 (Figure 4.4).



27	40	29	6	0
8	9	0	0	0
12	0	16	0	0
17	0	0	0	0
8	35	24	26	14

Figure 4.4 : Exemple de matrice de distribution (5*5) de la lettre alphabet arabe «jim »

Les différents modèles du 'jim' devront être les plus différents possibles pour couvrir la plus grande gamme de 'jim', mais chacun d'eux devra toujours être plus proche de la classe des 'jim' plutôt que de tout autre classe de lettres.

Pour le système proposé nous avons utilisé une matrice de distribution de taille 7 x 7 et l'image de caractère est de taille 70 x 70 après la normalisation. Nous présentons ci-après notre algorithme qui permet de construire la matrice de distribution.

Algorithme de construction de la matrice de distribution.

```
M : Image de lettre 70 x 70;
NB : compteur ;
D : Matrice de distribution 7 x 7 ;
I, J, L, H : Entier ;

L=0 ;   H=0 ;
Tant que ( L < 70 )  faire
  Tant que ( H < 70 )  faire
    NB=0 ;
    Pour ( I=L jusqu'à L+10)
      Pour ( J=H jusqu'à H+10)
      Si pixel M[I,J] est noire alors NB=NB+1 ;

    D[L div 10, H div 10]=NB ;
    H=H+10 ;

  L=L+10 ;
```

4.4.4.2 Détection du point diacritique

Dans les caractères arabes l'existence des points diacritiques est très important, il y a plus de la moitié des caractères qui possèdent du point qui représente dans beaucoup de cas la seule différence entre deux caractères (deux classes).

Nous avons introduit dans cette phase de notre système un algorithme qui permet de détecter l'existence du point diacritique dans l'image de caractère. Et cette propriété sera ajoutée au vecteur de caractéristique de cette image.

Le principe de notre algorithme est simple, c'est de parcourir l'image de caractère par un petit carrée (10 x 10) et si nous avons trouvé que le centre de carré est noir et tous les pixels de périmètre sont blancs donc il y a un point diacritique dans l'image.

4.4.4.3 Structure de vecteur de caractéristique

Dans notre système de reconnaissance le vecteur de caractéristique est de taille 59, les 49 premier représentent les valeurs de matrice de distribution (7x7) et les dix derniers représentent la valeur VP, déterminé comme suit :

$$VP = \begin{cases} 100 & S'il\ existe\ un\ point\ diacritique\ dans\ l'image\ de\ caractère. \\ 0 & Sinon. \end{cases}$$

La valeur VP est répétée dix fois pour donner un poids considérable à cette caractéristique. Exemple de vecteur des caractéristiques de la lettre 'Jim' à la fin « ـج ».

1	2	3	4	…	44	45	46	47	48	49	50	51	52	…	58	59
43	60	60	70	…	65	60	60	61	14	0	100	100	100	…	100	100

4.4.4.4 Corpus

Le corpus est un fichier texte possède une structure particulière pour pouvoir être utilisée directement par l'application *SVM*^{multiclass}. Chaque ligne du corpus représente le vecteur de caractéristique d'une image, la première valeur de la ligne est le numéro de la classe, ensuite chaque valeur de vecteur de caractéristique est précédé par un index (Index:Valeur).

Nous présentons ci-dessous une ligne de corpus d'apprentissage pour la lettre 'Jim' à la fin « ـج » qui a le numéro de la classe 41.

41 1:43 2:60 3:60 4:70 5:65 6:29 7:0 8:0 9:0 10:34 11:63 12:40 13:2 14:0 15:0 16:49 17:45 18:46 19:30 20:30 21:31 22:45 23:32 24:0 25:21 26:30 27:30 28:25 29:61 30:0 31:8 32:19 33:0 34:0 35:0 36:61 37:2 38:0 39:0 40:0 41:0 42:0 43:19 44:65 45:60 46:60 47:61 48:14 49:0 50:100 51:100 52:100 53:100 54:100 55:100 56:100 57:100 58:100 59:100

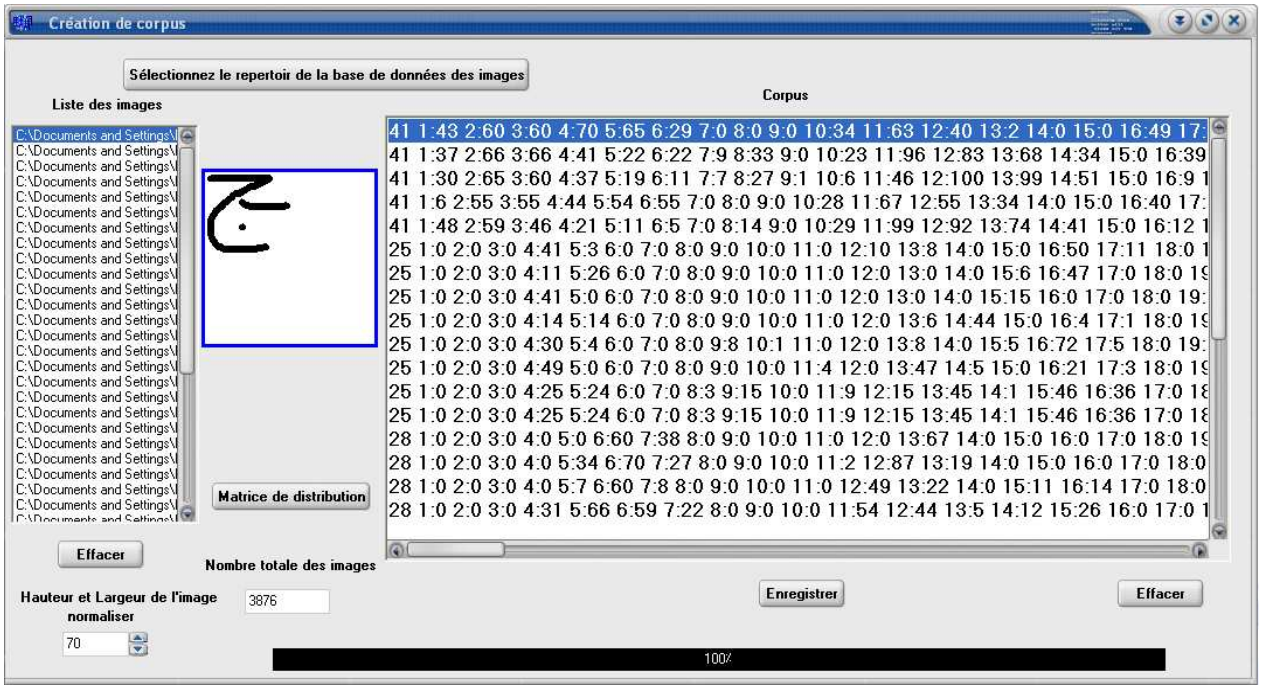


Figure 4.5 Interface de création des corpus.

4.5 Expérimentations et Résultats

Notre objectif est d'effectuer une série d'expérimentations et une recherche exhaustive afin de sélectionner le type du classification une-contre-une ou une-contre-reste, le noyau et les paramètres du noyau qui donnent les meilleurs résultats en termes de taux de prédiction et le temps CPU. Tout cela en utilisant la recherche tabou pour faire une exploration intelligente de l'espace de recherche.

En premier lieu nous avons divisé notre base de données des images en deux, une pour l'apprentissage contenant 3840 images et l'autre pour le test contenant 1000 images, ensuite nous avons créé les deux corpus correspondants (corpus d'apprentissage et corpus de test).

Pour la sélection nous avons implémenté notre stratégie de recherche basée sur la recherche tabou (présentée dans le chapitre précédent) pour varier est tester les hyper-paramètres du SVM. Dans cette étude nous avons utilisé six types de noyaux qui sont : le noyau polynomial, Gaussien (RBF), Laplacien, Sigmoïde, Linéaire et Linéaire normalisé. Les résultats obtenus sont très encourageants et sont présentés dans ce qui suit.

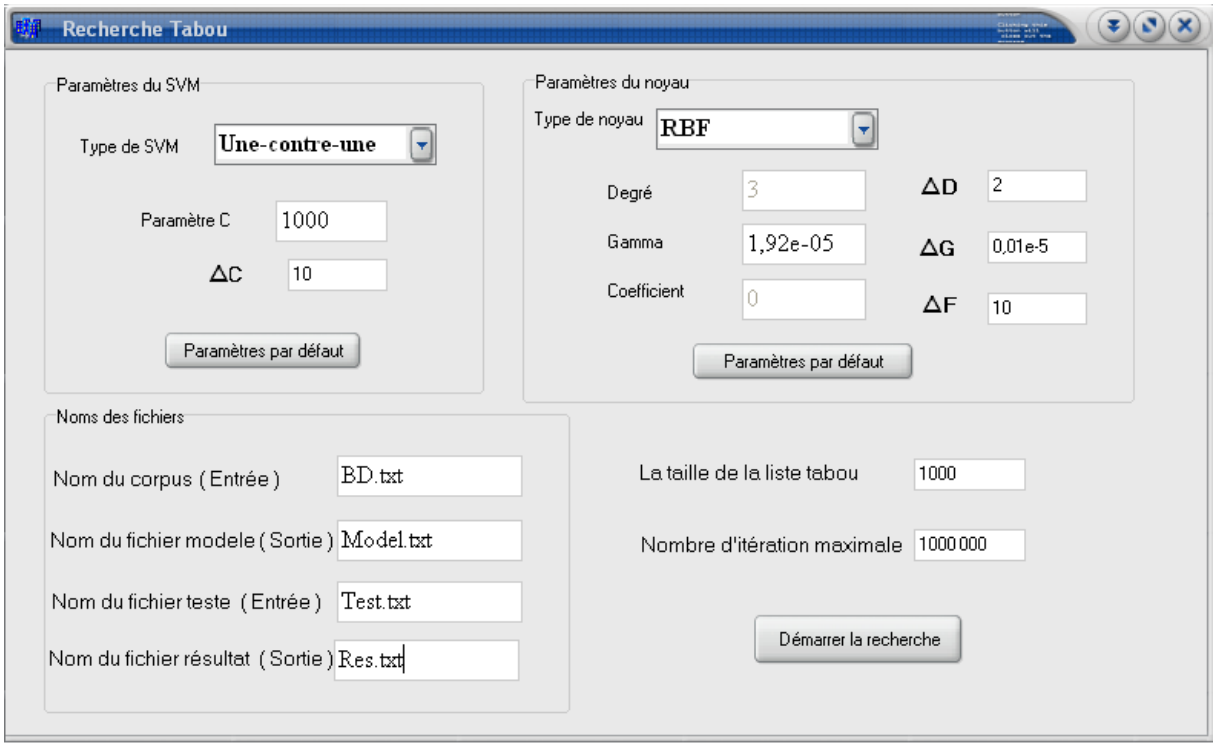


Figure 4.6 Interface de la recherche tabou.

4.5.1 SVM une-contre-reste

Nous présentons ici quelques résultats d'expérimentation en utilisant le type du SVM une-contre-reste, la dernière ligne de chaque table représente le meilleur résultat que nous avons trouvé pour chaque noyau.

- Noyau polynomial : $K(x, x') = (x.x' + coef)^d$

Paramètre C	Paramètre D	Coef	Durée d'apprentissage (s)	Durée de prédiction (s)	Taux de prédiction
1000	3	0	12,45	1,46	95,3 %
1000	10	10	11,46	1,12	95,5 %
1	3	0	12,71	1,39	96,0 %
1	3	1	12,20	1,18	96,4 %

- Noyau Gaussien (RBF) : $K(x, x') = exp^{-\|x-x'\|^2 * \sigma}$

Paramètre C	Paramètre σ	Durée d'apprentissage (s)	Durée de prédiction (s)	Taux de prédiction
1000	1,92 E -05	8,82	1,14	96,6 %
100	1,92 E -05	9,50	1,14	96,9 %
100	1,83 E -05	9,73	1,12	97,0 %

- Noyau Laplacien : $K(x, x') = exp^{-\sqrt{\|x-x'\| * \sigma}}$

Paramètre C	Paramètre σ	Durée d'apprentissage (s)	Durée de prédiction (s)	Taux de prédiction
1000	1,92 E -05	16,09	1,68	95,9 %
100	1,02 E -07	12,98	1,57	96,4 %
1000	1,02 E -07	12,93	1,57	96,6 %

- Noyau Sigmoïde : $K(x, x') = tanh(\sigma(x.x') + coef)$

Paramètre C	Paramètre σ	Coef	Durée d'apprentissage (s)	Durée de prédiction (s)	Taux de prédiction
1000	1,92 E -05	-10	13,21	2,15	58,5 %
1000	1,92 E -06	0	13,17	1,29	93,7 %
100	1,92 E -06	0	11,21	1,46	94,1 %
307	1,68 E -06	0	13,34	1,15	94,7 %

- Noyau Linéaire : $K(x, x') = x.x'$

Paramètre C	Durée d'apprentissage (s)	Durée de prédiction (s)	Taux de prédiction
5	10,81	0,78	90,8 %
1	11,84	0,76	91,0 %

- Noyau Linéaire normalisé : $K(x, x') = \frac{x.x'}{\sqrt{x_1.x_2 * x_1'.x_2'}}$

Paramètre C	Durée d'apprentissage (s)	Durée de prédiction (s)	Taux de prédiction
1000	9,20	1,39	92,4 %
100	9,53	1 ,71	94,6 %
161	8,95	1.98	94,7 %

4.5.2 SVM une-contre-une

Nous présentons ici quelques résultats d'expérimentation en utilisant le type du SVM une-contre-une, la dernière ligne de chaque table représente le meilleur résultat que nous avons trouvé pour chaque noyau.

- Noyau polynomial : $K(x, x') = (x.x' + coef)^d$

Paramètre C	Paramètre D	Coef	Durée d'apprentissage (s)	Durée de prédiction (s)	Taux de prédiction
1000	10	0	23,59	1,48	91,7 %
10	3	0	23,37	1,50	95,2 %
1000	3	0	22,75	1,45	96,5 %
1000	3	10	22,93	1,53	97,0 %

- Noyau Gaussien (RBF) : $K(x, x') = exp^{-\|x-x'\|^2 * \sigma}$

Paramètre C	Paramètre σ	Durée d'apprentissage (s)	Durée de prédiction (s)	Taux de prédiction
1000	1,02 E -05	22,87	1,43	97,1 %
10	1,92 E -05	24,50	1,50	97,4 %
1000	1,92 E -05	24,03	1,68	97,7 %
1000	1,83 E -05	23,78	1.48	97,9 %

- Noyau Laplacien : $K(x, x') = exp^{-\sqrt{\|x-x'\|*\sigma}}$

Paramètre C	Paramètre σ	Durée d'apprentissage (s)	Durée de prédiction (s)	Taux de prédiction
10	1,92 E -05	28,28	1,82	95,0 %
1000	1,02 E -05	27,39	1,79	96,2 %
1000	1,92 E -05	28,40	1,85	96,3 %

- Noyau Sigmoïde : $K(x, x') = tanh(\sigma(x.x^{'}) + coef)$

Paramètre C	Paramètre σ	Coef	Durée d'apprentissage (s)	Durée de prédiction (s)	Taux de prédiction
1000	1,92 E -05	1	22,23	1,82	58,6 %
1000	1,92 E -05	0	22,75	1,71	65,2 %
100	1,92 E -05	0	22,42	1,84	68,4 %
1000	1,00 E -06	0	22,81	1,54	97,2 %

- Noyau Linéaire : $K(x, x') = x.x'$

Paramètre C	Durée d'apprentissage (s)	Durée de prédiction (s)	Taux de prédiction
10	22,23	1,28	96,9 %
1000	22,14	1,34	96,9 %

- Noyau Linéaire normalisé : $K(x, x') = \frac{x.x'}{\sqrt{x_1.x_2 * x'_1.x'_2}}$

Paramètre C	Durée d'apprentissage (s)	Durée de prédiction (s)	Taux de prédiction
1000	21.57	2.25	96,9 %
100	22.12	2.20	97,4 %
70	21.68	2.48	97,6 %
77	22.00	2.39	97,8 %

4.6 Evaluation des résultats

Les nombreuses expérimentations que nous avons réalisées montrent l'efficacité de notre système dans son intégrité et en particulier l'efficacité de notre stratégie pour la sélection des paramètres avec l'utilisation de la recherche tabou pour le choix et le balayage d'un grand espace de paramètres. Ce choix a une grande influence sur les performances du classificateur final, et aussi sur le temps de calcul.

Nous avons remarqué que le type du SVM une-contre-une conduit à des résultats meilleurs que le SVM une-contre-reste au dépend d'un temps d'exécution plus important. Nous avons constaté aussi que le noyau RBF est le plus adapté à la reconnaissance des caractères arabes manuscrits.

En effet nous obtenons avec ce noyau des résultats meilleurs que les autres noyaux notamment un taux de reconnaissance de 97,0% (pour le SVM une-contre-reste, $\sigma = 1{,}83\ \mathrm{E} - 05\ , \mathrm{C} = 100$) et un taux de reconnaissance égale à 97,9% (pour le SVM une-contre-une, $\sigma = 1{,}83\ \mathrm{E} - 05\ , \mathrm{C} = 1000$).

Nous avons remarqué aussi que le noyau linéaire et linéaire normalisé donnent des résultats important dans l'approche une-contre-une (97,8%). Les deux noyaux sont un petit peu stable au changement du paramètre C, par contre, le noyau sigmoïde ($\mathrm{K}(\mathrm{x}, \mathrm{x}') = \tanh(\sigma(\mathrm{x}.\mathrm{x}') + \mathrm{coef})$) est sensible au changement du paramètre σ et nécessite une recherche intensive pour trouver des bons résultats. Pour le noyau polynomial ($\mathrm{K}(\mathrm{x}, \mathrm{x}') = (\mathrm{x}.\mathrm{x}' + \mathrm{coef})^{\mathrm{d}}$), il faut que le paramètre coef prenne une valeur différente de zéro pour tirer des meilleurs résultats. Néanmoins pour le noyau sigmoïde ce paramètre n'a pas une grande importance. Le noyau laplacien quant à lui donne des résultats similaires entre les deux approches une-contre-une et une-contre-reste.

Enfin, ces résultats montrent que notre algorithme de sélection de modèles SVM basé sur la recherche tabou permet d'obtenir des résultats meilleurs par rapport à l'état d'art actuel.

4.7 Conclusion

Nous avons présenté dans ce chapitre une description détaillée de notre système de reconnaissance de caractères Arabes manuscrits hors-ligne, avec l'évaluation de performance de chaque phase de processus de reconnaissance. Nous avons présenté aussi les résultats d'expérimentations par l'utilisation de deux types du SVM multi-classe (une-contre-une et une-contre-reste) et les différents noyaux.

Notre système a donné des bons résultats au niveau de la reconnaissance, cela montre l'efficacité de la méthode utilisée pour l'extraction des primitives et la stratégie utilisée pour la recherche et la sélection des paramètres du modèle SVM multi-classe. Notre système a pu atteindre un taux de reconnaissance élevé, sans avoir sacrifié beaucoup de la simplicité du système ou du temps d'exécution.

Conclusion générale

La reconnaissance de l'écriture arabe manuscrite (hors ligne ou en ligne) est un axe de recherche récent et qui est prometteur pour des travaux futurs en vue d'augmenter le taux de reconnaissance, la fiabilité, et d'optimiser les performances de nos machines et l'intégrité des systèmes de reconnaissance. Pour les améliorer, il faut chercher des nouvelles méthodes plus efficaces que les anciennes avec moins d'erreurs.

Le SVM est l'une des méthodes de classification à avoir plus marqué la reconnaissance de formes en fournissant un cadre théorique statistique et non plus connexionniste. Dans ce mémoire, nous avons proposé une méthode pour la sélection optimale du modèle SVM pour la reconnaissance des caractères arabes manuscrits.

Notre système a été testé sur une base contenant 4840 images. Nous avons constitué notre propre base puisqu'il n'a pas une base standard pour les lettres arabes manuscrites, qui nous permet de comparer notre système avec ceux de la littérature.

Notre système a donné des très bons résultats au niveau de la reconnaissance, cela montre l'efficacité de la méthode utilisée pour l'extraction des primitives et la stratégie utilisée pour la recherche et la sélection des paramètres de modèles SVM multi-classes.

Cette expérience s'est avérée intéressante et encourageante mais des possibilités d'extensions futures restent envisageables :

- Utiliser une base d'image issue de la segmentation d'une base réelle de mots arabes, pour que ce système soit une continuité pour d'autres systèmes.
- Vérifier d'autres types de caractéristiques pour améliorer encore plus le taux de la reconnaissance.
- Notons enfin la nécessité de posséder une base et un protocole commun de validation de résultats entre les différentes approches utilisées en reconnaissances de l'écriture arabe manuscrite.

Bibliographie

[1] Mustapha Kadri, « Méthode de reconnaissance de l'écriture arabe manuscrite en utilisant les réseaux neuronaux » Mémoire de magister Université des Sciences et des Technologies Mohamed Boudiaf d'Oran, 2010.

[2] Zaïz Faouzi et al « SVM pour la reconnaissance de caractères manuscrits arabes » Laboratoire LESIA, Département d'Informatique, Université Mohamed Khider Biskra, 2010.

[3] Nedjem eddine Ayat, « sélection de modèle automatique des machines à vecteurs de support: application à la reconnaissance d'images de chiffres manuscrits » Thèse de doctorat montréal, le 20 janvier 2004.

[4] Muhammad Sarfraz, « Computer -Aided intelligent recognition techniques and applications » (section : Offline Arabic Character Recognition) King Fahd University of Petroleum and Minerals, Kingdom of Saudi Arabia, 2005.

[5] S. Snoussi maddouri, « Modèle perceptif neuronal à vision globale-locale pour la reconnaissance de mots manuscrits arabes », 2002.

[6] Haitaamar Schahrazed, « Segmentation de textes en caractères pour la reconnaissance optique de l'écriture arabe ». Thèse de magister université Elhadj-Lakhdar Batna, 2007.

[7] Richard Alan Peters II, « Image processing (digital image and matlab, frequency filtering) », 2007.

[8] Somaya A. S. Al-Ma'adeed, « Recognition of Off-line Handwritten Arabic Words », University of Nottingham for the degree of Doctor of Philosophy, June 2004.

[9] N. Ben Amara « Utilisation des modèles de Markov cachés planaires en reconnaissance de l'écriture arabe imprimée ». These de doctorat, specialite Genie Electrique, Universite des sciences, des Techniques et de medecine de Tunis II, 1999.

[10] B. Al-Badr, S.A. Mahmoud: « Survey and bibliography of Arabic optical text recognition ». Signal processing, vol. 41, pp. 49-77, 1995.

[11] I.R. Tsang: « Pattern recognition and complex systems ». These de doctorat, université d'Anterwerpen, 2000.

[12] Fahmy, S.Al Ali: « Automatic recognition of handwritten Arabic characters using their geometrical features ». Studies in informatics and control journal (SIC journal), vol. 10, No 2, 2001

[13] J. Anigbogu : « Reconnaissance de textes imprimes multi-fontes a l'aide de modèles

stochastiques et métriques ». thèse de doctorat, Université de Nancy I, 1992.

[14] Riadh Bouslimi, « Système de reconnaissance hors-ligne des mots manuscrits arabe pour multi-scripteurs » mémoire de master soutenue le 21/10/2006.

[15] Abdel Belaïd, « Reconnaissance automatique de l'écriture et du document » LORIA-CNRS Campus scientifique B.P. 239 54506 Vandœuvre-lès-Nancy, 2002

[16] A. Belaïd, « Reconnaissance de l'Écriture et Analyse de Documents : Numérisation, Prétraitements », 2006.

[17] Maged Mohamed Mahmoud Fahmy, « Automatic Recognition Of Handwritten Arabic Characters Using Their Geometrical Feature », 2000.

[18] O. D. Trier and T. Taxt. « Evaluation of binarization methods for document images, On Pattern Analysis and Machine Intelligence », vol. 11, n.12, pp. 312- 314, December 1995.

[19] Y. Liu and S. Srihari. « Document image binarization on texture features, On Pattern Analysis and Machine Intelligence », vol. 19, n.5, pp. 540-544, May 1997.

[20] H. Emptoz, F. Lebourgeois, V. Eglin, Y. Leydier. « La reconnaissance dans les images numérisées : OCR et transcription, reconnaissance des structures fonctionnelles et des métadonnées », 2003.

[21] Sylvain Chevalier, « Reconnaissance d'écriture manuscrite par des techniques markoviennes: une approche bidimensionnelle et générique ». Thèse de doctorat de l'Université René Descartes - Paris 5, 2004.

[22] A. Belaïd, « Analyse et reconnaissance des documents », http://www.loria.fr/~abelaid/Teaching (support de cours), 2006.

[23] Dr Suici-Meslati, « Reconnaissance de formes et écriture arabe manuscrite », support de cours 2008.

[24] Najoua Ben Amara et al, « Modélisation Pseudo Bidimensionnelle pour la Reconnaissance de Chaînes de Caractères Arabes Imprimés », 1999.

[25] Yousef Al-Ohali, MohamedCheriet et Ching Suen, « Databases for recognition of handwritten Arabic cheques ». Pattern Recognition Society 36 111 – 121. 2003.

[26] Najoua Ben Amara et al, « Une méthode stochastique pour la reconnaissance de l'écriture arabe imprimée », 1996.

[27] Philippe BOLON ET AL, « Analyse d'image, Traitements de bas niveau » (support de cours), 1998.

[28] M. Mansour, M. Benkhadda & A. Benyettou, « Optimized Segmentation Techniques

for Handwritten Arabic Word and Numbers Character Recognition », 2005.

[29] S. Kermi : « Classifieur neuronal base connaissances, application a la reconnaissance des caractères arabes isoles manuscrits ». Thèse de magister, université Badji Mokhtar, Annaba, Algérie 1999.

[30] C.H. TEH et R.T. CHIN: « On image analysis by the methods of moments ». IEEE Trans. Pattern Analysis and Machine Intelligence, 10(4):496–513.

[31] A. Choksuriwong et al, « Etude Comparative de Descripteurs Invariants d'Objets » ENSI de Bourges – Université d'Orléans Laboratoire Vision et Robotique - UPRES EA 2078, 2005

[32] Denis Arrivault, « Apport des Graphes dans la Reconnaissance Non-Contrainte de Caractères Manuscrits Anciens » Thèse de doctorant de l'université de Poitiers, 2006

[33] Muhammad Sarfraz, « Computer-Aided intelligent recognition techniques and applications » (section: Offline Arabic Character Recognition) King Fahd University of Petroleum and Minerals, Kingdom of Saudi Arabia, 2005.

[34] Trevor Hastie and Patrice Y. Simard, « Metrics and Models for Handwritten Character Recognition ». Statistical science Vol. 13 N° 1, 54 -65, 1998.

[35] Sander Bohte, « Spiking neural networks », Thesis University Leiden.2003,

[36] Antoine Cornuéjol et Laurent Miclet, Livre : « Apprentissage artificiel : concepts et algorithmes », Préface de Tom Mitchell. Édition EYROLLES deuxième tirage 2003.

[37] P. J. Werbos. Beyond regression : « New Tools for Prediction and Analysis in the Behavioral Sciences ». PhD thesis, Masters Thesis, Harvard University, 1974.

[38] D. E. Rumelhart et al. « Learning internal representations by error backpropagation ». In J. L. McClelland In D. E. Rumelhart and the PDP research group, editors, Parallel distributed processing: explorations in the microstructure of cognition, pages 318–362. 1986.

[39] Y. LeCun et al. « Backpropagation applied to handwritten zip code recognition ». Neural Computation, 1(4):541–551, Winter 1989.

[40] A. Kolmogorov. « Three approaches to the quantitative definition of information ». Problems Inform. Transmission, 1(1) :1–7, 1965.

[41] S. Alma'adeed, C. Higgens, D. Ellima,« Recognition of Off-Line Handwritten Arabic Words Using Hidden Markov Model Approach », ICPR, University of Nottingham, 2002.

[42] Aburas A. A., Rehiel S. M. A., « Off-line Omni-style Handwriting Arabic Character

Recognition System Based on Wavelet Compression », International Islamic University Malaysia, Electrical and Computer Engineering, Malaysia, 2007.

[43] Jannoud I. A., « Automatic Arabic Hand Written Text Recognition System ». Damascus University, Damascus, Syria and Al-zaytoonah University, Amman, Jordan, American Journal of Applied Sciences 4 (11): 857-864, 2007.

[44] Sari T., Souici L., Sellami M., « Off-line Handwritten Arabic Character Segmentation Algorithm: ACSA », Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR'02), Algeria, 2002.

[45] V.N. Vapnik. « Estimation of Dependences Based on Empirical Data ». springer Verlag, Berlin, 1982.

[46] V. Blanz, B. Scholkopf, H.H. Bulthoff, C. Burges, V. Vapnik, and T. Vetter. « Comparison of view-based object recognition algorithms using realistic 3d models ». In ICANN, pages 251–256, 1996.

[47] C. Cortes and V. Vapnik. « Support-vector networks. Machine Learning », 20(3):273 297, 1995.

[48] BISSON, G. (2009). « Intelligence artificielle ».

[49] H.Mohamadally, B.Fomani « SVM : Machines à Vecteurs de Support ou Séparateurs à Vastes Marges » Versailles St Quentin, France 16 janvier 2006

[50] J.Milgram, « Contribution à l'intégration des machines à vecteurs de support au sein des systèmes de reconnaissance des formes : Application à la lecture automatique de l'écriture manuscrite ». MONTRÉAL, LE 29 JUIN 2007

[51] Steve R. Gunn, « Support Vector Machines for Classification and Regression » University of Southampton, 10 May 1998

[52] VAPNIK, V. (1998). « The Nature of Statistical Learning Theory ». Springer-Verlag.

[53] BOTTOU, L. et CHIH-JEN, L. « Support Vector Machine Solvers, in Large Scale Kernel Machines ». MIT Press. 2007.

[54] Ludovic Mercier, « Les machines à vecteurs support pour la classification en imagerie hyperspectrale : implémentation et mise en œuvre. » UE ENG111 - Epreuve TEST Travail d'Etude et de Synthèse Technique en Informatique, 11 février 2010

[55] Anis Ben Ishak, « Sélection de variables par les machines à vecteurs supports pour la discrimination binaire et multiclasse en grande dimension » Thèse de doctorat de l'université de la méditerranée, Tunis, 2007.

[56] S. Abe and T. Inoue. « Fuzzy support vector machines for multiclass problems ». In

Proceedings of the Tenth European Symposium on Artificial Neural Networks, pages 116-118, Bruges, Belgium, (2002).

[57] S. Abe. « Analysis of multiclass support vector machines ». In Proceedings of International Conference on Computational Intelligence for Modeling Control and Automation, pages 385-396, Vienna, Austria, (2003).

[58] S. Knerr, L. Personnaz, and G. Dreyfus. « Single-layer learning revisited: a stepwise procedure for building and training a neural network ». In Neurocmputing: Algorithms, Architectures and Applications, J. Fogelman, editor, Springer-Verlag, (1990).

[59] U. H. G. Kreßel. « Pairwise classification and support vector machines ». In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, Advances in Kernel Methods Support Vector Learning, pages 255-268, The MIT Press, Cambridge, (1999).

[60] J. H. Friedman. « Another approach to Polychotomous classification ». Technical report, Department of Statistics, Stanford University, (1996).

[61] S. Abe and T. Inoue. « Fuzzy support vector machines for multiclass problems ». In Proceedings of the Tenth European Symposium on Artificial Neural Networks, pages 116-118, Bruges, Belgium, (2002).

[62] J. C. Platt, N. Cristianini, and J. Shawe-Taylor. « Large margin DAGs for multiclass classification ». In S. A. Solla, T. K. Leen, and K. R. Müller, editors, Advances in Neural Information Processing Systems 12, pages 547-553, The MIT Press, (2000).

[63] M. Pontil and A. Verri. « Support vector machines for 3-d object recognition ». IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(6) : 637-646, (1998).

[64] B. Kijsirikul, N. Ussivakul, and S. Meknavin. « Adaptive directed acyclic graphs for multiclass classification. In PRICAI 2002, pages 158-168, (2002).

[65] T. Phetkaew, B. Kijsirikul, and W. Rivepiboon. « Reordering adaptive directed acyclic graphs for multiclass support vector machines ». In Proceedings of the Third International Conference on Intelligent Technologies, (2002).

[66] T. Phetkaew, B. Kijsirikul, and W. Rivepiboon.Multiclass « Classification of Support Vector Machines by Reordering Adaptive Directed Acyclic Graph ». In International Workshop on Intelligent Systems, (2003).

[67] HSU, C.-W., CHANG, C.-C. et LIN, « A practical guide to support vector classification ». Rapport technique, National Taiwan University.2009

[68] C.-C. CHANG & C.-J. LIN. « LIBSVM: a library for support vector machines ». Sofware Available at http://www.csie.ntu.edu.tw/˜cjlin/libsvm, 2001.

[69] J. LEE & C. LIN. « Automatic Model Selection for Support Vector Machines. Technical Report ». http://www.csie.ntu.edu.tw /˜cjlin/papers/modelselect.ps.gz, 2000.

[70] J. DRÉO, A. PÉTROWSKI, P. SIARRY & E. TAILLARD. « Métaheuristiques pour l'optimisation difficile ». Groupe Eyrolles, 2003.

[71] Gilles Lebrun, « Sélection de modèles pour la classification supervisée avec des SVM(Séparateurs à VasteMarge). Application en traitement et analyse d'images ». DOCTORAT de l'UNIVERSITE de CAEN/BASSE-NORMANDIE, soutenue le 24 novembre 2006.

[72] HOLLAND, J. « Adaptation in Natural and Artificial Systems ». The University of Michigan Press,1975.

[73] RENNARD, J.-P. « Vie artificielle ». Vuibert informatique.2002.

[74] F. GLOVER. « Tabu search: part I ». Dans *on Computing*, 1(3), pages 190–206, 1989.

[75] F. GLOVER. « Tabu search: part II ». Dans *on Computing*, 2(1), pages 4–32, 1989.

[76] G. C. CAWLEY. «Model Selection for Support VectorMachines via Adaptive Step-Size Tabu Search ». Dans *ICANNGA*, 2001.

[77] Thorsten Joachims, « Making Large-Scale SVM Learning Practical »University of Dortmund, Computer Science Department,1998

[78] Robert Azencott, « Application des SVM pour la discrimination de contenus musicaux » Cours d'introduction aux théories de l'apprentissage, Paris, le 31janvier 2005

[79] Ouarda Hachour, « Reconnaissance hybride des caractères Arabes imprimés » Laboratoire Communication Parlée, Centre de Recherche Scientifique et Technique pour le Développement de la langue Arabe (C.R.S.T.D.L.A) Université d'Alger.2004.

[80] Abderrazak Zahour,et AL , « Contribution à la segmentation de textes manuscrits anciens » ,2004