الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي و البحث العلمي

جامعة وهران للعلوم و التكنولوجيا محمد بوضياف

THÈSE

En vue de l'obtention du

Diplôme de Doctorat en Sciences

Présenté par : MAMOUNI El Mamoun

Intitulé

Sélection automatique de modèle SVM: application à la reconnaissance d'écriture arabe manuscrite.

Faculté: ***des Mathématiques et Informatique***

Département: ***Informatique***

Spécialité: ***Informatique***

Option: ***Informatique***

Devant le Jury Composé de :

Membres de Jury	*Grade*	*Qualité*	*Domiciliation*
Mr DJEBBAR Bachir	*Professeur*	*Président*	*USTO-MB*
Mr SADOUNI Kaddour	*Professeur*	*Directeur de Thèse*	*USTO-MB*
Mr BOUAMRANE Karim	*Professeur*	*Examinateur*	*Univ- Oran 1.AB*
Mr ATMANI Baghdad	*Professeur*	*Examinateur*	*Univ-Oran 1.AB*
Mr MEFTAH Boudjelal	*MCA*	*Examinateur*	*Univ-Mascara.MS*
Mr TLEMSANI Redouane	*MCA*	*Examinateur*	*USTO-MB*
Mr ZENNAKI Mahmoud	*MCB*	*Invité*	*USTO-MB*

Année Universitaire : 2019/2020

Remerciements

Je tiens à adresser mes remerciements à mon directeur de thèse Monsieur SADOUNI Kaddour, Professeur à l'Université des Sciences et de la Technologie d'Oran, Mohammed BOUDIAF, pour sa vision scientifique, son soutien constant et ses conseils avisés.

Je souhaite exprimer ma gratitude à Monsieur DJEBBAR Bachir, Professeur à l'Université des Sciences et de la Technologie d'Oran, Mohammed BOUDIAF, qui m'a fait l'honneur de faire partie du jury de cette thèse et d'en être le Président.

J'adresse mes remerciements à Monsieur BOUAMRANE Karim, Professeur à l'Université d'Oran 1, Ahmed BEN BELLA, d'avoir accepté de faire partie du jury de cette thèse et pour le temps et l'attention qu'il a consacré à la lecture et la critique de ce travail.

Je ne manquerai pas d'adresser mes remerciements à Monsieur ATMANI Baghdad, Professeur à l'Université d'Oran 1, Ahmed BEN BELLA, d'avoir accepté d'être membre de ce jury et pour son intérêt, ses remarques et ses critiques constructives.

Que Monsieur MEFTAH Boudjelal, Maitre de Conférences A à l'Université de Mascara, Mustapha Stambouli, trouve ici ma reconnaissance d'avoir accepté de faire partie de ce jury et de l'attention qu'il a porté à cette thèse ainsi que le temps qu'il a consacré à la lecture et l'évaluation de ce travail.

Je remercie également Monsieur TLEMSANI Redouane, Maitre de Conférences A à l'Université des Sciences et de la Technologie d'Oran, Mohammed BOUDIAF, qui a accepté lui aussi de faire partie de ce jury et pour le temps de lecture attentive et ses réflexions précieuses.

Mes remerciements s'adressent aussi à Monsieur ZENNAKI Mahmoud, Maitre de Conférences B, à l'Université des Sciences et de la Technologie d'Oran, Mohammed BOUDIAF, pour son aide et son soutien pendant la réalisation de cette thèse.

Et enfin, mes chers parents et familles, trouvent ici l'expression de mes remerciements les plus sincères et les plus profonds en reconnaissance de leur sacrifice, soutien et encouragement.

ملخص

تتمثل مساهمة هذه الأطروحة في إجراء دراسة تجريبية تهدف إلى الاختيار التلقائي لنماذج آلة الأشعة الحاملة أو ما يعرف بـ machine à vecteurs de support (SVM) للتعرف على الكتابة العربية المكتوبة بخط اليد. يتطلب استخدام طريقة آلة الأشعة الحاملة اختيار العديد من المعلمات ، مثل نوع استراتيجية التصنيف (واحد مقابل الكل أو واحد مقابل واحد) ، ومعلمة التنظيم، ونوع النواة ومعلماتها. اختيار هذه المعلمات له تأثير كبير على أداء المصنف النهائي وكذلك على زمن المعالجة.

في هذه الأطروحة، استخدمنا طريقة التحقق من الصحة المتقاطعة (validation croisée)، وشبكة البحث (grille de recherche)، وطريقة البحث المحرم (recherche Tabou)، وتقنية تحسين سرب الجسيمات (PSO) التي سمحت لنا باختيار المعلمات وتحليل مساحة كبيرة بشكل سريع. تم تقديم أيضًا دراسة مقارنة لنماذج آلة الأشعة الحاملة لدراسة سرعة التقارب ونتائج كل نموذج. لقد أنجزنا نظامًا كاملاً للتعرف على الأحرف العربية المكتوبة باليد ، أظهرت النتائج التي تم الحصول عليها أن استخدام الطرق الفوقية المساعدة على البحث (méta-heuristique) يمكن أن يكون مفيدًا في التعرف بشكل أفضل على الأحرف العربية بواسطة آلة الأشعة الحاملة.

تُعد تجزئة النص العربي المكتوب بخط اليد مهمة أساسية في تطوير نظام التعرف على الكتابة، حيث أنه إذا لم يتم تجزئة كلمة ما بشكل صحيح حتماً لن يتم التعرف على الأحرف المكونة لهذه الكلمة. في هذا العمل، اقترحنا أيضًا طريقة جديدة لتقسيم الكلمات إلى أحرف، يتم استخراج نقاط التجزئة باستخدام تحليل فعال لمخطط الرسم البياني للإسقاط العمودي. تم اختبار نهجنا باستخدام مجموعة من الكلمات من قاعدة بيانات IFN / ENIT ، وتم الحصول على نتائج واعدة.

الكلمات المفتاحية : آلة الأشعة الحاملة، اختيار النموذج، التعرف على الأحرف العربية، الطرق الفوقية المساعدة على البحث (méta-heuristique) ، تحسين سرب الجسيمات، البحث المحرم، التجزئة.

Abstract

The contribution of this thesis is to carry out an experimental study aiming to do an automatic model selection of support vector machines (SVM) for recognition of handwritten Arabic script. The use of the SVM method requires the selection of several parameters, such as the type of multi-class strategy (one-against-all or one-against -one), the regularization parameter, the kernel function and their parameters. The choice of these parameters has a great influence on the performances of the final classifier and on the calculation time.

In this thesis we considered the method of cross validation, Grid search, Tabu search and particle swarm optimization technique (PSO) which allowed to quickly select and analyze a large space of parameters. A comparative study of SVM models is also presented to examine the convergence speed and the results of each model. We have realized a complete system of recognition of handwritten Arabic characters. The results obtained show that the use of meta-heuristics can be beneficial to a better recognition of manuscript Arabic characters by the SVM.

Segmentation of handwritten Arabic text is an essential task in the development of character recognition systems, as poorly segmented characters will automatically be unrecognized. In this work, we also propose a new approach to segment handwritten Arabic characters using an efficient analysis of the vertical projection histogram. Our approach was tested using a set of handwritten Arabic words from the IFN/ENIT database, and promising results were obtained.

Keywords

SVM, model selection, Arabic character recognition, meta-heuristics, PSO, Tabu search, segmentation.

Résumé

La contribution de cette thèse est de procéder à une étude expérimentale ayant pour objectif la sélection automatique de modèle des machines à vecteurs de support(SVM), pour la reconnaissance d'écriture arabe manuscrite. L'utilisation de la méthode SVM nécessite la sélection de plusieurs paramètres, tels que le type de stratégie multi-classe (une-contre-tous ou une-contre-une), le paramètre de régularisation, la fonction noyau et leurs paramètres. Le choix de ces paramètres a une grande influence sur les performances du classifieur final et aussi sur le temps de calcul.

Dans cette thèse nous avons considéré la méthode de validation croisée, grille de recherche, recherche Tabou et la technique d'optimisation par essaims de particules (PSO) qui ont permis de sélectionner et d'analyser rapidement un grand espace de paramètres. Une étude comparative des modèles SVM est également présentée pour examiner la vitesse de convergence et les résultats de chaque modèle. Nous avons réalisé un système complet de reconnaissance des caractères arabes manuscrits hors-ligne. Les résultats obtenus montrent que l'utilisation de méta-heuristiques peut être bénéfique à une meilleure reconnaissance des caractères arabes manuscrits par les SVM.

La segmentation du texte arabe manuscrit est une tâche essentielle dans le développement de système de reconnaissance d'écriture, car un caractère mal segmenté sera automatiquement non reconnu. Dans ce travail, nous proposons aussi une nouvelle approche pour segmenter un mot en caractères, les points de segmentations sont extraits en utilisant une analyse efficace de l'histogramme de projection verticale. Notre approche a été testée à l'aide d'un ensemble de mots de la base de données IFN / ENIT, des résultats prometteurs ont été obtenus.

Mots clés

SVM, sélection de modèle, reconnaissance de caractère arabe, méta-heuristique, PSO, recherche Tabou, segmentation.

Sommaire

Remerciements……..….. I
Résumé……...…... II
Sommaire…….. V
Listes des figures……... IX
Liste des tableaux…….. XII
Introduction générale…… 01

Chapitre I : Reconnaissance de l’écriture arabe manuscrite

1.1 Introduction……...…….. 05
1.2 Caractéristiques de l’écriture Arabe…… 05
1.3 Processus de reconnaissance……... 09
1.3.1 Phase d’acquisition…… 09
1.3.2 Phase de prétraitement…….. 10
1.3.2.1 Binarisation …….. 10
1.3.2.2 Transformation par érosion…….. 11
1.3.2.3 Transformation par dilatation…….. 11
1.3.2.4 Ouverture morphologique…… 12
1.3.2.5 Fermeture morphologique…… 12
1.3.2.6 Squelettisation d'une image…….. 12
1.3.2.7 Normalisation……... 13
1.3.3 Phase de segmentation……... 14
1.3.3.1 Segmentation implicite……. 14
1.3.3.2 Segmentation explicite……. 14
1.3.4 Phase d’extraction des caractéristiques (primitives) ……. 15
1.3.4.1 Caractéristiques topologiques……... 15
1.3.4.2 Caractéristiques structurelles……... 16
1.3.4.3 Caractéristiques statistiques…….. 16
1.3.4.4 Superposition des modèles et corrélation……. 17
1.3.5 Phase de classification……... 17
1.3.5.1 Apprentissage……... 17

1.3.5.2 Reconnaissance et décision…….. 17
1.3.6 Phase de post traitement…… 19
1.4 Conclusion……... 19

Chapitre II : Machines à vecteurs de support

2.1 Introduction…… 21
2.2 Notion d'apprentissage…… 22
2.3 Principe de fonctionnement général…… 22
2.3.1 Linéarité et non linéarité…… 24
2.3.1.1 Cas linéairement séparable……... 24
2.3.1.2 Cas non linéairement séparable…… 24
2.4 Fondements mathématiques…… 26
2.4.1 Principe…….. 26
2.4.2 Forme primale…… 27
2.4.3 Forme duale……... 27
2.5 Principe du noyau…… 29
2.5.1 Exemple……. 29
2.5.2 Condition de Mercer…….. 30
2.5.3 Exemples de noyaux…….. 30
2.6 Marges souples…… 31
2.7 SVM multi-classes…….. 33
2.7.1 Approche Une-contre-Tous (Une-contre-Reste) …….. 33
2.7.2 Approche Une-contre-Une……. 38
2.7.3 SVM floue……. 41
2.7.4 Graphe de décision acyclique orienté (DDAG) …… 43
2.7.5 Graphe acyclique orienté adaptatif (ADAG) …… 44
2.7.6 Graphe acyclique orienté adaptatif réordonné (RADAG) …… 47
2.8 Avantages et inconvénients des SVM……. 48
2.8.1 Inconvénients…….. 48
2.8.2 Avantages……. 48
2.9 Conclusion……... 49

Chapitre III : Sélection de modèles de machines à vecteurs de support

3.1 Introduction…… 51
3.2 Métriques de performances…… 51
3.2.1 Taux de reconnaissance…… 51
3.2.2 Matrice de confusion…… 52
3.3 Les méthodes de sélection de modèle SVM…… 53
3.3.1 Méthode HoldOut…….. 54
3.3.2 Validation croisée…….. 55
3.3.3 Leave-one-out…… 56
3.3.4 Bootstrap……... 56
3.3.5 Grille de recherche…… 57
3.3.6 Bornes sur l'erreur de généralisation…… 57
3.4 Conclusion……... 58

Chapitre IV : Optimisation avec méta-heuristiques

4.1 Introduction…… 60
4.2 Recuit simulé……..... 61
4.3 Algorithmes génétiques…….. 63
4.4 Recherche avec tabous…… 64
4.4.1 Concepts de base…… 64
4.4.2 Principe de la méthode…….. 65
4.4.3 Critère d'arrêt…… 66
4.4.4 Critère d'aspiration…… 66
4.4.5 Stratégie d'intensification……. 66
4.4.6 Stratégie de diversification…… 66
4.4.7 Stratégie de détermination de la taille de la liste tabou……. 66
4.4.8 Avantages et inconvénients……... 67
4.4.9 Adaptation de la recherche Tabou à notre problème……. 67
4.5 Optimisation par essaim particulaire…….. 69
4.6 Adaptation de PSO à notre problème…….. 71
4.7 Conclusion……... 73

Chapitre V : Implémentation et expérimentations

5.1 Introduction…… 75
5.2 SVM^{light}…….. 75
5.3 Description de notre système de reconnaissance…… 76
5.3.1 Notre base de données……... 76
5.3.2 Normalisation……. 77
5.3.3 Extraction des primitives…… 77
5.4 Expérimentations et Résultats……. 79
5.4.1 Expérimentation 1…….. 79
5.4.2 Expérimentation 2…….. 82
5.4.3 Expérimentation 3…….. 84
5.5 Conclusion……... 87

Chapitre VI : Contribution à la segmentation de l'écriture arabe manuscrite

6.1 Introduction……. 89
6.2 Approche proposée…….. 89
6.2.1 Histogramme de projection verticale…….. 89
6.2.2 Diviser le mot en parties (sous-mots) ……... 90
6.2.3 Algorithme d'extraction des points de segmentation……. 90
6.2.4 Amélioration de la position des points de segmentation……... 91
6.2.5 Détection de la ligne de base……. 92
6.2.6 Nombre de transitions…… 92
6.3 Résultats expérimentaux…… 93
6.4 Conclusion……... 96
Conclusion générale ……. 97
Bibliographie……. 99
Production scientifique……..…… 107

Liste des figures

Figure 1.1 : Cursivité de l'écriture arabe……. 06
Figure 1.2 : Exemple de pseudo-mots constituant des mots arabes. ……..……. 07
Figure 1.3 : Ligatures verticales. ……..……..……..……. 08
Figure 1.4 : Présence des chevauchements verticaux. ……..……..…….. 09
Figure 1.5 : Schéma général du système de reconnaissance des caractères...……. 09
Figure 1.6 : Effets de certaines opérations de prétraitement. ……..…… 10
Figure 1.7 : Image traitée par érosion. ……..……..……..…… 11
Figure 1.8 : Image traitée par dilatation. ……..……..……..……. 11
Figure 1.9 : Image traitée par Ouverture ……..……..……..……. 12
Figure 1.10 : Image traitée par Fermeture. ……..……..……... 12
Figure 1.11 : Squelettisation d'une Image. ……..……..……..…….. 13
Figure 2.1 : Arbre de classification des méthodes d'apprentissage à base de noyaux…… 21
Figure 2.2 : Séparation de deux ensembles de points par un Hyperplan H……. 23
Figure 2.3 : Hyperplan optimal, marge et vecteurs de support…… 24
Figure 2.4 : linéarité et non linéarité…… 25
Figure 2.5 : Transformation de l'espace de représentation et l'hyperplan séparateur dans le cas non linéairement séparables. ……. 25
Figure 2.6 : Transformation linéaire des données en une séparation linéaire dans un nouvel espace. ……. 29
Figure 2.7 : Hyperplan de séparation optimale avec marge souple dans un cas non linéairement séparable. ……... 31
Figure 2.8 : Association de classes aux hyperplans. …….. 34
Figure 2.9 : Représentation de la région d'ambiguïté (espace hachuré). …… 36
Figure 2.10 : Règle de décision continue : bissectrices des secteurs d'ambiguïté formant la nouvelle frontière de classification. …… 36
Figure 2.11 : Architecture du système en stratégie Un-contre-Tous……... 37
Figure 2.12 : Réduction de région d'ambiguïté hachurée pour l'approche une-contre-une. …….. 39
Figure 2.13 : Architecture du système en stratégie Une-contre-Une……... 40
Figure 2.14 : Courbe de niveau définie par frontière pour la fonction $M_k(x)$ d'adhésion à la classek. Fonction vaut 1 sur tout point de la zone hachurée. ……... 42

Figure 2.15 : Résolution de la région d'ambiguïté par les FSVMs. ……... 42
Figure 2.16 : Graphe de décision acyclique orienté à trois classes. ……... 43
Figure 2.17 : Favorisation de la feuille du milieu pour DDAG pour l'affectant la région d'ambiguïté. …… 44
Figure 2.18 : ADAG à huit classes. …… 45
Figure 2.19 : Illustration par listes d'un ADAG à sept classes…… 46
Figure 2.20 : Différentes étapes du RADAG. ……. 47
Figure 3.1 : Compromis biais-variance……... 54
Figure 4.1 : Principe des opérateurs génétiques dans le cas d'individu codé sur 7 bits….. 63
Figure 4.2 : Stratégie de déplacement des particules……... 70
Figure 5.1 : Echantillons de notre base de données…… …….. 77
Figure 5.2 : Extraction de caractéristiques de la lettre « jim »…… 77
Figure 5.3 : Résultats de la méthode grille de recherche pour le noyau RBF dans l'approche une-contre-une…… 80
Figure 5.4 : Résultats de la méthode grille de recherche pour le noyau Laplacien dans l'approche une-contre-une…… 80
Figure 5.5 : Résultats de la méthode grille de recherche pour le noyau RBF dans l'approche une-contre-tous…….. 81
Figure 5.6 : Résultats de la méthode grille de recherche pour le noyau Laplacien dans l'approche une-contre-tous. ……. 81
Figure 5.7 : Durée en secondes de chaque modèle SVM…… 82
Figure 5.8 : Temps de calcul en secondes de chaque noyau dans l'approche une-contre-reste……. 83
Figure 5.9 : Temps de calcul en secondes de chaque noyau dans l'approche une-contre-une…… 83
Figure 5.10 : Comparaison des résultats des différents noyaux pour l'approche une-contre-une…… 85
Figure 5.11 : Durée en secondes de chaque noyau pour l'approche une-contre-une…….. 85
Figure 5.12 : Comparaison des résultats des différents noyaux pour l'approche une-contre-reste……..... 86
Figure 5.13 : Durée en secondes de chaque noyau pour l'approche une-contre-reste…… 86
Figure 6.1 : Division du mot en parties (sous-mots)…….. 90
Figure 6.2 : Exemple de vecteur de projection verticale avec Tb = 9 et Tp = 5. …… 90

Figure 6.3 : Exemple d'extraction des points de segmentation. ……. 91
Figure 6.4 : Amélioration des points de segmentation…… 92
Figure 6.5 : Suppression du point de segmentation…… 92
Figure 6.6 : Exemple de nombre de transitions supérieur à deux. …… 93
Figure 6.7 : Résultats de segmentation avec le seuil S fixé à 50……. 94
Figure 6.8 : Exemple de sur-segmentation…….. 94
Figure 6.9 : Résultats de la segmentation avec Tb = 20 et Tp = 15. ……..... 95
Figure 6.10 : Exemple de sous-segmentation……. 95

Liste des tableaux

Tableau 1.1 : Lettres arabes avec leurs positions dans le mot…….. 6
Tableau 1.2 : Lettres arabes ayant des points diacritiques…… 8
Tableau 3.1 : Matrice de confusion pour la classification binaire…… 52
Tableau 5.1 : Lettres arabes utilisées avec ses différentes formes (96 classes)……. 76
Tableau 5.2 : Résultats obtenus pour chaque noyau dans l'approche une-contre-reste… 82
Tableau 5.3 : Résultats obtenus pour chaque noyau dans l'approche une-contre-une…. 83
Tableau 5.4 : Meilleurs résultats obtenus pour chaque noyau pour l'approche une-contre-une…….. 85
Tableau 5.5 : Meilleurs résultats obtenus pour chaque noyau pour l'approche une-contre-reste…… 86

Introduction générale

L'écriture manuscrite bien que progressivement remplacée par des applications numériques reste cependant extrêmement présente dans notre monde. Nous continuons d'écrire nos chèques, nos adresses postales, nos cartes postales, nous remplissons nos formulaires, nous posons nos idées sur des feuilles de brouillon. Aujourd'hui, malgré l'avènement des nouvelles technologies, l'écriture reste un moyen de communication incontournable.

La reconnaissance de l'écriture manuscrite demeure un sujet de recherche vivace et vaste. Elle a connu ces dernières années de grands progrès, et les succès des travaux de recherches ont donné lieu à de nombreuses applications industrielles, notamment dans le domaine de la lecture automatique de formulaires, de chèques ou d'adresses postales.

Deux types distincts de reconnaissance d'écriture existent : reconnaissance hors-ligne et reconnaissance en-ligne. Le premier, consiste à déterminer les lettres ou les mots présents dans une image numérique d'un document manuscrit. Le second est relativement plus simple car il peut utiliser des informations supplémentaires non disponibles pour les systèmes hors ligne tels que l'information temporelle comme la pression et l'ordre séquentiel de l'écriture. En outre, le cas hors ligne est celui qui correspond à la tâche de lecture classique effectuée par les humains.

Les travaux de recherches sur la reconnaissance de 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) basées sur la théorie de l'apprentissage

statistique [1]. Les SVM introduites au début des années 90, réalisent un grand succès dans la théorie de l'apprentissage statistique. 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.

La difficulté majeure liée à l'utilisation de classificateurs SVM est la nécessité d'ajuster les variables conditionnant le processus d'apprentissage. Ces variables sont appelées hyper-paramètres. Dans le but de choisir adéquatement ces hyper-paramètres, une sélection manuelle des valeurs des hyper-paramètres peut être entreprise. Dans ce cas, il est important de pouvoir définir au préalable une zone d'intérêt dans laquelle plusieurs valeurs seront testées. Sans aucune information a priori, celle-ci n'est pas facile à trouver. Aussi, considérer une grande zone d'intérêt est très prohibitif en temps de calcul.

En outre, il est nécessaire d'échantillonner cet intervalle de façon à assurer la plus grande résolution. Cette procédure devient exagérément complexe dès lors qu'il existe plus qu'un hyper-paramètre à considérer. Il est donc très important de disposer d'une procédure automatique de sélection de modèle qui s'affranchit de la sélection manuelle des valeurs des hyper-paramètres.

Dans cette thèse nous avons adapté des techniques méta-heuristiques pour la sélection automatique de modèle SVM, une étude comparative a été effectuée dans laquelle nous discutons l'effet des valeurs des hyper-paramètres sur le taux de prédiction et le temps de calcul. Nous avons développé un système complet de reconnaissance des caractères arabes manuscrits hors-ligne et la création d'une base contenant 4840 images de caractères dans leurs différentes positions (début, milieu, fin et isolé) et nous avons contribué aussi par le développement d'une approche permettant la segmentation d'un mot en caractères.

Cette thèse est constituée de six chapitres. Le premier chapitre sera consacré essentiellement à la description de la problématique associée à la reconnaissance de l'écriture arabe manuscrite. Nous mettrons l'accent sur les caractéristiques de l'écriture arabe, et nous présenterons l'architecture générale d'un système de reconnaissance d'écriture manuscrite.

Nous présentons dans le deuxième chapitre un état de l'art sur les machines à vecteur de support en général avec une synthèse des différentes approches SVM multi-classes couramment utilisées.

Le troisième chapitre présente une revue de la littérature des principales méthodes de sélection de modèle, précédées par les techniques d'évaluation des modèles de décision obtenus par la méthode SVM.

Dans le quatrième chapitre, nous rappelons les fondements essentiels des méthodes méta-heuristiques et nous présentons notre adaptation des méthodes recherche avec tabous et optimisation par essaim de particules au problème de sélection de paramètres de modèle SVM.

Nous présentons dans le chapitre cinq une description de notre système de reconnaissance de caractères Arabes manuscrits, ainsi qu'une discussion et évaluation des différents résultats obtenus. Dans le dernier chapitre nous présentons notre contribution dans la segmentation de l'écriture Arabe manuscrite.

Chapitre I

Reconnaissance de l'écriture arabe manuscrite

1.1 Introduction

La reconnaissance de l'écriture a eu un grand essor de la part des chercheurs dès les années1970; un grand nombre de travaux ont été effectués et le taux de reconnaissance s'est progressivement amélioré. La reconnaissance de l'écriture renferme aussi bien la reconnaissance de caractères imprimés que manuscrits. Si la reconnaissance de caractères imprimés est plus ou moins maîtrisée et donne de bons résultats, en revanche la reconnaissance des caractères manuscrits s'avère difficile et demeure un plein axe de recherche.

Contrairement au Latin, la reconnaissance de l'écriture arabe manuscrite reste encore aujourd'hui au niveau de la recherche et de l'expérimentation. Cependant et depuis quelques années elle a pris un nouvel essor et fait l'objet d'applications de plus en plus nombreuses. Parmi ces applications, nous citons le traitement automatique des dossiers administratifs, des formulaires d'enquêtes, des chèques bancaires, numérisation et sauvegarde du patrimoine culturel manuscrit, l'indexation de documents et la recherche d'informations dans les bibliothèques numériques, l'identification du scripteur dans le domaine biométrie…etc. Dans ce chapitre nous présenterons les caractéristiques de l'écriture arabe ainsi que les différentes étapes constituant un système de reconnaissance d'écriture manuscrite hors-ligne.

1.2 Caractéristiques de l'écriture Arabe

L'écriture arabe a été développée à partir d'un type d'Araméen. La langue araméenne comporte moins de consonants que l'arabe, alors de nouvelles lettres ont été créées en ajoutant des points aux lettres déjà existantes. D'autres petites marques appelées diacritiques sont utilisées pour indiquer de courtes voyelles, mais elles ne sont généralement pas utilisées [2], [3]. Elle possède les caractéristiques suivantes [4] :

- Contrairement aux écritures occidentales qui s'écrivent de gauche-à-droite, l'arabe s'écrit de droite-à-gauche.
- Il n'y a pas de différence entre les lettres manuscrites et les lettres imprimées. Les notions de lettre capitale et lettre minuscule n'existent pas, l'écriture est donc monocamérale.

- L'alphabet Arabe est composé de 28 lettres dont la forme change selon la position dans le mot. Les lettres s'écrivent différemment selon qu'elles sont isolées, au début, au milieu ou à la fin du mot (voir Tableau 1.1). [5].

Tableau 1.1 : Lettres arabes avec leurs positions dans le mot

lettre	nom	fin	milieu	début	lettre	nom	fin	milieu	début
ا	alif	ـا	ـا	ا	ط	Ṭa	ـط	ـطـ	طـ
ب	ba	ـب	ـبـ	بـ	ظ	Ẓa	ـظ	ـظـ	ظـ
ت	ta	ـت	ـتـ	تـ	ع	ɛayn	ـع	ـعـ	عـ
ث	tha	ـث	ـثـ	ثـ	غ	ghayn	ـغ	ـغـ	غـ
ج	jim	ـج	ـجـ	جـ	ف	fa	ـف	ـفـ	فـ
ح	Ḥa	ـح	ـحـ	حـ	ق	qaf	ـق	ـقـ	قـ
خ	kha	ـخ	ـخـ	خـ	ك	kaf	ـك	ـكـ	كـ
د	dal	ـد	ـد	د	ل	lam	ـل	ـلـ	لـ
ذ	dhal	ـذ	ـذ	ذ	م	mim	ـم	ـمـ	مـ
ر	ra	ـر	ـر	ر	ن	nun	ـن	ـنـ	نـ
ز	zay	ـز	ـز	ز	ه	ha	ـه	ـهـ	هـ
س	sin	ـس	ـسـ	سـ	و	waw	ـو	ـو	ـو
ش	shin	ـش	ـشـ	شـ	ي	ya	ـي	ـيـ	يـ
ص	Ṣad	ـص	ـصـ	صـ	ء	hamza	أ ؤ إ ئ		
ض	Ḍad	ـض	ـضـ	ضـ					

- La plupart des lettres s'attachent entre elles, même en imprimé, ce qui offre à l'écriture arabe la caractéristique de cursivité [6], la Figure 1.1 illustre un exemple tiré du sacré Coran.

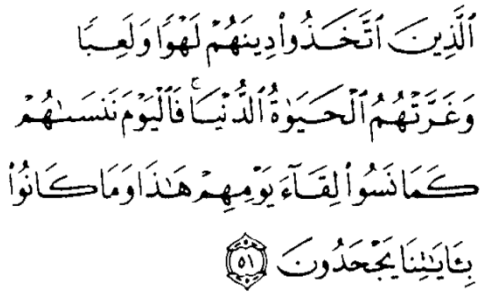


Figure 1.1 : Cursivité de l'écriture arabe.

- Les caractères arabes ne possédant pas une taille fixe (hauteur et largeur), leur taille varie d'un caractère à un autre et d'une forme à une autre à l'intérieur d'un même caractère.

- Comme on a indiqué précédemment six lettres ne s'attachent jamais à la lettre suivante : "ذ" ,"د" ,"ر" ,"ز","أ", et "و", de ce fait, un mot unique peut être entrecoupé d'un ou plusieurs espaces donnant plusieurs pseudo-mots ou composantes connexes ou également sous-mots [7], ce qui est le cas du mot représenté dans la Figure 1.2. Un pseudo-mot est une unité connexe regroupant une ou plusieurs lettres sous forme d'une séquence. En manuscrit, l'espacement entre les différents pseudo-mots d'un même mot n'est pas forcément systématiquement supérieur à l'espacement entre deux mots différents, ce qui pose parfois des problèmes de segmentation.

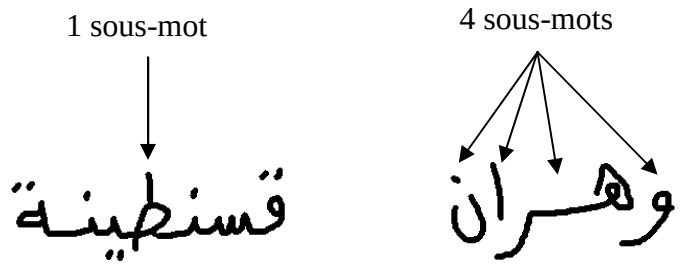


Figure 1.2 : Exemple de pseudo-mots constituant des mots arabes.

- La plupart des caractères sont composés de boucles et de courbes, souvent tracées dans le sens horaire.
- La bande centrale est généralement la plus chargée au point de vue densité d'informations en pixels [8]. Elle correspond à l'emplacement des ligatures horizontales, aux caractères centrés et aux boucles, on l'appelle ligne de base.
- Dans l'alphabet arabe, 15 lettres parmi les 28 possèdent un ou plusieurs points. Ces points diacritiques sont situés soit au-dessus, soit en dessous de la forme à laquelle ils sont associés, mais jamais les deux à la fois. Le nombre maximal de points que peut avoir une lettre est de trois points au-dessus du caractère, ou deux points en dessous. Ces points permettent de différencier la prononciation des lettres arabes. Le Tableau 1.2 présente les lettres ayant des points diacritiques ainsi que leurs nombres et positions.

Tableau 1.2 : Lettres arabes ayant des points diacritiques.

Position / Nombre de points	Au dessous	Au dessus
Un point diacritique	خ ز ض ظ غ ف ن	ب ج
Deux points diacritiques	ت ق	ي
Trois points diacritiques	ث ش	

- En écriture arabe, il n'y a pas de liaisons hautes comme le 'v' ou le 'o' en latin : les ligatures se situent au niveau de la ligne support de l'écriture qui est la ligne de base. En revanche, les scripteurs sont libres de constituer certains groupes de deux ou trois lettres liées verticalement en début de pseudo-mot. Ce sont les ligatures verticales qui sont utilisées pour des raisons d'esthétique (voir la Figure 1.3). Ces symboles peuvent être reconnus tels qu'ils sont. Les ligatures sont donc des combinaisons de deux lettres ou plus.

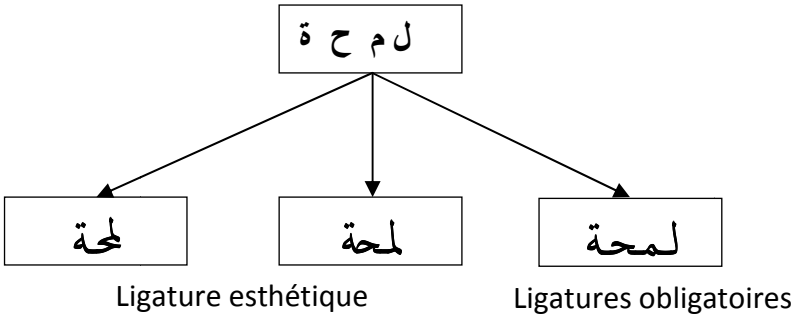


Figure 1.3 : Ligatures verticales

- La présence de chevauchements verticaux qui peuvent se produire par l'intersection des composantes connexes (pseudo-mots), ou des mots comme montre l'exemple de la Figure 1.4.

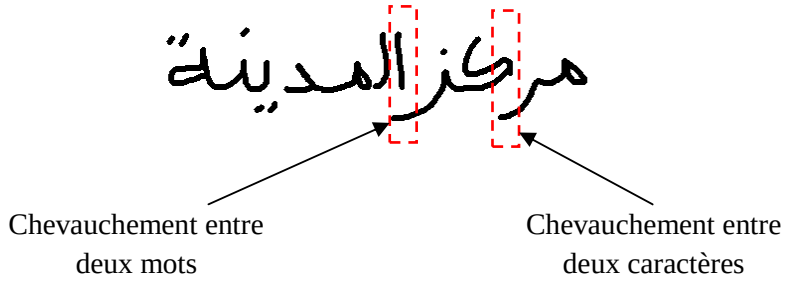


Figure 1.4 : Présence des chevauchements verticaux.

1.3 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.

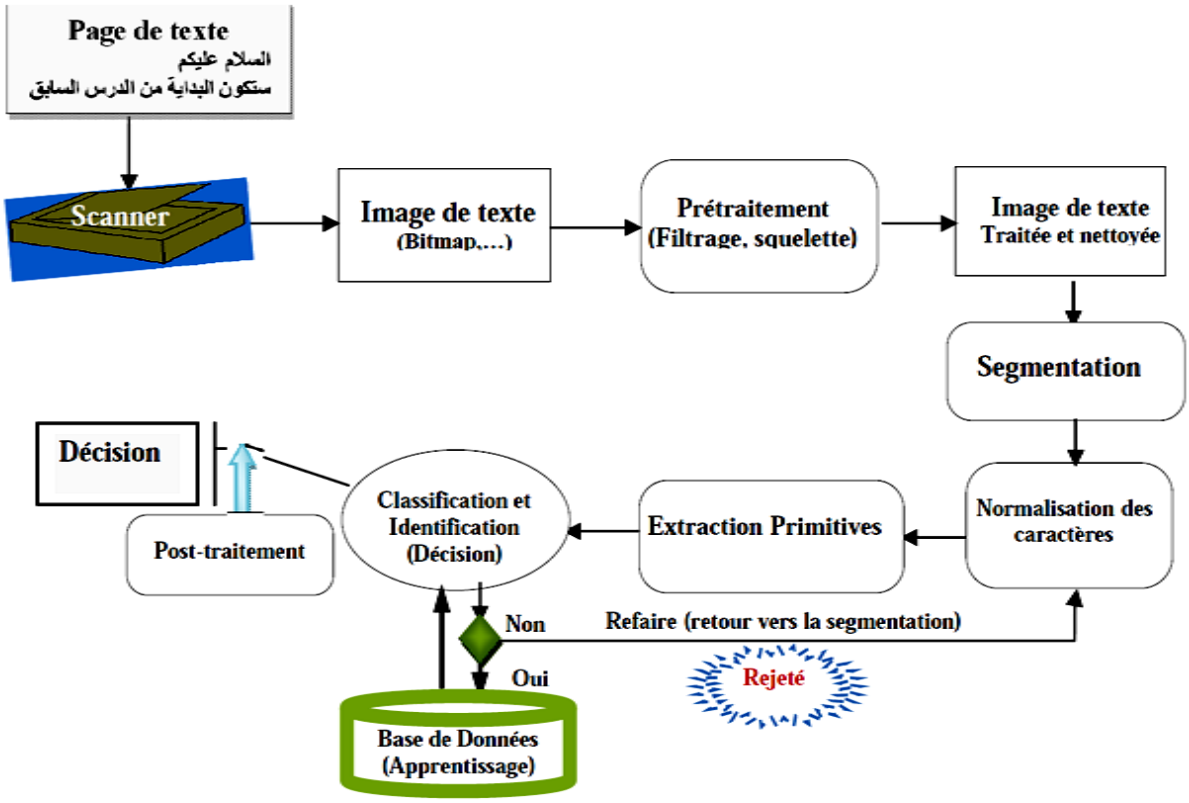


Figure 1.5 : Schéma général du système de reconnaissance des caractères hors-ligne [9].

1.3.1 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. Elle consiste en deux phases :

- Echantillonnage (numérisation) 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)) [10].

1.3.2 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 d'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).

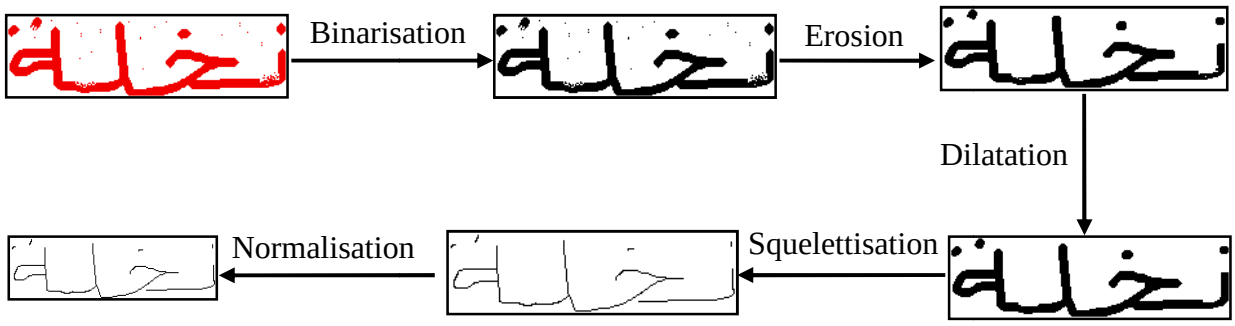


Figure 1.6 : Effets de certaines opérations de prétraitement.

1.3.2.1 Binarisation

La binarisation 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 développées 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 :

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

$I_n(x, y)$ 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 attribue 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 attribue la valeur d'intensité minimale [11].

1.3.2.2 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) [11].

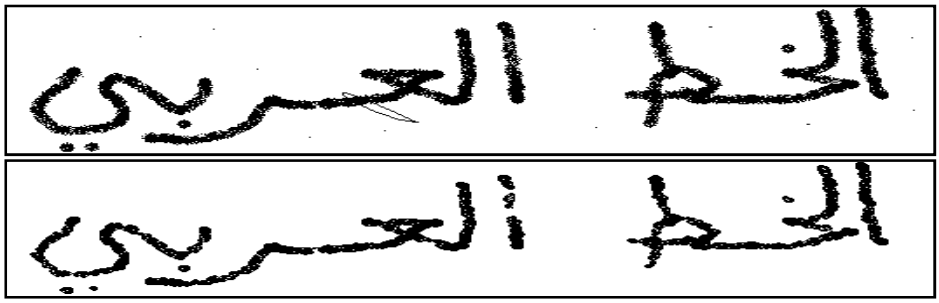


Image avant « érosion »

Image après « érosion »

Figure 1.7 : Image traitée par érosion

1.3.2.3 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 ($P_{voisin}(x,y)=0$), alors on affecte à ce pixel la couleur noir (p(x,y)=0).

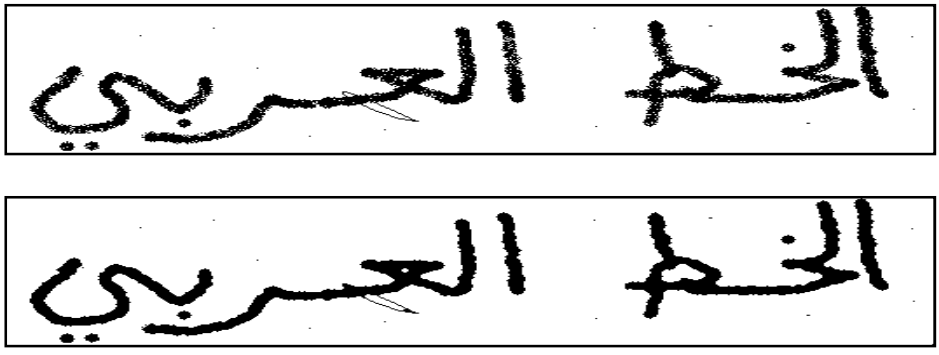


Image avant «Dilatation»

Image après «dilatation»

Figure 1.8 : Image traitée par dilatation

1.3.2.4 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 et simplifier les formes en lissant les bosses tout en conservant l'allure globale [11].

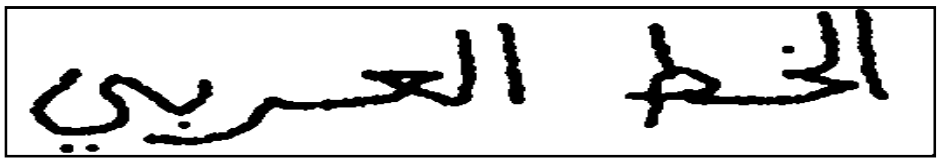


Figure 1.9 : Image traitée par Ouverture

1.3.2.5 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.

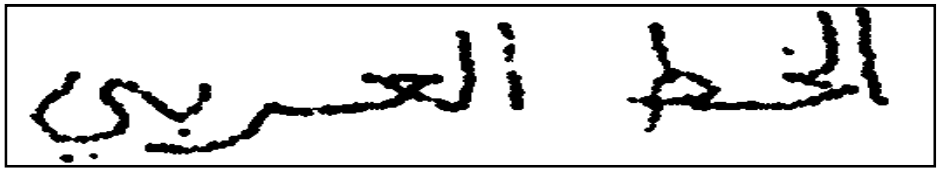


Figure 1.10 : Image traitée par Fermeture

1.3.2.6 Squelettisation d'une image

La procédure de squelettisation s'effectue sur une image binaire, et a pour but de réduire l'épaisseur du tracé d'un mot à un pixel, tout en conservant la continuité de celui-ci. Le principe de cette procédure est d'effectuer une succession d'opérations d'érosion conditionnelle, jusqu'à ce que le but recherché soit atteint [12].

Il n'existe pas de définition unique du squelette. Le squelette doit seulement remplir trois conditions [13] :

– Il doit être aussi fin que possible (typiquement, 1 pixel d'épaisseur)

– Il doit respecter la connexité

– Il doit être centré dans la forme qu'il représente

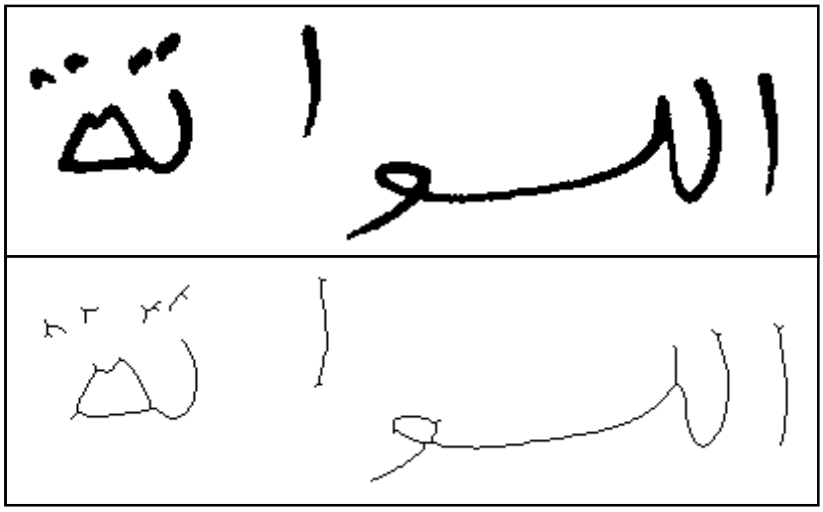


Figure 1.11 : La squelettisation d'une Image [9].

1.3.2.7 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 éliminés à la suite de la normalisation, ce qui peut entraîner à des confusions entre certains caractères [14].

Une application doit être définie pour effectuer cette tâche. 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 :

$$\begin{cases} G(x,y) = \text{noir} & \text{si pixel } (\mu_1 x, \mu_2 y) = \text{noir} \\ G(x,y) = \text{blanc} & \text{sinon} \end{cases}$$

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)

1.3.3 Phase de segmentation

En reconnaissance de l'écriture manuscrite, les données à traiter sont des images. La mise en œuvre d'une étape de segmentation permet de diviser l'image en différentes imagettes de taille moins importantes qui peuvent être des graphèmes, des lettres ou bien des sous-mots. Cependant une imagette reste une matrice de pixels [4]. Il existe deux techniques permettant la mise en œuvre de la segmentation :

1.3.3.1 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 qui correspondent à ses classes de lettres [15]. 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.

1.3.3.2 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 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 [15].

1.3.4 Phase d'extraction des caractéristiques (primitives)

C'est l'une des étapes les plus délicates et les plus importantes en OCR (Optical Character Recognition). L'extraction de primitives consiste à transformer une image (caractère, graphème, bande verticale, ...) en un vecteur de primitives de taille fixe. Cette transformation revient à changer l'espace de représentation des données, du plan de l'image vers un espace à N dimensions [13]. Les primitives doivent être extraites à base des informations significatives et appropriées afin de minimiser la variabilité entre les exemples de la même classe, et de maximiser la variabilité entre les exemples des classes [16]. Le choix de ces primitives est critique, et influence nettement le résultat. Quatre groupes de caractéristiques peuvent être distingués :

- caractéristiques topologiques,
- caractéristiques structurelles,
- caractéristiques statistiques,
- superposition des modèles et corrélation.

1.3.4.1 Caractéristiques topologiques

Dans ce type de primitives, on compte les profils et histogrammes. Les caractéristiques peuvent être définies par le nombre des trous, l'évaluation des concavités, mesures des pentes et autres paramètres de courbures et évaluer des orientations, mesurer la longueur et l'épaisseur des traits, détecter les croisements et les jonctions des traits, mesures les surfaces et les périmètres, etc. [17].

1.3.4.2 Caractéristiques structurelles

Ces caractéristiques sont extraites à partir d'une image contenant le squelette ou le contour et non pas de l'image brute. Dans ce cas, on s'intéresse à décrire la forme du squelette ou du contour. Cette définition intègre [17] :

- 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 symboles diacritiques 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'elles soient extraites d'une courbe, d'un trait ou d'un segment de contour.

1.3.4.3 Caractéristiques statistiques

Elles décrivent la forme d'un texte en utilisant des fonctions (ex : transformée de Fourier et transformée de Hough) statistiques. Parmi les caractéristiques utilisées dans la reconnaissance du manuscrit arabe on trouve [2] :

- Le zonage (zoning), consiste à superposer une grille n × m sur l'image du caractère et pour chacune des régions résultantes, calculer la moyenne ou le pourcentage de points en niveaux de gris, donnant ainsi un vecteur de taille n×m de caractéristiques.
- caractéristiques de lieu géométrique : en utilisant la méthode Loci qui 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.
- La méthode des moments : les moments d'une forme par rapport à son centre de gravité sont invariants par rapport à la translation et peuvent être invariants par rapport à la rotation. Ils sont aussi indépendants de l'échelle. Ces caractéristiques peuvent être facilement et rapidement extraites d'une image de texte, ils peuvent tolérer modérément les bruits et les variations.

1.3.4.4 Superposition des modèles 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 [2].

1.3.5 Phase de classification

La phase de classification permet d'attribuer un label ou étiquette à chaque vecteur de caractéristique obtenu dans la phase précédente. Elle travaille selon deux principaux modes : le premier est l'apprentissage et le second est la reconnaissance et décision [18].

1.3.5.1Apprentissage

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'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.

L'apprentissage est dit supervisé quand le concepteur indique l'étiquette de la forme en entrée. Par contre, lorsque le processus est automatisé, on dit que l'apprentissage est non supervisé [19].

1.3.5.2 Reconnaissance et décision

Dans cette étape, le système attribue un label ou étiquette à des nouveaux exemples en se basant sur la base des modèles références du langage créé durant l'étape précédente. La réponse du système peut être unique dans le cas où un seul modèle répond à la description de la forme en question, comme elle peut être multiple ce qui conduit à une confusion. En plus, elle peut être un rejet s'il n'y a pas de modèle qui sont proches du modèle d'entrée [18].

Plusieurs approches ont été développées dans les systèmes de la reconnaissance de l'écriture manuscrite cursive afin de réaliser la tâche de classification. Parmi les méthodes les plus utilisées en classification on trouve [2] :

- Méthode bayésienne,
- Méthode du plus proche voisin,

- Réseaux de neurones,
- Machines à Vecteurs de Support (SVM).

a. Méthode Bayésienne

Dans cette méthode, l'étiquette choisie est celle qui possède une suite de primitives avec une plus forte probabilité à posteriori par rapport aux modèles préalablement appris [20].

b. Méthode du k plus proches voisins (KPPV)

La méthode du k plus proches voisins (En anglais KNN: K Nearest Neighbors) consiste à calculer une distance entre l'exemple en entrée et les exemples dans la base des modèles. Cette distance est obtenue en utilisant une fonction de mesure de similarité par exemple euclidien, gaussien, Manhattan,…etc. Ensuite, elle attribue à l'exemple en question l'étiquette des k modèles de référence les plus proches. Elle permet d'avoir de bons résultats, avec une simplicité de réalisation. Cependant, la vitesse de traitement s'affaiblie à des calculs de distances quand la taille de base des modèles devient importante [2], [20].

c. 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 [21], Rumelhart [22] et Lecun [23]. Ce classifieur a trouvé son 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.

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 [24].

d. Machines à Vecteurs de Support (SVM)

Ce classificateur consiste à trouver une frontière séparant deux classes de données, avec la possibilité d'enrichir le modèle par une projection du problème dans un autre espace dans

lequel les données peuvent être séparées facilement [25]. Nous allons présenter en détail cette méthode dans le chapitre II.

1.3.6 Phase de post traitement

Les post-traitements permettent de corriger des erreurs effectuées lors de la reconnaissance. On parle de post-traitement, mais de nombreux algorithmes dits de post-traitements sont intégrés dans les algorithmes de reconnaissance, ils agissent en tant que règles supplémentaires pour déterminer la solution.

Cette étape est nécessaire car l'écriture manuscrite est un processus qui engendre beaucoup de bruit lors de la reconnaissance. Elle permet de corriger des erreurs minimes qui n'ont pas de sens au vu de la langue traitée, comme par exemple corriger « stode » en « stade» en français. La suppression des ambiguïtés peut se faire au niveau caractère pour modéliser des mots ou au niveau mot pour modéliser des phrases. Ce besoin de correction est lié à des difficultés à appréhender le contexte par les algorithmes de reconnaissance [26].

1.4 Conclusion

Nous avons présenté dans ce chapitre le concept de reconnaissance optique des caractères et de l'écriture en générale, en décrivant les différentes phases de processus de reconnaissance de l'écriture manuscrite ainsi que les caractéristiques morphologique 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 particulier. Ce modèle sera revu en détail au chapitre suivant.

Chapitre II

Machines à vecteurs de support

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 [27]. 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 [28]. En 1995, Cortes et al. proposent une version régularisée du SVM qui tolère les erreurs d'apprentissage avec pénalités [29] [30].

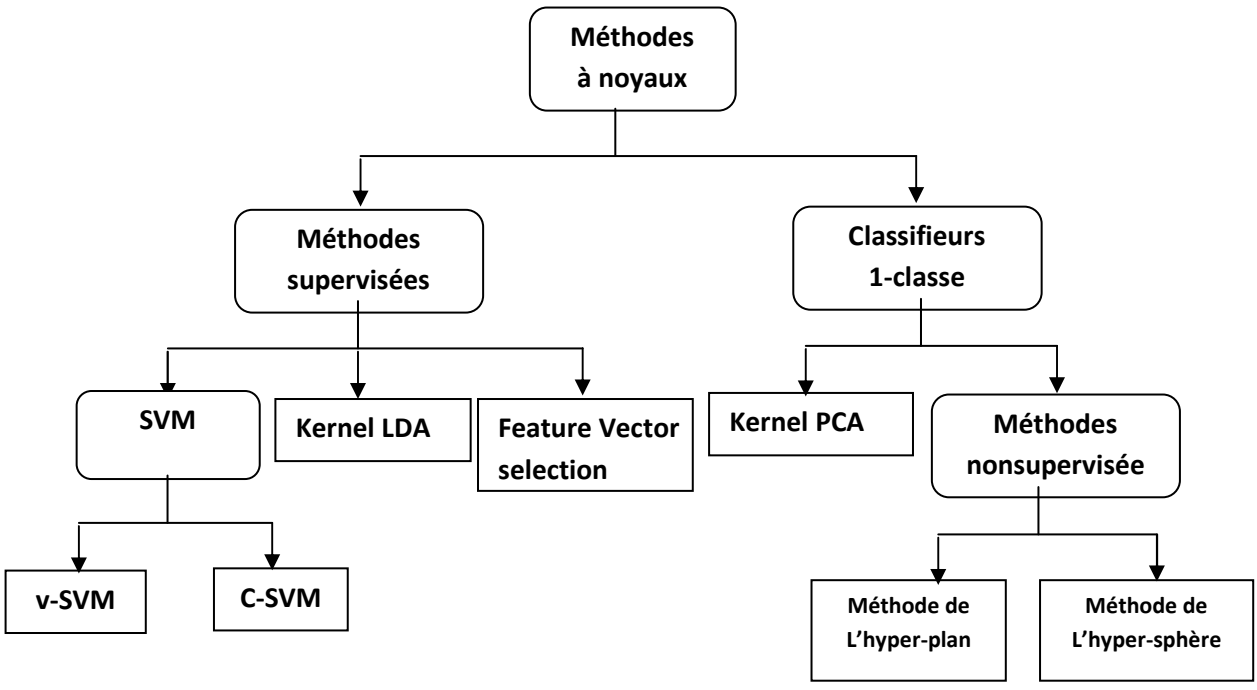


Figure 2.1 : Arbre de classification des méthodes d'apprentissage à base de noyaux [30]

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. 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 » [31]. Dans le schéma qui suit, on détermine un hyperplan qui sépare deux ensembles de points.

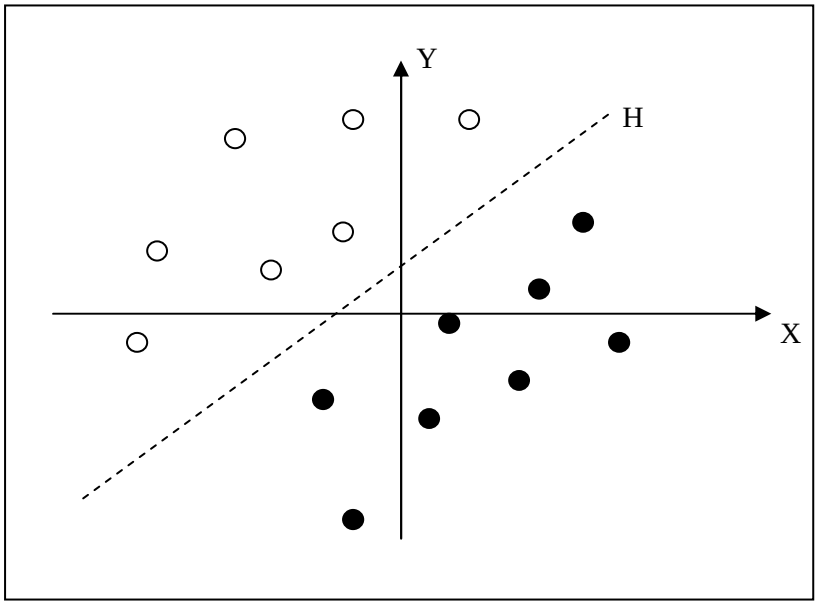


Figure 2.2 : Séparation de deux ensembles de points par un Hyperplan H

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'hyperplans valides mais la propriété remarquable des SVM est que l'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 [32]. Dans le schéma qui suit, on détermine un hyperplan qui sépare les deux ensembles de points.

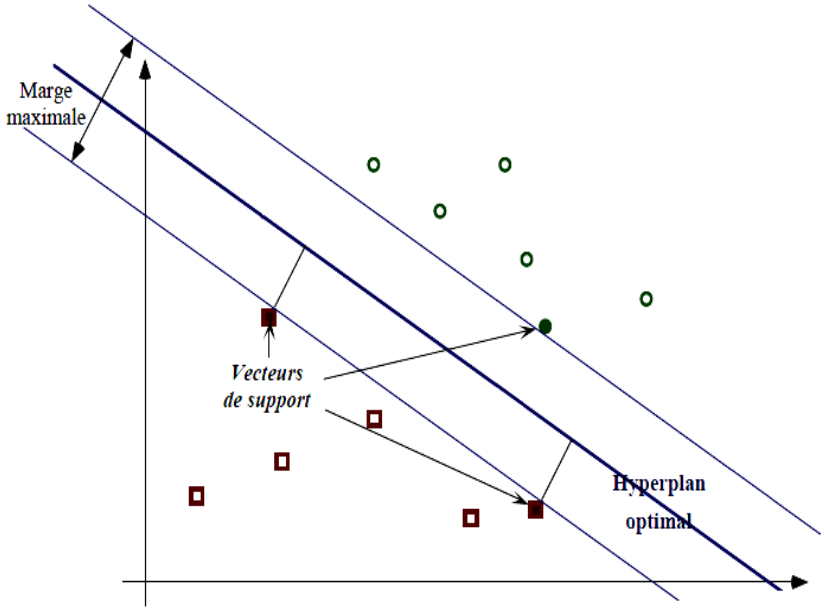


Figure 2.3 : Hyperplan optimal, marge et vecteurs de support

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 :

2.3.1.1Cas linéairement séparable

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

2.3.1.2Cas 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 [33] (Figure 2.4).

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 redescription» de façon à ce que les données deviennent linéairement séparables.

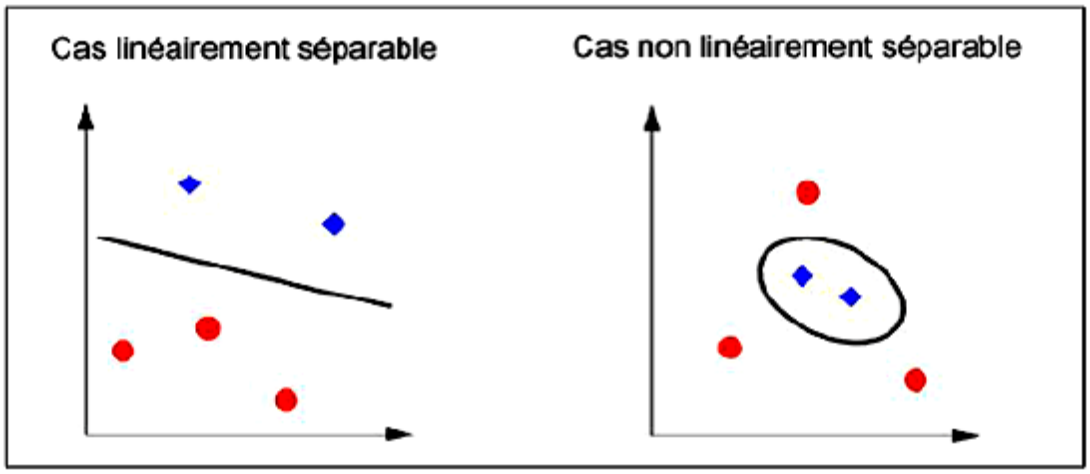


Figure 2.4 : Linéarité et non linéarité

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 description de plus grande dimension. Cette transformation est réalisée via une fonction noyau (Figure 2.5).

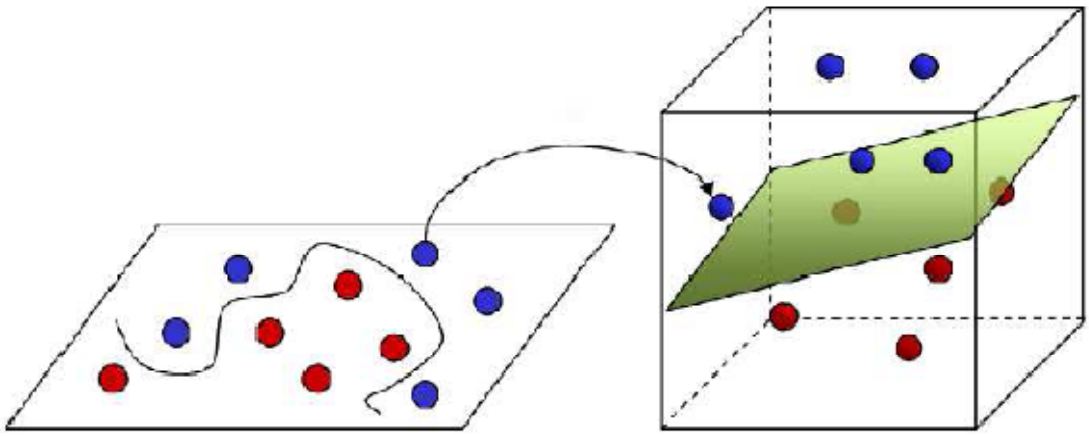


Figure 2.5 : Transformation de l'espace de représentation et l'hyperplan séparateur dans le cas non linéairement séparable.

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^dva faire correspondre une valeur de sortie $y \in \{-1, 1\}$:

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

Dans le cas linéaire, une fonction discriminante h est obtenue par combinaison linéaire d'unvecteur d'entrée $x = (x_1, \dots, x_d)$ 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 classifieurh 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 [1], 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 $L(w, b, \alpha)$. Le lagrangien *L* doit être minimisé par rapport à *w* et *b* et maximisé par rapport àα. 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 $y_i (w . x_i + b) - 1 = 0$. 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'exemples. 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 résoudre ce problème il suffit de projeter les données dans un espace de dimension supérieure 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 2.6).

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 :

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

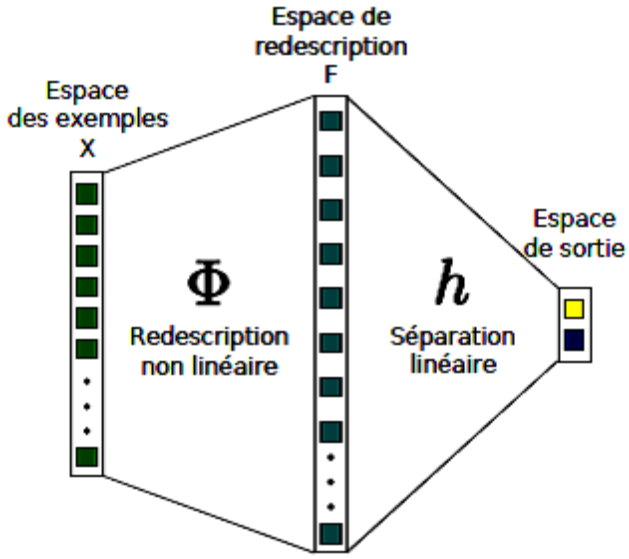


Figure 2.6 : Transformation linéaire des données en une séparation linéaire dans un nouvel espace.

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 noyau doit satisfaire la condition de Mercer présentée 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 [24]. Dans ce cas, on montre qu'il existe un espace H et une fonction Φ tels que :

$$k(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$$

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 [24].

2.5.3Exemples de noyaux

- Linéaire : $K(x_i, x_j) = x_i . x_j$
- Polynomial : $K(x_i, x_j) = (\gamma\, x_i . x_j + coef)^d$
- Gaussien (RBF) : $K(x_i, x_j) = exp^{-\gamma \|x_i - x_j\|^2}$
- Laplacien : $K(x_i, x_j) = exp^{-\sqrt{\gamma}\|x_i - x_j\|}$
- Sigmoïde : $K(x_i, x_j) = tanh(\gamma(x_i . x_j) + coef)$

Les noyaux gaussiens sont des noyaux dits de type radial (fonction à base radiale abrégée RBF [34]), indiquant qu'ils dépendent de la distance entre deux exemples. L'hyperplan séparateur se réécrit avec la fonction noyau sous la forme suivante :

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

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 sûr 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 côté de l'hyperplan comme le montre la Figure 2.7. Cortes et Vapnik en 1995 ont donc introduit le concept de marge souple [29]. Certains exemples d'apprentissage peuvent violer la contrainte (2.4) que l'on retrouve dans l'équation de la forme primale (2.5).

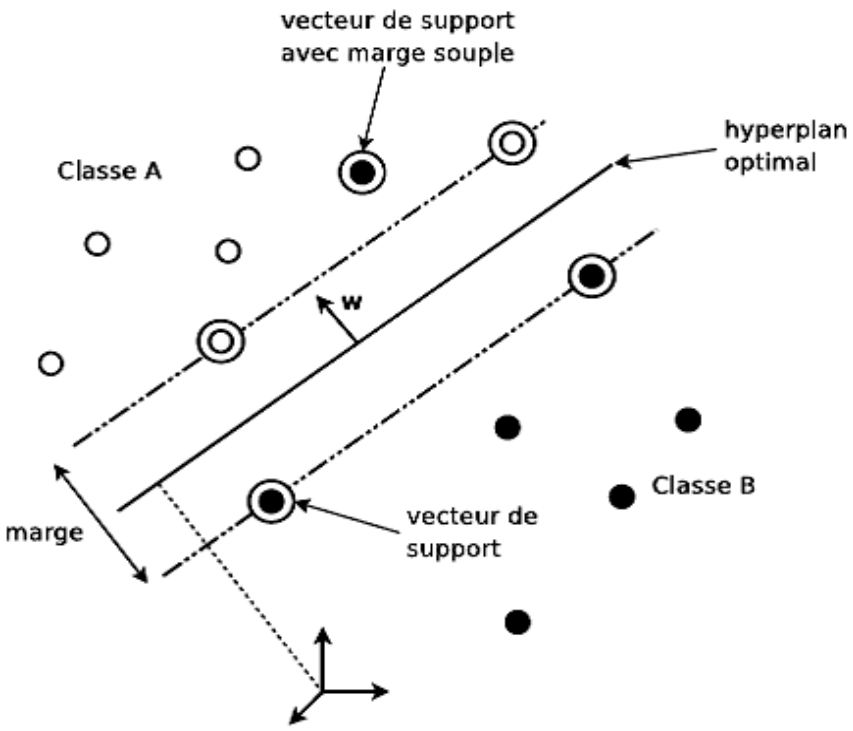


Figure 2.7 : Hyperplan de séparation optimale avec marge souple dans un cas non linéairement séparable.

On introduit par conséquent des variables dites « ressorts » $\xi = \xi_1, \dots, \xi_P$ permettant 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'expressiondu 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 parune 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 [35]. 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).

Dans les paragraphes précédents, on a souligné que la solution dépend uniquement d'un nombre restreint d'exemples 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 pour les cas à N classes 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 (Une-contre-Reste)

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 [1] 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 $Y = \{-1, +1\}$. Àchaque classe est associé un hyperplan défini par les fonctions de décision $f_k(x) = \langle w_k \cdot x \rangle + b_k, k = -1, +1$.

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

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

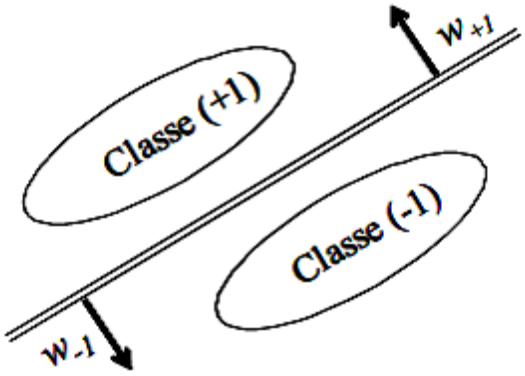


Figure 2.8 : Association de classes aux hyperplans.

H_{+1} est associé à la classe (+1) et H_{-1} est associé à la classe (−1). Une illustration graphiqueest 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 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 vecteurssupports 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'obtienten résolvant le problème,

$$Minimiser_{w^k,\xi^k,b} \frac{\|w_k\|^2}{2} + C\sum_{j=1}^{l}\xi_j^k, \tag{2.17}$$

$$sous \qquad \langle w_k.\Phi(x_j)\rangle + b_k \geq 1 - \xi_j^k, \qquad si\ \ y_j = k,$$

$$\langle w_k.\Phi(x_j)\rangle + b_k \leq -1 + \xi_j^k,\ \ si\ \ y_j \neq k,$$

$$\xi_j^k \geq 0, \qquad j = 1,2,\dots,l.$$

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 :

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

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 [36].

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.

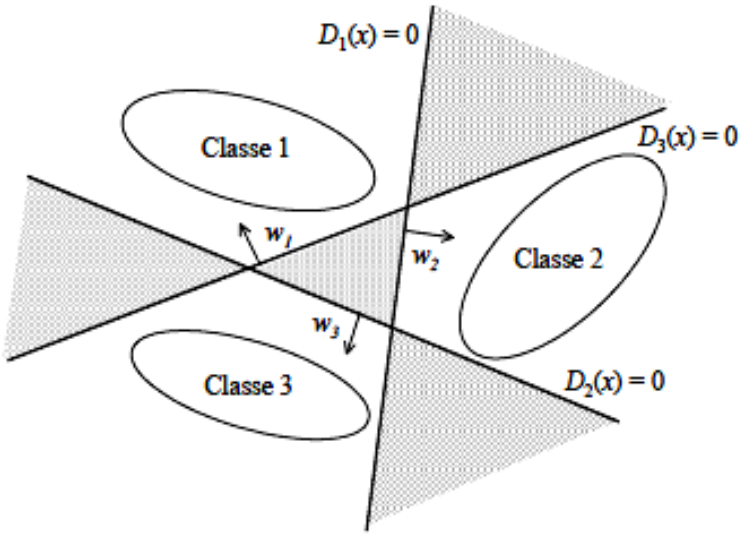


Figure 2.9 : Représentation de la région d'ambiguïté (espace hachuré).

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.

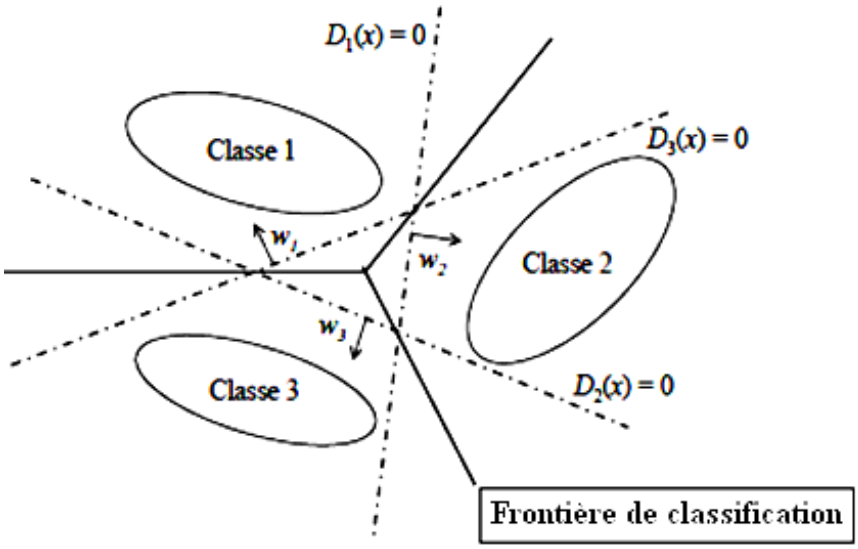


Figure 2.10 : Règle de décision continue : bissectrices des secteurs d'ambiguïté formant la nouvelle frontière de classification.

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 [37] le but de fournir une règle de décision au niveau de la région d'ambiguïté. Le même auteur aprouvé dans [38] 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 [36].

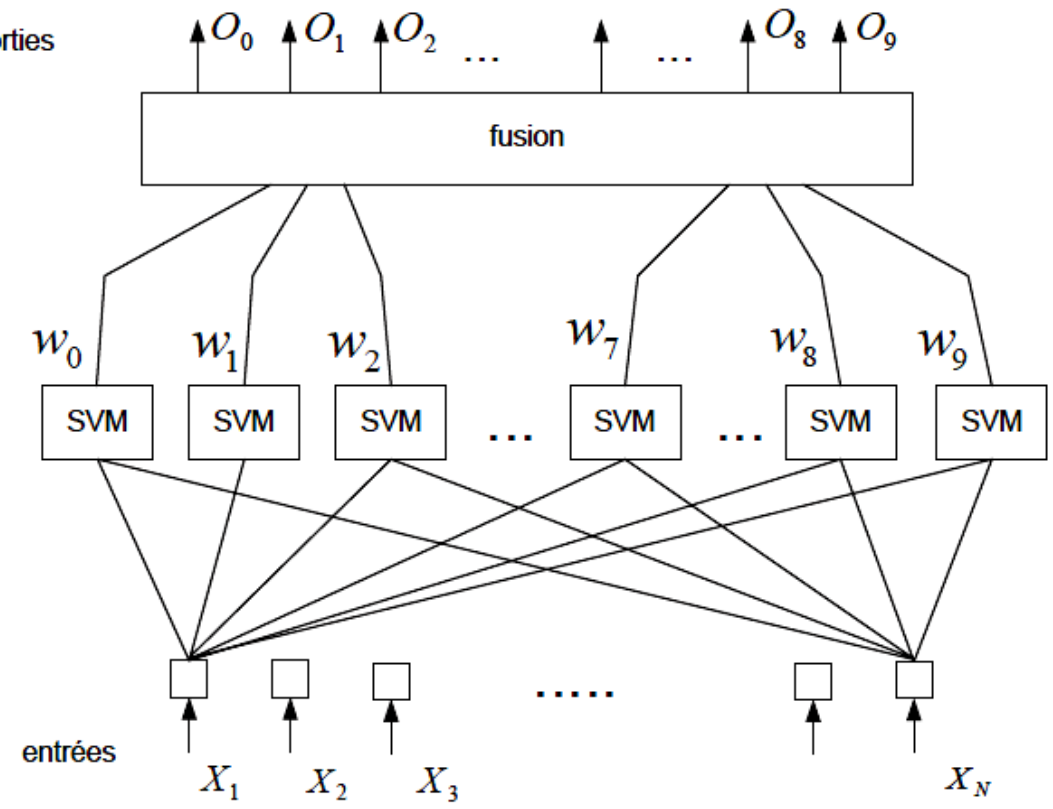


Figure 2.11 : Architecture du système en stratégie Un-contre-Tous [30]

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

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

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. [39] et utilisé pour la première fois dans le contexte des machines à vecteurs supports par Kreßel [40]. 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 :

$$Minimiser_{w^{ks},\xi^{ks},b} \frac{\|w^{ks}\|^2}{2} + C\sum_{i=1}^{l_{ks}} \xi_i^{ks}, \tag{2.20}$$

$$\begin{aligned} sous \qquad & \langle w^{ks}.\Phi(x_i)\rangle + b^{ks} \geq 1 - \xi_i^{ks}, \qquad si\ \ y_i = k, \\ & \langle w^{ks}.\Phi(x_i)\rangle + b^{ks} \leq -1 + \xi_i^{ks}, \qquad si\ \ y_i = s, \\ & \xi_i^{ks} \geq 0, \qquad \forall\, i = 1,2,\dots,l_{ks}. \end{aligned}$$

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 dansla 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 $k \in \{1, 2, \ldots, m\}$. 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 :

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

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

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

Cette règle proposée par Friedman [41] est connue sous le nom de vote majoritaire, et elle a été appliquée pour la première fois avec les SVM par Kreßel [40].

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).

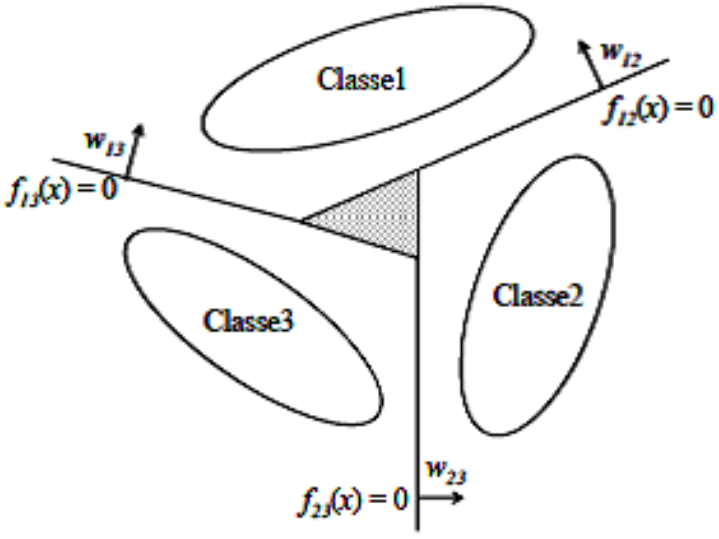


Figure 2.12 : Réduction de région d'ambiguïté hachurée pour l'approche une-contre-une.

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.

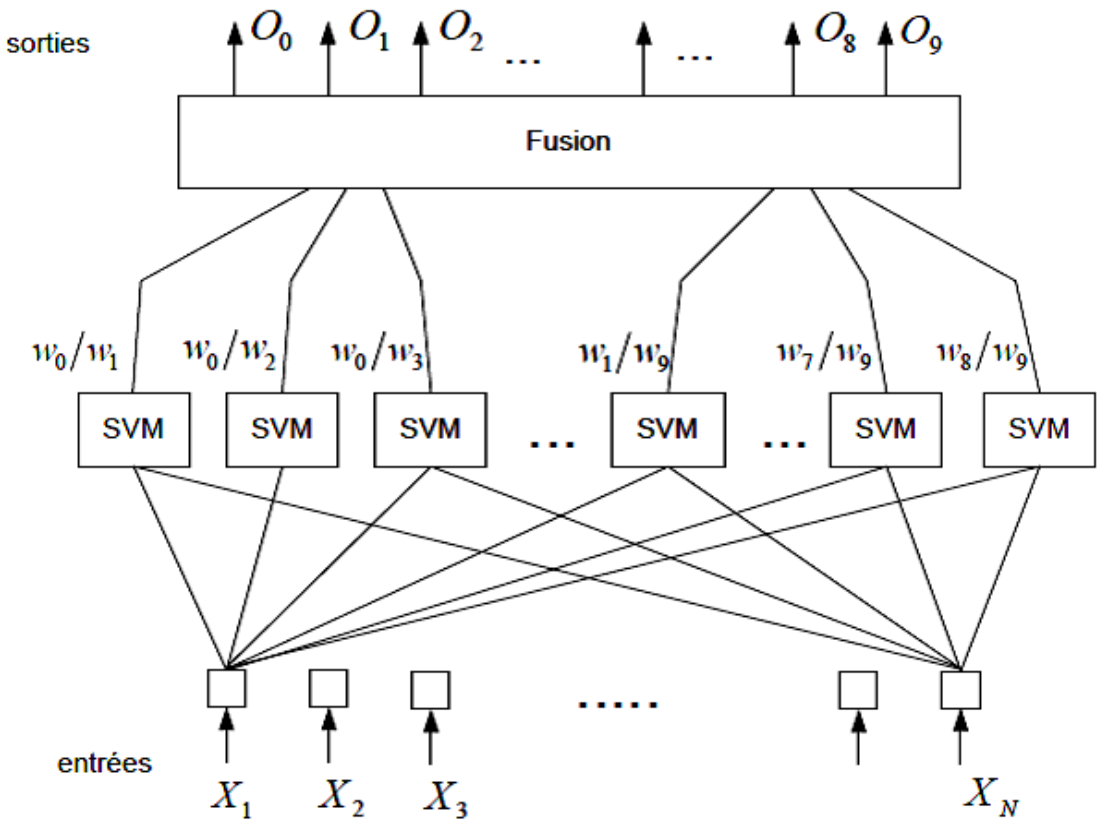


Figure 2.13 : Architecture du système en stratégie Une-contre-Une [30]

2.7.3 SVM floue

D'une façon similaire aux FSVM proposées pour le schéma de décomposition en une-contre-reste, S. Abe [37] 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} siest 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 posentles observations critiques pour être classées par l'hyperplan H_{ks} ; plus $M_{ks}(x)$ est petite plus la classification de x est difficile.

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

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

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.

[3]Membership functions

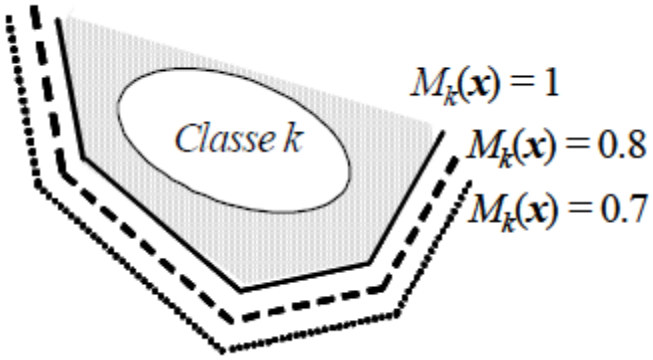


Figure 2.14 : Courbe de niveau définie par frontière pour la fonction $M_k(x)$ d'adhésion à la classe k. Fonction vaut 1 sur tout point de la zone hachurée.

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.

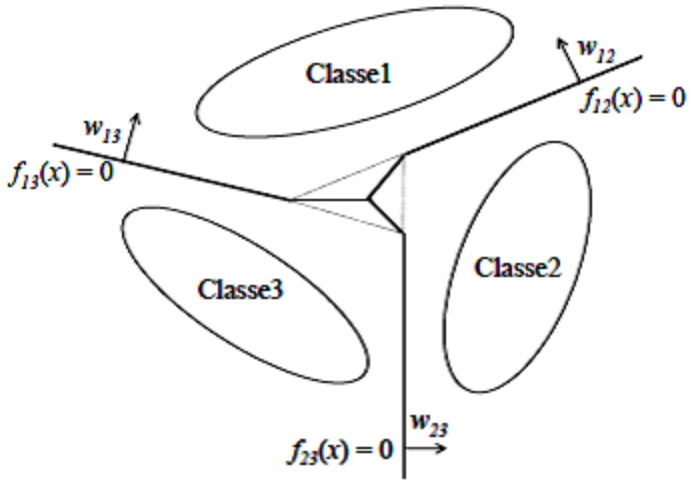


Figure 2.15 : Résolution de la région d'ambiguïté par les FSVMs.

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

Platt et al. [42] 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 tous 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 nœud 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.

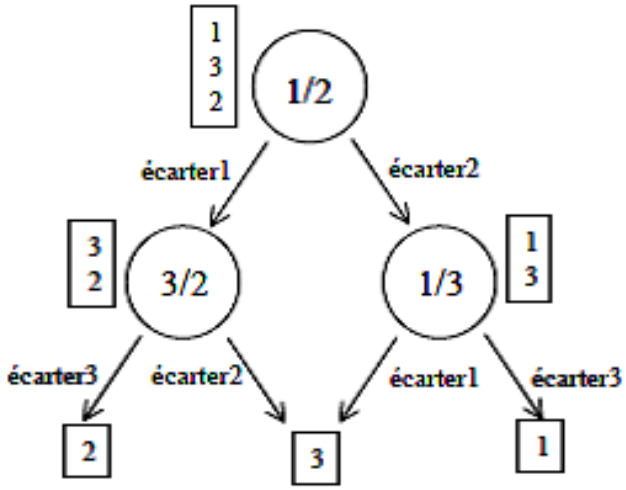


Figure 2.16 : Graphe de décision acyclique orienté à trois classes.

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,

[4]Decision Directed Acyclic Graph.

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 laliste initiale du haut vers le bas est le même que celui qu'on retrouve sur les feuilles de droiteà 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.

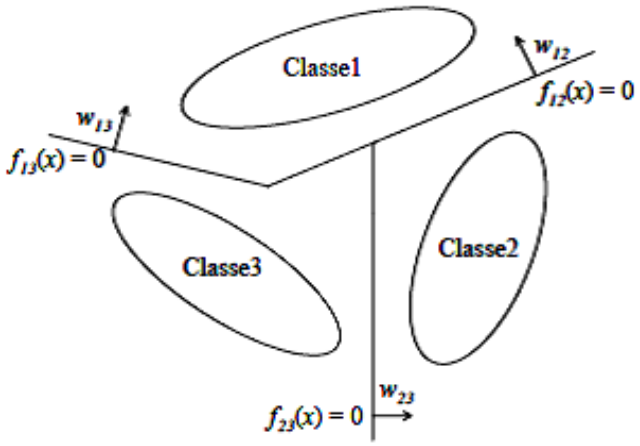


Figure 2.17 : Favorisation de la feuille du milieu pour DDAG pour l'affectant la région d'ambiguïté.

2.7.5 Graphe acyclique orienté adaptatif (ADAG[5])

Pontil et al. [43] 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. [44] ont employé la même méthode, qu'ils ont nommée graphe orienté acyclique adaptatif, dans le but d'améliorer les performances du DDAG.

[5]Adaptive Directed Acyclic Graph.

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 $\frac{m}{2}$ nœuds de la première couche. Au cours de cette première manche, chaque décision 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.

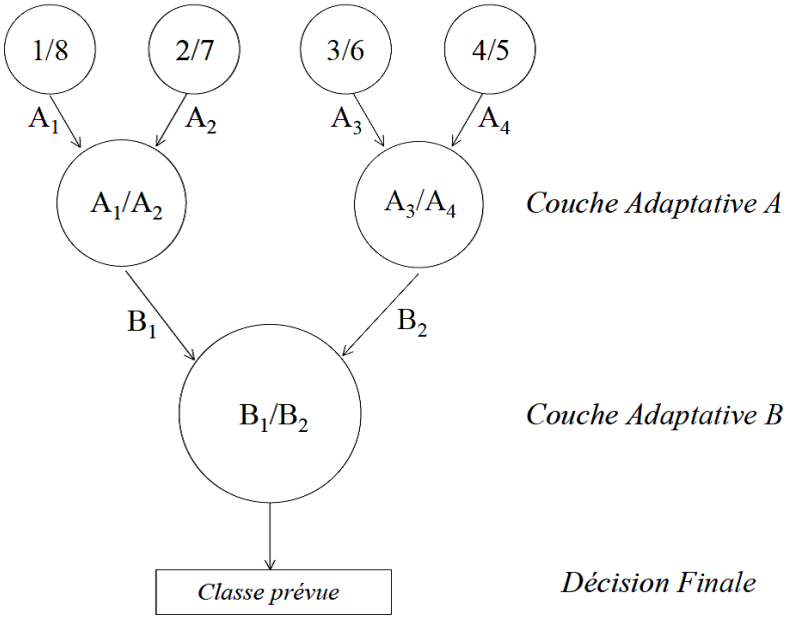


Figure 2.18 : ADAG à huit classes.

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 celuilié 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 $m = 7$.

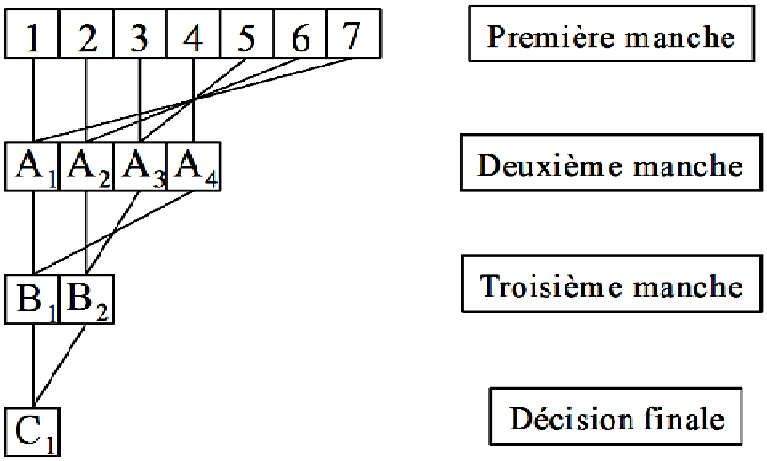


Figure 2.19 : Illustration par listes d'un ADAG à sept classes.

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 $\frac{m!}{\left[\frac{m}{2}\right]2^{\left[\frac{m}{2}\right]}}$cas de figures possibles pour la première couche de nœuds selon l'ordre de la liste initiale [36].

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

Dans le but de trouver une architecture optimale de l'ADAG, Phetkaew et al. [45] [46] 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.

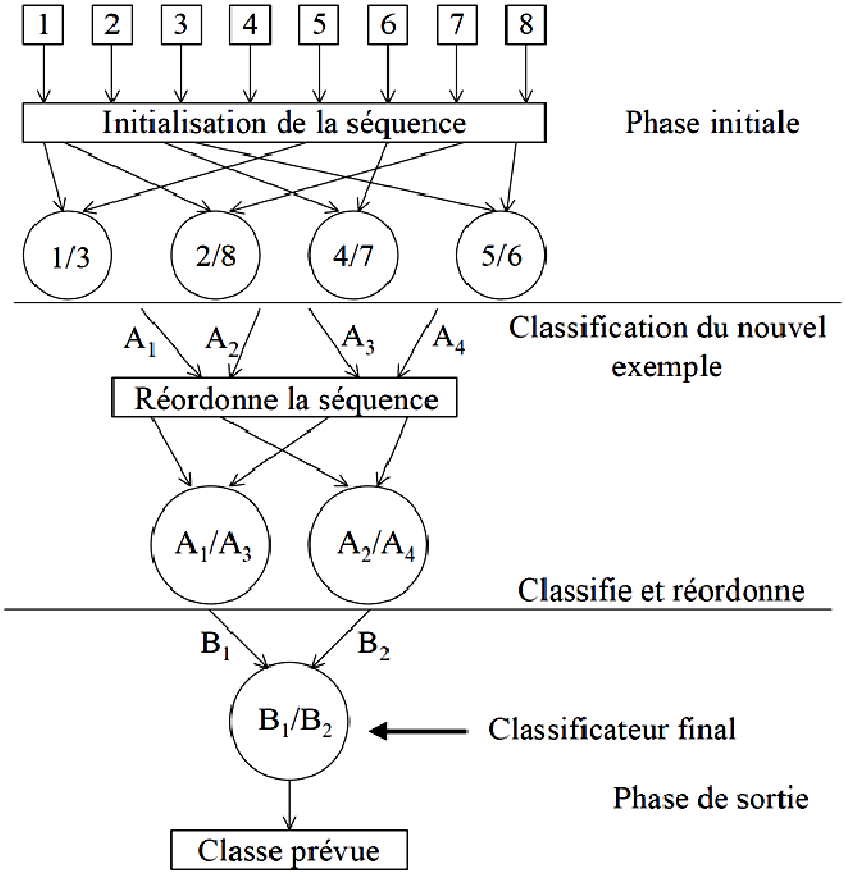


Figure 2.20 : Différentes étapes du RADAG.

[6]Reordering Adaptive Directed Acyclic Graph.

2.8 Avantages et inconvénients des SVM

2.8.1 Inconvénients

- Les SVM peuvent avoir des problèmes dans le cas où beaucoup de termes sont non significatifs pour la discrimination de classe [47].
- Elles ne servent pas à calculer des probabilités d'appartenance à la classe [48].
- L'utilisation des fonctions noyaux ne permet pas de sélectionner des caractéristiques importantes pour la classification [48].

2.8.2Avantages

- Étant donné que la complexité du modèle ne dépend pas de la dimension de l'espace de caractéristiques, les SVM sont particulièrement adaptées aux données à dimensions élevées (telles que les représentations vectorielles de texte).
- En utilisant seulement les vecteurs de support dans la frontière de décision, il est possible d'obtenir des résultats de faible densité en traitant avec des grands ensembles de données.
- La complexité du classeur peut être contrôlée par le paramètre C en rendant la marge plus lisse ou plus sinueuse au besoin.
- Il est possible d'avoir une idée de la capacité de généralisation du classeur même sans avoir vu qu'une seule donnée de test.
- Avec la SVM linéaire, c'est-à-dire, sans noyau, la frontière de décision construite correspond à un hyperplan dont les coefficients peuvent être interprétés comme l'importance que le classeur donne aux caractéristiques d'après leur importance pour la classification [49].
- En plus du premier et du quatrième avantage mentionné en haut, l'utilisation de la SVM linéaire est particulièrement bonne pour la classification des documents de texte pour les raisons suivantes, mentionnées par [50] :
    - La plupart des problèmes de catégorisation des textes sont linéairement séparables.
    - Il y a peu de caractéristiques insignifiantes.

2.9 Conclusion

Nous avons commencé ce chapitre par la présentation du principe de fonctionnement générale des machines à vecteurs de support et leur fondement mathématique. Nous avons exposé les deux cas des données linéairement séparable et non linéairement séparable. 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, Nous avons vu qu'il existe deux approches principales qui sont : une une-contre-une et une-contre-reste, 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 le 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 recherches se sont intéressées très tôt à la problématique de la sélection de modèle 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 toute fois 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 Métriques de performances

Généralement, les bons paramètres pour un modèle SVM sont ceux qui minimisent le taux d'erreur, nous devons considérer deux critères : l'erreur d'apprentissage et l'erreur de généralisation. 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. Donc il est très important d'avoir des métriques permettant de mesurer la performance d'un modèle SVM pour pouvoir comparer différents modèles de décision entre eux.

Deux métriques sont généralement utilisées [51] :

3.2.1 Taux de reconnaissance

La métrique intuitive utilisée est la précision du modèle appelée aussi le taux de reconnaissance. Elle représente le rapport entre le nombre d'exemples correctement classés et le nombre total d'exemples testés. L'équation 3.1 donne la formule utilisée.

$$P = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i))$$

Avec, (3.1)

$$L = \begin{cases} 1 & si\ y_i = f(x_i) \\ 0 & \text{sinon} \end{cases}$$

Généralement, la précision est donnée sous forme de pourcentage ce qui nécessite de multiplier la précision de l'équation 3.1 par 100.

3.2.2 Matrice de confusion

La mesure précédente donne le taux d'erreurs commises par le modèle de décision obtenu (100 − précision), mais ne donne aucune information sur la nature de ces erreurs. Dans la plupart des cas d'application, il est très important de connaître la nature des erreurs commises. Par exemple, lors de l'utilisation d'un modèle de décision pour des objectifs médicaux, considérer un échantillon non cancéreux alors qu'il l'est, est beaucoup plus grave de considérer un échantillon cancéreux alors qu'il ne l'est pas.

Dans le cas de classification binaire, le résultat de test d'un modèle peut être une possibilité parmi quatre :

$$\begin{cases} f(x_i) = +1 \quad et \quad y_i = +1 \quad correcte \quad positive \\ f(x_i) = +1 \quad et \quad y_i = -1 \quad fausse \quad positive \\ f(x_i) = -1 \quad et \quad y_i = -1 \quad correcte \quad négative \\ f(x_i) = -1 \quad et \quad y_i = +1 \quad fausse \quad négative \end{cases} \tag{3.2}$$

Si le modèle donne une classe positive pour un exemple d'une classe positive, on dit que c'est une classe correcte positive (CP). Si par contre l'exemple appartient à la classe négative on dit que c'est une classe fausse positive (FP). Si le modèle donne une classe négative pour un exemple d'une classe négative, le résultat est une classe correcte négative (CN), si, par contre, la classe de l'exemple est positive le résultat est qualifié de classe fausse négative (FN).

La matrice de confusion (Tableau3.1) est une matrice qui contient en lignes les observations y, et en colonnes les prédictions$f(x)$. Les éléments de la matrice représentent le nombre d'exemples correspondants à chaque cas.

Tableau 3.1 : Matrice de confusion pour la classification binaire

Observations (y)	Prédictions (f)	
	$+1$	-1
$+1$	CP	FN
-1	FP	CN

Un modèle sans erreurs aura ses résultats concentrés sur la diagonale de sa matrice de confusion (CP et CN). Dans le cas multiclasse la matrice sera plus large avec les classes possibles au lieu des deux classes +1 et -1. La précision P du modèle peut être calculée à partir de la matrice de confusion comme suit :

$$P = \frac{CP + CN}{CP + FP + CN + FN} \tag{3.3}$$

Deux autres mesures sont utilisées dans la littérature : la sensitivité Sv et la spécificité Sp. La sensitivité représente le rapport entre les observations positives correctement prédites et le nombre des observations positives, et la spécificité représente le rapport entre les observations négatives correctement prédites et le nombre total des observations négatives (équation 3.4).

$$\begin{cases} Sv = \dfrac{CP}{CP + FN} \\ Sp = \dfrac{CN}{CN + FP} \end{cases} \tag{3.4}$$

3.3 Méthodes de sélection de modèle SVM

La qualité d'un modèle de décision obtenu par le SVM dépend de plusieurs paramètres, à savoir le paramètre de pénalisation sur les multiplicateurs de Lagrange C, le noyau utilisé et ses paramètres et les exemples utilisés pour l'entrainement (les vecteurs supports). Le choix des valeurs de ces paramètres se fait actuellement à travers plusieurs essais et évaluation pour atteindre des performances satisfaisantes du modèle.

Les paramètres optimaux pour un modèle donné sont les paramètres qui lui permettent de donner une précision de 100%.Cette situation serait idéale si l'ensemble des exemples représentait parfaitement l'ensemble de tous les exemples possibles. Le modèle appris peut donner une très grande précision face aux exemples d'entrainement, mais se comporte très mal avec les nouveaux exemples.

Cela représente un phénomène très connu en apprentissage qui est le sur-apprentissage ou l'apprentissage par cœur. Le sur-apprentissage donne, généralement, des modèles à faible capacité de généralisation, et par conséquent la mesure de précision n'est pas suffisante pour qualifier les performances d'un modèle. Les méthodes d'évaluation permettent de tirer des conclusions sur le comportement d'un modèle face à tout l'espace d'exemples en limitant

l'influence des exemples d'entrainement et du bruit qui peut y exister (erreurs d'étiquetage, erreurs d'acquisition, ...) et leur ordre sur le modèle appris [51].

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 [30]. C'est le dilemme «Biais-Variance» bien connu en théorie d'apprentissage (Figure 3.1).

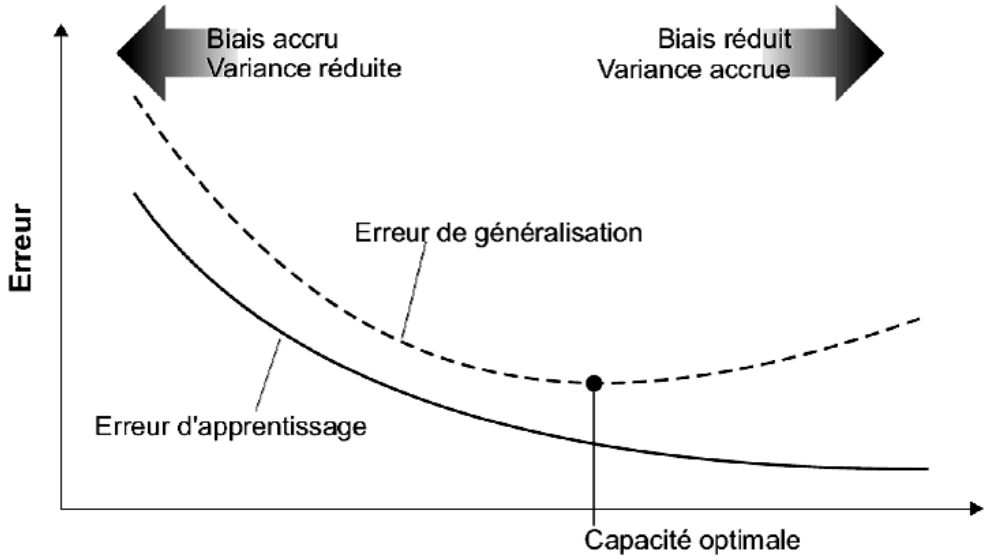


Figure 3.1 : Compromis biais-variance

Dans ce qui suit, nous présentons un aperçu des principales techniques utilisées pour la sélection de modèle SVM.

3.3.1 Méthode HoldOut

La méthode HoldOut suppose que les exemples disponibles couvrent suffisamment tout l'espace d'exemples. Elle consiste à diviser l'ensemble des données en deux parties, la première partie est utilisée pour l'entrainement et la deuxième pour les tests. Le test du modèle obtenu sur la partie de test permet de donner une idée sur son comportement en dehors des exemples d'entrainement et éviter le phénomène de sur-apprentissage. Le modèle qui maximise la précision pour tout l'espace d'exemple est donc celui qui la maximise pour la partie de test du fait que cette partie représente la majorité de l'espace [51].

Une question importante qui se pose pour cette méthode est comment choisir les deux parties, puisque ce choix a une grande influence sur la qualité du modèle. Le pire est de mettre les exemples positifs dans une partie et les exemples négatifs dans l'autre. La méthode qui suit répond à cette question.

3.3.2 Validation croisée

La validation croisée (K-fold cross validation) est une bonne technique d'évaluation de la performance en généralisation d'un classifieur. L'idée derrière cette technique est de tester le modèle du classifieur obtenu sur des données qui n'ont pas participé à l'apprentissage, afin de permettre de prédire le comportement du classifieur face aux nouvelles données.

Pour minimiser l'influence du choix du partitionnement de l'ensemble des exemples, la validation croisée subdivise l'ensemble d'entrainement initial en k sous ensemble disjoints $D_1, D_2, .., D_k$ de même taille.L'entrainement et le test sont effectués k fois. A l'itération i le sous-ensemble D_i est réservé pour le test et le reste des exemples sont utilisés pour entrainer le modèle. La précision finale du modèle est égale à la moyenne des k précisions de test.

Plus précisément, si l'on a une base d'apprentissage A_p contenant N éléments : $A_p = \{X_1, \dots, X_N\}$ la validation croisée consiste à appliquer les cinq étapes suivantes :

1. Découper l'ensemble des exemples en k sous-ensembles disjoints de taille $\frac{N}{k}$.
$$A_p = D_1, D_2, .., D_k$$
2. Apprendre sur les $(k-1)$ sous-ensembles.
3. Calculer la précision sur la k^{ime} partie.
4. Réitérer le processus k fois.
5. Obtenir la précision finale en calculant la moyenne des k précisions 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 [52]. La variance des résultats obtenus est réduite lorsque k croît, mais l'inconvénient, c'est le temps de calcul qui devient très long.

3.3.3 Leave-one-out

La méthode Leave-One-Out [53] est un cas particulier de la validation croisée où$k = N$. Il y a donc *N* phases d'entraînement, une pour chaque exemple de la base A_P. Á chaque itération, le modèle est entrainé sur $(N-1)$ exemples et testé sur l'exemple exclu de l'entrainement. On obtient à la fin *N* précisions, la précision du modèle est égale à leur moyenne.

Cette estimation de la précision 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.

3.3.4 Le Bootstrap

La méthode de Bootstrap [54], appelée aussi échantillonnage par remplacement, entraine le modèle sur un ensemble de *N* exemples choisis aléatoirement de l'ensemble des exemples, des exemples peuvent être choisis plus d'une fois et d'autre ne se seront pas choisis du tout. Les exemples non choisis pour l'entrainement sont utilisés pour le test. Cette opération peut être répétée plusieurs fois pour obtenir une précision moyenne du modèle.

Parmi les méthodes de Bootstrap les plus utilisées, la méthode "632" qui tire son nom du fait que 63.2 % des exemples contribuent à l'entrainement et les restants (36.8%) contribuent aux tests.

À chaque prélèvement, un exemple a une probabilité $\frac{1}{N}$ d'être choisi et $(1-\frac{1}{N})$ de ne pas l'être, et puisqu'on répète le prélèvement *N* fois, chaque exemple aura une probabilité de $(1-\frac{1}{N})^N$ de ne pas être choisi du tout dans un ensemble d'entrainement. Si *N* est grand cette probabilité approche de $e^{-1} = 0.368$. La méthode répète le processus k fois et la précision finale *P* est donnée par :

$$P = \sum_{i=1}^{k} (0{,}632) \times P_{i_{test}} + 0{,}368 \times P_{i_{\text{entr}}} \tag{3.5}$$

Où $P_{i_{test}}$ est la précision du modèle entrainé sur les exemples choisis dans l'itération i, appliqué sur les exemples de test dans la même itération. $P_{i_{\text{entr}}}$ est la précision du même modèle appliqué sur les données d'entrainement.

3.3.5 Grille de recherche

C'est une méthode classique qui discrétise l'espace des modèles [55]. 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 [56], 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, on cherche le meilleur couple (C, γ). On peut utiliser des séquences exponentielles [52] $:\mathrm{C} = 2^{-5}, 2^{-4}, \dots, 2^{15}$ et $\gamma = 2^{-15}, 2^{-14}, \dots, 2^{3}$. 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 on peut donc facilement paralléliser la recherche [35].

3.3.6 Bornes sur l'erreur de généralisation

Dans l'idéal, nous souhaitons choisir les bons paramètres qui minimisent l'erreur de généralisation pour un modèle SVM, cette erreur est inaccessible en pratique, mais pour laquelle on dispose soit d'estimations soit de majorants. Un des avantages essentiels des SVM est que la solution produite à partir d'un ensemble d'apprentissage fournit des informations sur l'erreur en généralisation en plus de la définition classique d'une fonction de décision.

Plusieurs travaux ont été réalisés pour déterminer une borne supérieure de l'erreur de généralisation.

Il a été démontré par Vapnik [1] que l'erreur de généralisation des SVM est bornée par R^2/γ^2 , où R est le rayon de la plus petite boule recouvrant les données d'apprentissage et γ désigne la marge. De plus, dans le cas où les données sont linéairement séparables, la quantité $T = R^2/\gamma^2$ fournit une estimation de l'erreur de généralisation.

Depuis cette première borne proposée par Vapnik [1] qui emploie le nombre des vecteurs supports, plusieurs majorants ont été proposés à travers les années. Sous certaines conditions, nous pouvons toujours établir une relation entre les différentes bornes proposées (Voir Vapnik et al. [57], Chapelle et al. [58] et Chapelle [59] pour une revue détaillée).

3.4 Conclusion

Le problème de réglage des paramètres de modèle SVM semble lourd à résoudre à cause de la difficulté de choisir un bon compromis biais-variance. Pour ce faire, on fait souvent appel à la procédure de validation croisée ou à des méthodes minimisant l'erreur de généralisation ou l'une des bornes de généralisation par rapport à ces paramètres, Aucune de ces méthodes ne donne des résultats idéaux, ce qui indique la difficulté de la tâche à accomplir.

Il existe d'autres méthodes proposées pour résoudre ce type de problème, un sous-ensemble d'entre elles utilise des méthodes désignées sous le terme de méta-heuristiques, décrites dans le chapitre suivant.

Chapitre IV

Optimisation avec méta-heuristiques

4.1 Introduction

Les problèmes d'apprentissage correspondent à la recherche d'une fonction de décision h^*de qualité optimale, suivant un critère q, l'espace des fonctions de décision admissibles étant défini par un ensemble de paramètres ou variables θlibres, généralement contraint dans un ensemble de valeurs possibles Θ. La recherche de la fonction de décision optimale correspond donc à un problème d'optimisation :

$$\theta^* = \arg\max\ q(h_\theta)\ \ ,\theta\epsilon\Theta \tag{4.1}$$

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

$$\theta^* = \arg\max\ f(\theta)\ \ ,\theta\epsilon\Theta \tag{4.2}$$

Avec θ^* la solution de l'espace des solutions Θ qui optimise la fonction objectif $f(\mathrm{f} \equiv q\ (h_\theta))$ qui représente la qualité d'une solution θ pour un problème donné [60].

Dans le cas des SVM, le choix des valeurs optimales de γ et C (pour un noyau gaussien par exemple) correspond à la recherche d'un modèle θ optimal ($\theta \equiv (C, \gamma)$). 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 méthodes méta-heuristiques ont été fortement développées et utilisées dans les années 1980 pour résoudre initialement des problèmes d'optimisation difficiles issus de la recherche opérationnelle. Un exemple d'application industrielle est le placement automatique des composants électroniques afin que la minimisation de la longueur des pistes dans l'implantation des circuits intégrés soit proche de l'optimal. Ce problème a comme illustre parent celui du voyageur de commerce. C'est un problème combinatoire pour lequel les méta-heuristiques sont connues pour produire des solutions très efficaces même lorsqu'il est de grande taille.

Les méta-heuristiques ont été ensuite appliquées à de nombreux domaines dont l'apprentissage artificiel. La problématique qui correspond à la sélection d'un sous-ensemble réduit d'attributs pertinents ou d'exemples pertinents est une des applications majeures des méta-heuristiques dans le cadre de l'apprentissage artificiel [61]. Donc tout problème d'apprentissage confronté à des problèmes d'explosion combinatoire ou d'espace de recherche avec des minima locaux peut tirer profit des méthodes méta-heuristiques.

L'avantage des méthodes méta-heuristiques est qu'elles sont suffisamment génériques pour permettre d'optimiser une large gamme de problèmes différents, sans nécessiter de changements profonds dans le principe algorithmique employé. Pour un problème particulier, l'adéquation d'un algorithme de type méta-heuristique consiste à définir des briques élémentaires adaptées à ce problème.

Ces briques élémentaires sont alors directement utilisées par l'ossature générale de la méthode méta-heuristique. Un des avantages de ces méthodes est que lorsque deux problèmes sont de nature similaire, l'effort d'adaptation est réduit pour passer de l'optimisation de l'un à l'autre, car certaines briques de base peuvent être réutilisées ou nécessiter des modifications mineures.

Dans ce chapitre, nous nous proposons de rappeler les fondements essentiels des méthodes méta-heuristiques en présentant brièvement le recuit simulé et les algorithmes génétiques, et nous insistons sur les méthodes désignées sous les termes de recherche avec tabous et optimisation par essaim de particules lesquelles seront adaptées au problème de sélection de paramètres de modèle SVM. Cette adaptation représente notre première contribution dans cette thèse.

4.2 Recuit simulé

Le recuit simulé est fondé sur une analogie entre un processus physique (le recuit) et le problème de l'optimisation. Le recuit simulé [62], en tant que méta-heuristique s'appuie en effet sur des travaux de Metropolis [63], visant à simuler l'évolution d'un solide vers son état d'énergie minimale.

Dans le cadre d'un problème d'optimisation, la fonction objective à minimiser est alors assimilée à l'énergie du système. On introduit également un paramètre fictif T, apparenté à la température, que l'on fait décroître au fur et à mesure des itérations, afin de simuler le refroidissement. De la valeur de ce paramètre, va dépendre la probabilité d'acceptation des nouvelles solutions.

On démarre alors l'algorithme avec une solution unique x que l'on cherche à améliorer. On perturbe cette solution afin d'obtenir une nouvelle solution x_0 dans le voisinage de la première. Ensuite, on calcule l'écart des valeurs de la fonction objectif pour ces deux solutions : $\Delta f = f(x_0) - f(x)$. On se retrouve alors dans deux cas possibles :

$\Delta f \leq 0$, la nouvelle solution est meilleure que la solution initiale, on la remplace donc par : $x = x_0$.

$\Delta f > 0$, la nouvelle solution est moins bonne que la solution initiale. Cependant, on a tout de même la possibilité de la remplacer avec une probabilité d'acceptation :

$$prob(\Delta f, T) = \exp\left(\frac{-\Delta f}{k_B.T}\right) \tag{4.3}$$

Où T est la température du système et k_B une constante physique connue sous le nom de constante de Boltzmann.

L'acceptation de la nouvelle solution est décidée en générant de manière aléatoire un nombre q [0,1]. Si q est inférieur ou égal à $prob\,(\Delta f, T)$, alors la nouvelle solution est acceptée. Autrement la solution actuelle est maintenue. L'utilisation répétée d'une telle règle fait évoluer le système vers un état d'équilibre thermique. Lorsqu'aucun état nouveau n'est accepté à une température T donnée, on considère que le système est gelé et on suppose qu'il a atteint la meilleure solution [64].

Le recuit simulé présente l'avantage d'offrir des solutions de bonne qualité, tout en restant simple à programmer et à paramétrer. Il offre autant de souplesse d'emploi que l'algorithme de recherche local classique. Aarts, Korst et Laarhoven [65] ont par ailleurs démontré que, sous certaines conditions de décroissance de la température, l'algorithme du recuit simulé converge en probabilité vers un optimum global lorsque le nombre d'itérations tend vers l'infini.

Par contre, l'un des inconvénients du recuit simulé est que l'algorithme peut être piégé à une basse température dans un minimum local. Le paramétrage de la température est le principal problème du recuit : il peut être rebutant et très empirique. On constate également qu'il faut une diminution de la température « suffisamment lente », donc un certain nombre important d'itérations, pour l'amélioration de la solution.

Dans [66], les auteurs proposent une approche SA-SVM basé sur le recuit simulé pour l'optimisation des paramètres des SVMs et pour obtenir le sous ensemble optimal d'indicateurs. Le sous ensemble sélectionné, est ensuite adopté pour l'entrainement et le test des SVM, afin d'obtenir un meilleur taux de reconnaissance en classification. Une comparaison des résultats avec ceux obtenus avec d'autres approches prouvent que l'approche SA-SVM permet d'améliorer les performances de classification.

4.3 Algorithmes génétiques

Les algorithmes génétiques ont été conçus par J. Holland [67]. Ce sont des algorithmes d'optimisation. On trouvera une présentation détaillée de ces processus dans de nombreux ouvrages [68]. 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'individus qui sont soumis 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 choisi aléatoirement dans le chromosome des parents,
- la mutation produit un nouvel individu à partir d'un seul individu ; on opère un complément à 1 d'un certain bit.

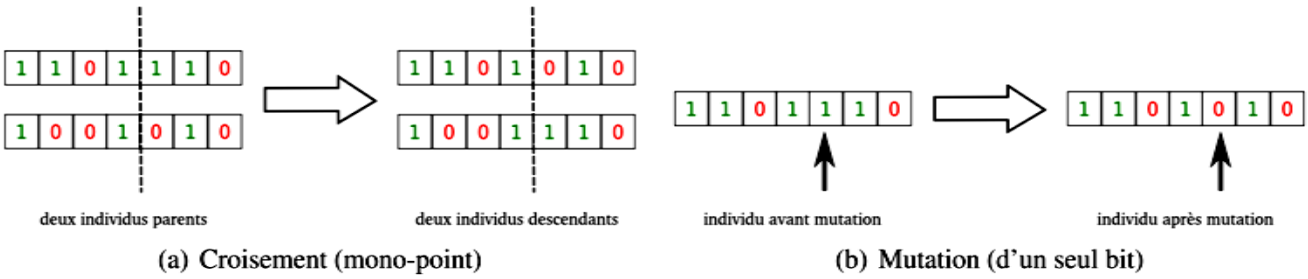


Figure 4.1 : Principe des opérateurs génétiques dans le cas d'individu codé sur 7 bits [35].

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 (Figure 4.1) font que les individus de la population courante interagissent entre eux au lieu d'évoluer chacun indépendamment.

Algorithme 4.1 : Algorithmes génétiques

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.

Les auteurs dans [69] proposent une méthode de sélection de modèle SVM basée sur un algorithme génétique hybride et un critère de minimisation de l'erreur empirique. La méthode de descente de gradient a été intégrée aux algorithmes génétiques afin de rechercher un meilleur paramètre du noyau RBF. Plusieurs autres applications des AGs pour la sélection de modèle SVM ont été également présentées dans [70], [71], [72].

4.4 Recherche avec tabous

La recherche Tabou[7] est une méthode méta-heuristique pour l'optimisation difficile de la famille des techniques de recherche locale, présentée pour la première fois par Fred Glover en 1986 [73]. Elle se distingue des méthodes de recherche locale simples par l'introduction de la notion de trace dans la stratégie d'exploration des solutions afin de diriger au mieux la recherche dans l'espace. Cette méthode s'est révélée particulièrement efficace et a été appliquée avec succès à de nombreux problèmes difficiles [74].

4.4.1 Concepts de base

- **Un Voisinage** est un sous-ensemble de solutions qu'il est possible d'atteindre par une série de transformations données appelées mouvements. Le voisinage a un impact important sur le comportement des méta-heuristiques.
- **Un Mouvement** est un déplacement par rapport à un point fixe dans l'espace de solution et à un moment déterminé. C'est une opération élémentaire permettant de passer d'une solution à une solution voisine (changement de la valeur d'une variable, échange de deux variables).
- **Un Mouvement non améliorateur** est un mouvement fournissant un minimum local associé à une solution voisine S' pire que l'actuelle (S^*).

[7] En anglais :Tabu Search.

- **Un Mouvement tabou** est un mouvement non désirable, tombait dans le cas d'un minimum local c'est-à-dire, c'est déjà classé dans la liste tabou.
- **Une Liste tabou** est une liste mémorisant les dernières solutions visitées et interdisant tout déplacement vers une solution de cette liste. Il peut exister plusieurs listes simultanément. Une liste T avec trop d'éléments peut devenir très restrictive. Il a été observé que trop de contraintes (tabous) forcent le programme à visiter des solutions voisines moins bonnes à la prochaine itération. Une liste T contenant peu d'éléments, peut conduire à des mouvements cycliques.

4.4.2 Principe de la méthode

L'algorithme se déplace de solution en solution (en visitant éventuellement des solutions moins bonnes) en s'interdisant de revenir à une solution déjà rencontrée [75].

Cette méthode peut être résumée par les étapes suivantes :

- A chaque itération, on examine une solution voisine $V(s)$.
- Pour éviter toute boucle, on crée une liste T qui mémorise les dernières solutions visitées et qui interdit tout déplacement vers une solution de cette liste. Cette liste T est appelée liste Tabou.
- On conserve en cours de recherche, la meilleure solution trouvée S^* .
- On s'arrête dès que le critère d'arrêt est bien vérifié.

Algorithme 4.2 : Algorithme général de la recherche tabou

Générer une solution initiale S de manière aléatoire $S^* \leftarrow S$; $C^* \leftarrow F(S)//S^*$est la meilleure solution rencontrée, C^* est son coût et F la fonction objectif.
Ajouter Sà la liste tabou; $k \leftarrow 0$;
Répéter tant qu'un critère de fin n'est pas vérifié
  Choisir parmi le voisinage de S_k, $V(S_k)$, le mouvement qui minimise F et qui n'appartient pas à la liste tabou, meilleur(S_k);
  $S_{k+1} =$ meilleur(S_k);
  Si la liste tabou est pleine alors
  Remplacer le dernier élément de la liste tabou par S_{k+1};
  Si non
   Ajouter S_{k+1} à la liste Tabou ;
  Fin si
  Si $F(S_{k+1}) < C^*$ alors
  $S^* \leftarrow S_{k+1}$; $C^* \leftarrow F(S_{k+1})$
  Fin si
Fin tant que

4.4.3 Critère d'arrêt

Le critère d'arrêt sert à détecter la fin du traitement de l'algorithme, le moment où l'on considère que la solution trouvée est assez bonne. Il peut être défini par :

- Un nombre maximum d'itérations, fixé à l'avance.
- Un nombre d'étapes n'ayant pas amélioré la solution S^*.
- Un temps limite après lequel la recherche doit s'arrêter.

4.4.4 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).

4.4.5 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.

4.4.6 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.

4.4.7 Stratégie de détermination de la taille de la liste tabou

La taille de la liste tabou est à déterminer empiriquement. Ces stratégies peuvent être statiques (détermination de cette taille en fonction de la nature du problème) ou dynamique (adaptation de la taille de la liste tabou en cas des cycles détectés dans le parcours de l'espace de recherche par l'algorithme tabou [76]).

Plus la taille de la liste est petite, plus l'algorithme explore des petites régions. Plus la taille est grande, plus la recherche est contrainte de s'opérer sur de plus larges étendues. Il est possible de régler automatiquement la taille: si le système constate que certaines

solutions reviennent régulièrement, il peut alors décider d'augmenter dynamiquement la taille de la liste, de façon à forcer une diversification.

4.4.8 Avantages et inconvénients

La méthode Tabou peut être facilement paramétrée. Il existe en effet au maximum deux paramètres qui peuvent réellement influencer la recherche taboue. Le premier consiste à bien choisir la taille de la liste taboue. Elle est généralement déterminée empiriquement et varie selon les problèmes, c'est une donnée essentielle. Une taille trop petite peut amener l'algorithme à boucler sur une zone locale de l'espace de recherche tandis qu'une trop grande taille saturera rapidement les ressources disponibles.

Le deuxième est le choix du critère d'arrêt. Ce dernier peut s'avérer difficile à déterminer pour éviter de ne prolonger trop longtemps la recherche ou de rater l'optimum global recherché. [77]. L'efficacité de la méthode Tabou fait qu'elle est largement employée dans les problèmes d'optimisation combinatoire: elle a été testée avec succès sur les grands problèmes classiques: problème du voyageur de commerce, problème d'ordonnancement d'ateliers, problème de constitution de planning ...etc [78].

Toutes les méthodes dites méta-heuristiques, ont une caractéristique commune : l'absence de garantie de résultat. En effet dans la majorité des cas, elles ne disposent d'aucune information sur la qualité des solutions obtenues. Il difficile de prévoir la performance de méta-heuristiques (qualité et temps). De plus, la méthode Tabou exige une gestion de la mémoire.

4.4.9 Adaptation de la recherche Tabou à notre problème

Nous avons utilisé le principe de la recherche Tabou pour sélectionner les paramètres de modèle SVM et la validation croisée (Section 3.3.2) comme métrique pour évaluer la précision du modèle.

A chaque itération, la précision de validation croisée est évaluée sur l'ensemble de points obtenus par les perturbations positives et négatives de chaque paramètre autour de la solution actuelle. La solution qui maximise la précision constitue alors le point de départ de l'itération suivante. L'heuristique de recherche Tabou interdit les déplacements susceptibles de récupérer les configurations récemment rencontrées.

Par exemple pour un noyau RBF, les mouvements possibles correspondent à ajouter ou soustraire un pas ΔC ou $\Delta\gamma$ respectivement à la constante de régularisation C ou à la largeur du noyau γ. Donc les valeurs des paramètres C ou γ 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'itérations maximal est atteint.

Donc un pas a été associé à chaque hyper paramètre du modèle SVM, à la fin de chaque itération les pas et la taille de la liste taboue sont ajustés de manière adaptative. Chaque fois que la précision de validation croisée est diminuée, le pas est multiplié par un facteur supérieur à l'unité et on augmente la taille de la liste Tabou pour diversifier la recherche, si la précision augmente, alors le pas est multiplié par un facteur inférieur à l'unité et on diminue la taille de la liste taboue pour intensifier la recherche. La mise à jour des pas est effectuée uniquement si le paramètre correspondant est modifié au cours de cette itération (voir l'algorithme 4.3).

Lorsque la liste taboue est pleine, celle-ci est gérée comme une liste circulaire en FIFO (First In First Out) : on élimine les plus anciens mouvements tabous et on insère la nouvelle solution.

Algorithme 4.3 : Recherche tabou pour la sélection de paramètres du noyau RBF

1- Initialiser les paramètres C et γ.
2- Initialiser la taille de la liste taboue T et le pas ΔC et $\Delta\gamma$.
3- Ajouter les paramètres C et γ à la liste taboue.
4- Calculer la précision P par la méthode de la validation croisée, $P^* \leftarrow P$.
5- Enregistrer les paramètres et la précision dans le fichier F de résultats.
6- Choisir le meilleur mouvement M qui n'existe pas dans la liste taboue parmi :
   - M1 : $C + \Delta C$ et γ
   - M2 : $C - \Delta C$ et γ
   - M3 : C et $\gamma + \Delta\gamma$
   - M4 : C et $\gamma - \Delta\gamma$
7- Ajouter les paramètres à la liste taboue.
8- Calculer la précision P de la validation croisée
   Si $P > P^*$ alors
      $P^* \leftarrow P$
      Ajouter les paramètres et P^* au fichier F.
   Fin Si
9- Adapter la taille T de la liste taboue et le pas ΔC ou $\Delta\gamma$ selon le principe expliqué ci-dessus.
10- Si le nombre d'itérations maximal n'est pas atteint aller à 6.

4.5 Optimisation par essaim particulaire

Les algorithmes d'optimisation par essaims particulaires (**P**article **S**warm **O**ptimization (PSO)) ont été initialement introduits en 1995 par James Kennedy et Russell Eberhart comme une technique d'optimisation globale [79]. Ces algorithmes sont inspirés des essaims d'insectes (ou des bancs de poissons ou des nuées d'oiseaux) et de leurs mouvements coordonnés. En effet, tout comme ces animaux se déplacent en groupe pour trouver de la nourriture ou éviter les prédateurs, les algorithmes à essaim de particules recherchent des solutions pour un problème d'optimisation. Les individus de l'algorithme sont appelés particules et la population est appelée essaim.

Chaque particule est une solution candidate au problème et est représentée par une vitesse, un emplacement dans l'espace de recherche et dispose d'une mémoire qui l'aide à se souvenir de sa meilleure position précédente. Un essaim se compose de N particules volant dans un espace de recherche de dimension d [80].

L'ensemble de particules auquel une particule i est topologiquement liée est appelé le voisinage de i. Le voisinage peut être la population entière ou un sous-ensemble de celle-ci. Diverses topologies ont été utilisées, les deux plus couramment utilisées sont connues sous le nom de P_g pour "global best" et P_i pour " local best ". La topologie traditionnelle des essaims particulaires connue sous le nom de P_g était celle où le meilleur voisin de toute la population a influencé la particule cible. Bien que cela puisse être conceptualisé comme un graphe entièrement connecté.

La topologie P_i est un réseau d'anneaux simples où chaque individu est connecté à K = 2 membres adjacents dans la matrice de population. [81] a souligné que la topologie P_g avait tendance à converger très rapidement avec une plus grande chance de se coincer dans l'optimum local. D'autre part, la topologie P_i était plus lente mais complètement explorée, et généralement finit par un meilleur optimum. Les effets de diverses topologies de population sur l'algorithme des essaims particulaires ont été étudiés dans [82].

Dans la phase d'initialisation de PSO, les positions et les vitesses de tous les individus sont initialisées au hasard. À chaque itération, une particule i ajuste sa position X_i et sa vitesse V_i le long de chaque dimension d de l'espace de recherche, en fonction de la meilleure position P_i qu'elle a rencontrée jusqu'à présent dans son vol (aussi appelé personal best) et la meilleure position P_g trouvée par toute autre particule dans son voisinage topologique.

La vitesse définit la direction et la distance à laquelle la particule devrait aller. Il est mis à jour selon l'équation suivante :

$$V_{id}(t+1) = V_{id}(t) + c_1 r_1 \big(P_{id}(t) - X_{id}(t)\big) + c_2 r_2 \left(P_{gd}(t) - X_{id}(t)\right) \tag{4.4}$$

Où $= 1,2, \dots, N$, et N est la taille de l'essaim; r_1 et r_2 sont deux nombres aléatoires uniformément répartis dans l'intervalle [0, 1], c_1 et c_2 sont des termes constants connus sous le nom de coefficients d'accélération. Ils représentent l'attraction qu'une particule a soit vers son propre succès (la partie cognitive), soit vers le succès de ses voisins (la partie sociale).

La position de chaque particule est également mise à jour dans chaque itération en ajoutant le vecteur de vitesse au vecteur de position, c'est-à-dire,

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \tag{4.5}$$

Il est à noter que le terme "vitesse" est abusif car les vecteurs V_i ne sont pas homogènes à une vitesse. Il serait plus approprié de parler de "direction de déplacement". La stratégie de déplacement d'une particule est illustrée dans la Figure 4.2.

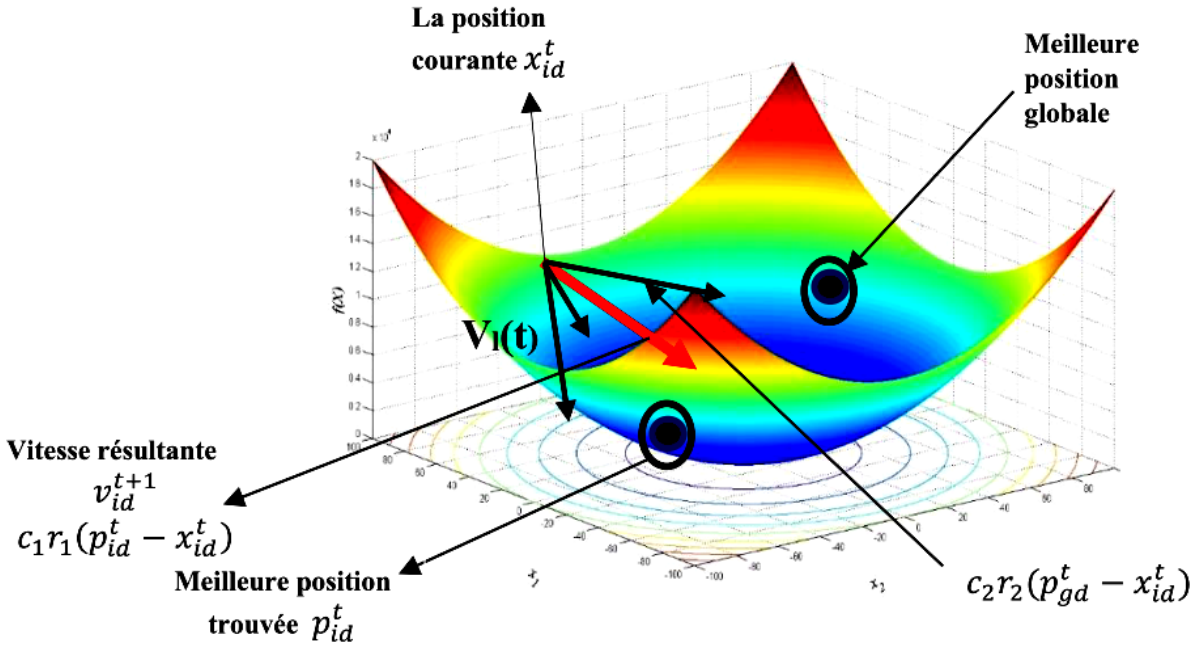


Figure 4.2 : Stratégie de déplacement des particules. [83]

La structure générale de l'algorithme PSO peut être résumée dans l'algorithme 4.4. Afin d'empêcher les particules de sortir de l'espace du problème, quelques paramètres sont définis pour limiter la vitesse de chaque particule [84], de sorte que chaque composante de V_i est

maintenue dans l'intervalle [$-V_{max}$, $+V_{max}$]. Le choix du paramètre V_{max} est important car il semble influencer l'équilibre entre exploration et exploitation [80].

Algorithme 4.4 : Optimisation par essaim particulaire (algorithme général)

Début
Initialiser aléatoirement une population de particules: positions et vitesses; sur d dimensions dans l'espace de recherche.
Tant que Critère d'arrêt n'est pas satisfait **faire**
  Pour chaque particule *i* **faire**
    Mettre à jour la vitesse de la particule selon l'équation (4.4) ;
    Mettre à jour la position de la particule selon l'équation (4.5) ;
    Evaluation de la fitness $f(X_i)$;
    Si $(f(X_i) < f(P_i))$ **alors**
      $P_i = X_i$;
    Fin Si ;
    Si $(f(X_i) < f(P_g))$ **alors**
      $P_g = X_i$;
    Fin Si ;
  Fin Pour ;
Fin Tant que ;
Fin.

Pour surmonter le problème de la convergence prématurée du PSO, de nombreuses stratégies ont été développées mais la plus populaire est l'inertie. Le poids d'inertie ω, introduit dans [85], a pour rôle d'équilibrer la recherche globale et la recherche locale. Il peut s'agir d'une constante positive ou même d'une fonction linéaire ou non-linéaire positive. Un poids d'inertie considérable encourage l'exploration globale (c'est-à-dire, diversifie la recherche dans tout l'espace de recherche) tandis qu'un faible poids d'inertie encourage l'exploitation locale (c'est-à-dire intensifie la recherche dans la région actuelle) [86]. En utilisant le poids d'inertie, l'équation (4.4) devient :

$$V_{id}(t+1) = \omega V_{id}(t) + c_1 r_1 \big(P_{id}(t) - X_{id}(t)\big) + c_2 r_2 \Big(P_{gd}(t) - X_{id}(t)\Big) \qquad (4.6)$$

4.6 Adaptation de PSO à notre problème

Le nombre de paramètres de modèle SVM dépend du type de noyau et leurs paramètres. Par exemple, le noyau polynomial a trois paramètres ($\gamma, coef\ et\ d$). Par conséquent, le modèle SVM a quatre paramètres si on ajoute le paramètre de régulation C. Dans ce cas, l'espace de recherche est de dimension $d = 4$.

L'algorithme PSO maintient un ensemble de particules. Chaque particule représente une solution candidate au problème étudié et considérée comme un objet présentant les caractéristiques suivantes.

- La position actuelle est un vecteur $X = (x_1,..,x_d)$ où, $x_1,..,x_d$ représentent les valeurs des paramètres du modèle SVM et d la dimension de l'espace de recherche.
- La vitesse actuelle $V = (v_1,..,v_d)$ permet de modifier progressivement les valeurs des paramètres du modèle SVM selon les équations (4.6) et (4.5).
- La meilleure position visitée $P = (p_1,..,p_D)$ est utilisée pour stocker les meilleures valeurs des paramètres trouvées par la particule (une valeur précédente de X).
- La meilleure position parmi toutes les positions de particules $G = (g_1,..,g_D)$ représente les meilleures valeurs des paramètres SVM trouvés jusqu'à présent.

L'algorithme est exprimé comme suit:

Algorithme 4.5 : Algorithme de sélection de modèle SVM par PSO

Initialiser le nombre de particules N,
Initialiser la vitesse V_i et initialiser de manière aléatoire la position X_i de chaque particule,
Déterminer le nombre d'itérations maximal T_{max}.
$t = 0$
Répéter
  Pour $i = 1$ à N
    Calculer la précision de validation croisée de modèle SVM avec les paramètres x_i
    Si la précision de validation croisée de modèle SVM avec les paramètres x_i est supérieure à la précision de modèle SVM avec les paramètres p_i, alors
      Mettre à jour $p_i = x_i$
    Fin Si
    Si la précision de validation croisée de modèle SVM avec les paramètres p_i est supérieure à la précision de modèle SVM avec les paramètres g, alors
      Mettre à jour $g = p_i$
    Fin Si
  Fin pour
  Pour $i = 1$ à N
    Mettez à jour x_i en utilisant les équations (4.6) et (4.5)
  Fin pour
  $t = t + 1$
Jusqu'au $(t > T_{max})$
Retour g

4.7 Conclusion

Les méta-heuristiques constituent une classe de méthodes approchées et adaptables à un grand nombre de problèmes d'optimisation combinatoire. Elles ont démontré leur grande efficacité sur de nombreuses classes de problèmes.

Nous avons présenté dans ce chapitre l'ensemble des différentes méthodes méta-heuristiques. A travers quelques références, nous avons montré que ces méthodes ont déjà été appliquées avec succès au problème de sélection du modèle SVM. Nous avons détaillé plus particulièrement les méta-heuristiques correspondant à la recherche avec tabous et aux algorithmes d'optimisation par essaims particulaires, aussi nous avons présenté notre adaptation des deux méthodes au problème de sélection du modèle SVM.

Chapitre V

Implémentation et expérimentations

5.1 Introduction

L'objectif est de sélectionner le modèle SVM et les valeurs des hyper-paramètres les plus adaptés à la reconnaissance de caractères Arabe manuscrits. Dans ce chapitre nous présentons en premier lieu, une description de notre système de reconnaissance de caractères Arabe manuscrits hors-ligne, ensuite nous présentons les différents résultats obtenus par l'implémentation des méthodes : validation croisée, grille de recherche, recherche avec tabous et la méthode d'optimisation par essaim de particules (PSO), ainsi qu'une étude comparative entre les différents modèles SVM en termes de taux de reconnaissance (précision de validation croisée) et le temps CPU.

5.2 SVM^{light}

Nous avons utilisé l'implémentation SVM^{light} qui est un moteur SVM développé par Thorsten Joachims en 2008. C'est une implémentation open-source de référence pour les machines à support vectoriel. Elle est disponible sur http://svmlight.joachims.org/ avec documentation, exemples, et références bibliographiques. Le code est implémenté en C et devenu un des moteurs les plus utilisés actuellement. Il possède de nombreuses fonctionnalités et des caractéristiques très attractives pour l'utilisateur :

- Il permet d'utiliser des nombreux kernels (noyaux) 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 [87].
- Il fournit 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 » [88].
- 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.

5.3 Description de notre système de reconnaissance

5.3.1 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 5.1). Les images sont de format bitmap avec deux couleurs noire et blanche (binaire).

Tableau 5.1 : Lettres arabes utilisées avec ses différentes formes (96 classes).

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

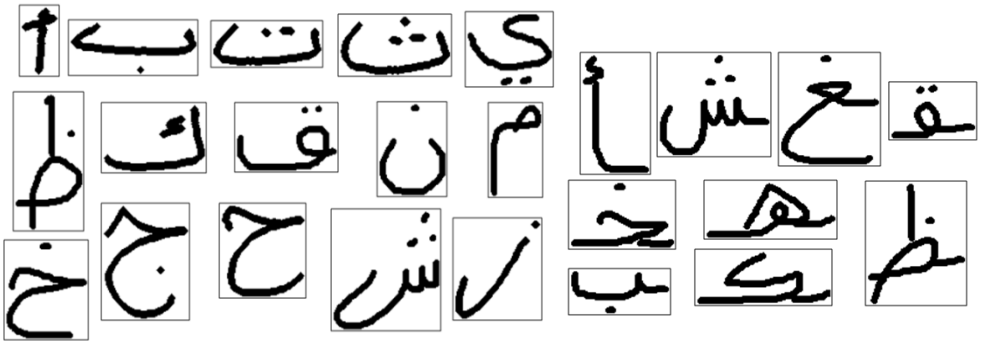


Figure 5.1 : Échantillons de notre base de données

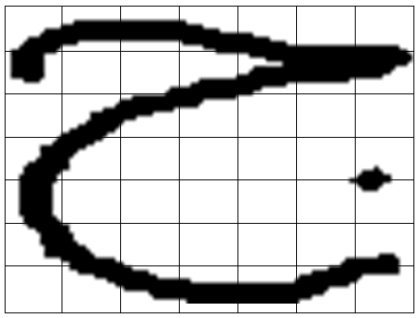
5.3.2 Normalisation

Nous avons effectué cette opération pour éliminer les conditions qui peuvent fausser les résultats, telle que la différence de taille. Après la normalisation, les images de tous les caractères se retrouvent définies dans une matrice de même taille, cela permet d'augmenter la ressemblance entre les différentes images traitées. Dans notre système, la taille proposée pour la normalisation est : Hauteur 70/ Largeur 70. Les dimensions de normalisation doivent être suffisamment élevées, pour éviter la perte d'information discriminante.

5.3.3 Extraction des primitives

C'est l'une des étapes les plus délicates et les plus importantes en OCR (Optical Character Recognition). 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.

Nous avons utilisé une technique simple et efficace qui est la technique de zonage [89], [90]. Le principe consiste à diviser l'image en 49 zones ou en blocs de 10 x 10 pixels, puis compter le nombre de pixels noirs dans chaque zone. La Figure 5.2 montre, par exemple, la lettre «jim», qui est écrite en arabe «ج» dans sa forme isolée.



28	50	51	52	38	20	17
32	0	3	33	65	69	47
0	37	49	19	0	0	0
33	34	0	0	0	0	10
59	0	0	0	0	3	13
28	42	0	0	0	3	11
0	26	54	51	50	56	26

Figure 5.2 : Extraction de caractéristiques de la lettre « jim »

L'algorithme d'extraction de caractéristiques est comme suit :

Algorithme 5.1 : Extraction de caractéristiques

```
M : Image de lettre 70 x 70;   // Entrée
V : Vecteur de caractéristiques ;   // Sortie
Compt : Entier ;  // Compteur
i, j, k, L, H : Entier ;
L←0 ;   k←0;
Tant que ( L< 70 )  faire
  H←0 ;
  Tant que ( H< 70 )  faire
     Compt ←0 ;
     Pour ( i allant de L jusqu'à L+10 )
        Pour ( j allant de H jusqu'à H+10 )
        Si pixel M[i,j] est noire alors Compt ←Compt  +1 ;
     V[k] ← Compt;
     k←k+1;
     H←H+10 ;
  L←L+10 ;
```

Dans les caractères arabes, l'existence des points diacritiques est très importante, il y a plus de la moitié des caractères qui possèdent un point qui représente dans beaucoup de cas la seule différence entre deux caractères (deux classes). Nous avons introduit un algorithme qui permet de détecter l'existence d'un point diacritique dans l'image de caractère. Cette propriété sera ajoutée au vecteur de caractéristique de cette image.

Le principe de notre algorithme est simple (voir l'algorithme 5.2 la fonction DPoint()) : nous parcourons l'image de caractère par un petit carrée (15 x 15) et si on trouve que le centre de carré est noir et tous les pixels de périmètre sont blancs, alors il y a donc un point diacritique dans l'image.

Algorithme 5.2 : Détection du point diacritique

```
M : Image de lettre 70 x 70;   // Entrée
Booléenne Fonction DPoint (M)
    Compt : Entier ;  // Compteur
    i, j, k : Entier ;
    Pour ( i allant de 0 jusqu'à 70 )
    Pour ( j allant de 0 jusqu'à 70 )
    Si pixel M[i,j] est noire alors
        Compt←0 ;
        Pour ( k allant de i-7 jusqu'à i+7 )
            Si pixel M[k,j-7] est blanc ou bien l'indice dépasse l'image alors Compt←Compt +1;
            Si pixel M[k,j+7] est blanc ou bien l'indice dépasse l'image alors Compt←Compt +1;
        Pour ( k allant de j-7 jusqu'à j+7 )
            Si pixel M[i-7,k] est blanc ou bien l'indice dépasse l'image alors Compt←Compt +1;
            Si pixel M[i+7,k] est blanc ou bien l'indice dépasse l'image alors Compt←Compt +1;
        Si Compt=60 alors return vrai ;
```

5.4 Expérimentations et Résultats

Nous avons effectué une série d'expérimentations et une recherche exhaustive afin de sélectionner le type de classification une-contre-une ou une-contre-reste, le noyau et les paramètres de noyau qui donnent les meilleurs résultats en termes de taux de prédiction et le temps CPU.

La sélection des paramètres est généralement effectuée en minimisant l'erreur de généralisation. Nous avons donc utilisé la validation croisée (section 3.3.2). Cette méthode consiste à diviser l'ensemble d'apprentissage en k sous-ensembles disjoints de même taille, puis apprendre sur les k-1sous-ensembles, et tester sur la k-ième partie, ce processus est répété k fois, la précision de la validation croisée est obtenue en calculant la moyenne de k précisions précédentes. Dans ces expérimentations, nous avons utilisé k = 5 (5-fold).

5.4.1 Expérimentation 1

Dans cette première expérimentation, la méthode grille de recherche (Section 3.3.5) a été implémentée. Étant donné que cette méthode est uniquement utilisée pour la sélection de modèles qui utilisent peu de paramètres (≤ 2), nous l'avons implémenté seulement avec le noyau RBF et le noyau Laplacien où il y a deux paramètres, C et γ; la plage de C utilisée est $[10^0, 10^1, \dots, 10^4, 10^{15}]$ et la plage de γ est $\left[10^{-15}, 10^{-14}, \dots, 10^1, 10^2\right]$, donc on a 16x18 = 288 itérations. Les résultats obtenus pour l'approche une-contre-une sont illustrés comme suit :

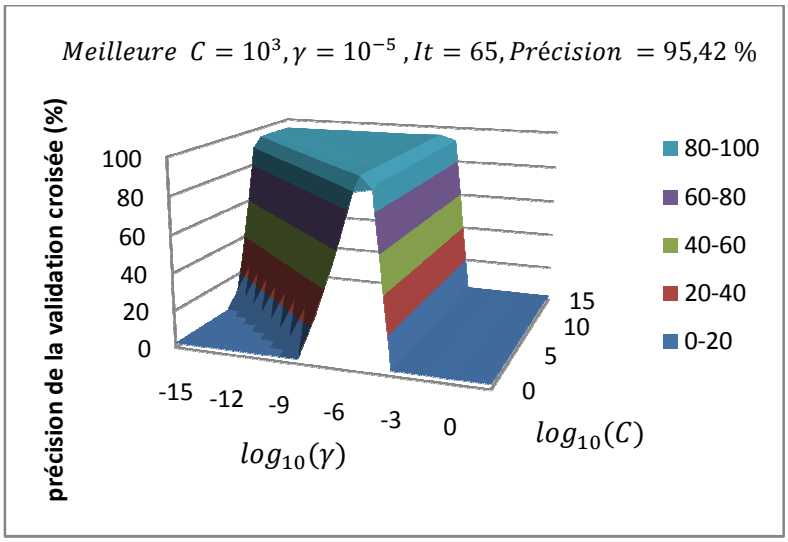


Figure 5.3 : Résultats de la méthode grille de recherche pour le noyau RBF dans l'approche une-contre-une

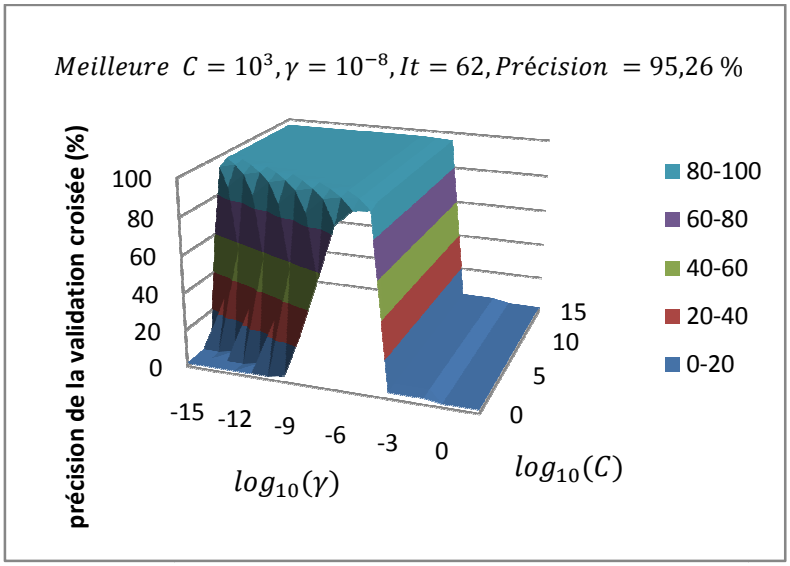


Figure 5.4 : Résultats de la méthode grille de recherche pour le noyau Laplacien dans l'approche une-contre-une

Pour l'approche une-contre-tous, les résultats obtenus sont illustrés comme suit :

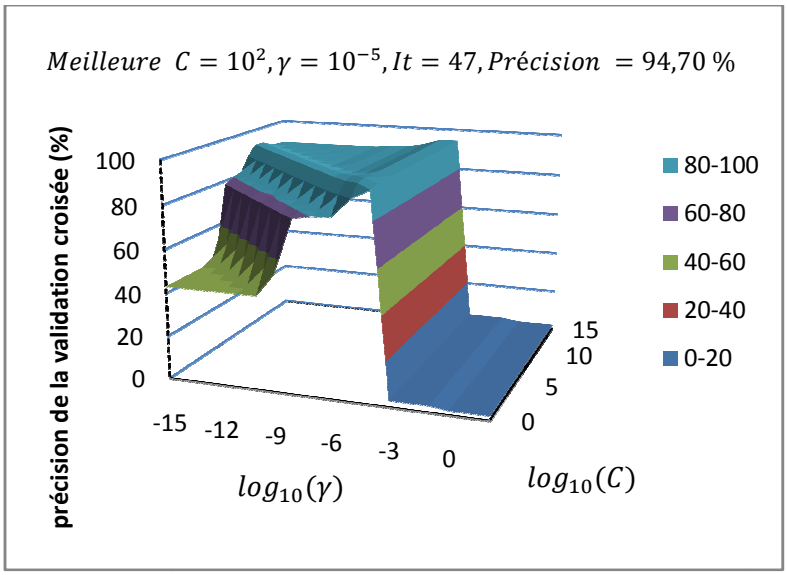


Figure 5.5 : Résultats de la méthode grille de recherche pour le noyau RBF dans l'approche une-contre-tous

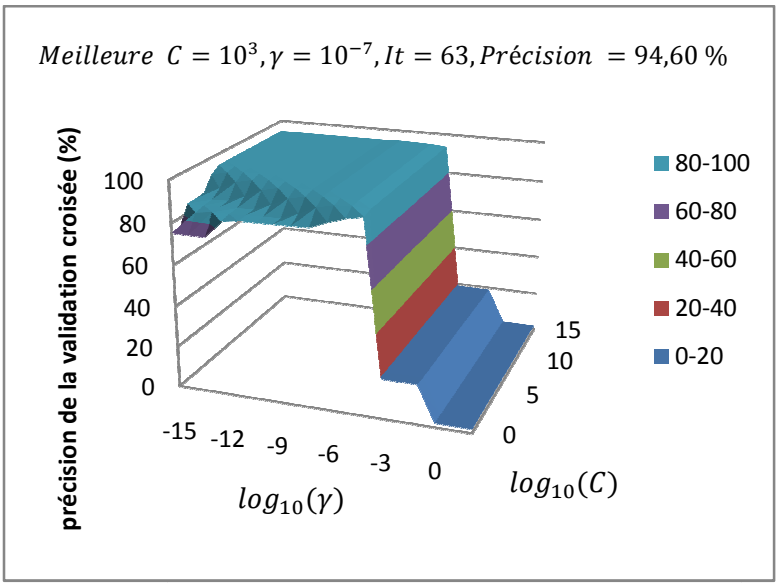


Figure 5.6 : Résultats de la méthode grille de recherche pour le noyau Laplacien dans l'approche une-contre-tous.

D'après ces résultats, la surface des bons résultats (Précision> 90%) du noyau Laplacien est grande par rapport au noyau RBF et ce dernier a donné le meilleur résultat. La même chose peut être constatée pour les deux approches SVM utilisées, la surface des bons résultats de l'approche une-contre-tous est grande comparée à l'approche une-contre-une, et les résultats de cette dernière sont les meilleurs.

Au cours de ces expériences, le temps CPU (durée) a été mesuré à chaque itération, les résultats obtenus sont présentés dans la Figure 5.7, Les résultats montrent que l'approche une-

contre-une (1x1) est plus rapide que l'autre approche (1xN) et le noyau Laplacien est plus rapide que le RBF. L'approche une-contre-tous devient très lente lorsque le paramètre γ prend de grandes valeurs telles que 1, 10 et 100. Ceci est évident sur la Figure 5.7.

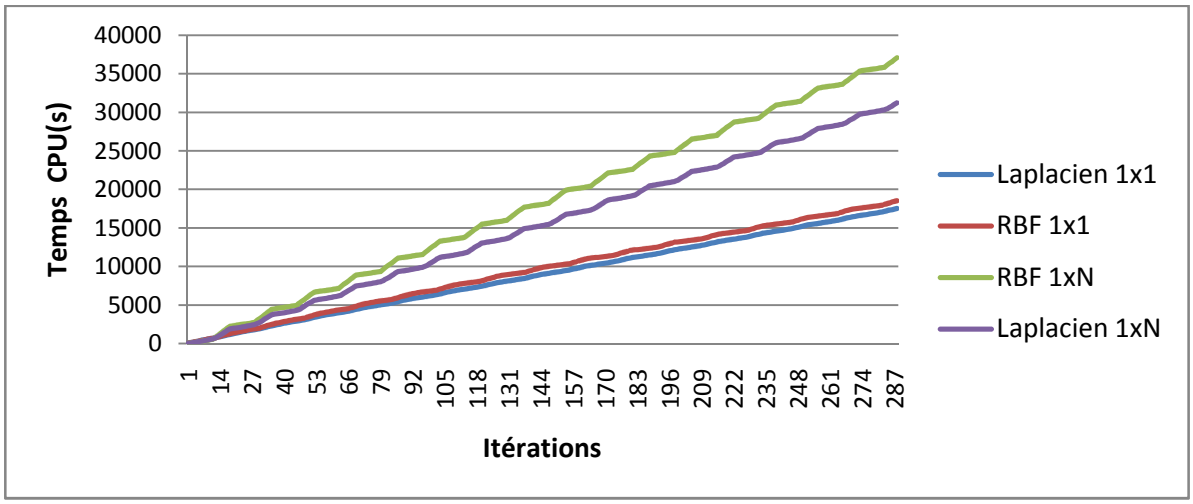


Figure 5.7 : Durée en secondes de chaque modèle SVM

5.4.2 Expérimentation2

Nous avons implémenté notre algorithme de sélection automatique de modèle SVM basé sur la recherche tabou avec un ajustement adaptatif de la taille de pas et la liste taboue. La validation croisée est utilisée pour l'estimation de la capacité de généralisation (voir l'algorithme 4.3).

Un des avantages de la recherche taboue est qu'elle est facile à étendre à des noyaux comportant plusieurs paramètres libres. Dans cette étude nous avons utilisé cinq types de noyaux qui sont : le noyau polynomial, Gaussien (RBF), Laplacien, Sigmoïde et Linéaire. Les résultats obtenus sont présentés dans ce qui suit :

- **SVM une-contre-reste**

Tableau 5.2 : Résultats obtenus pour chaque noyau dans l'approche une-contre-reste

	Itérations	C	γ	D	$Coef$	Précision (%)
RBF	65	1000	1,77E-05	/	/	94,44
Sigmoïde	61	1059	4,35E-06	/	3	93,68
Laplacien	58	100	3,60E-05	/	/	94,09
Polynomial	60	400	4,30E-05	4	2	93,37
Linéaire	15	10	/	/	/	87,26

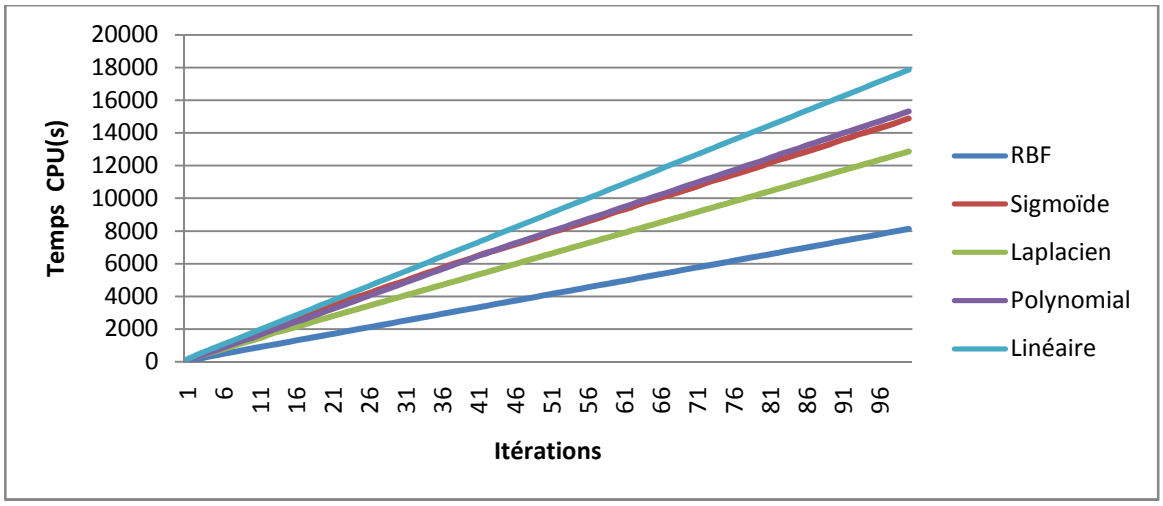


Figure 5.8 : Temps de calcul en secondes de chaque noyau dans l'approche une-contre-reste

- **SVM une-contre-une**

Tableau 5.3 : Résultats obtenus pour chaque noyau dans l'approche une-contre-une

	Itérations	C	γ	D	$Coef$	Précision (%)
RBF	71	11	1,92E-05	/	/	95,57
Sigmoïde	76	609	8,83E-07	/	1	95,05
Laplacien	69	840	3,49E-05	/	/	94,40
Polynomial	57	100	2,64E-04	2	11	94,93
Linéaire	21	10	/	/	/	94,83

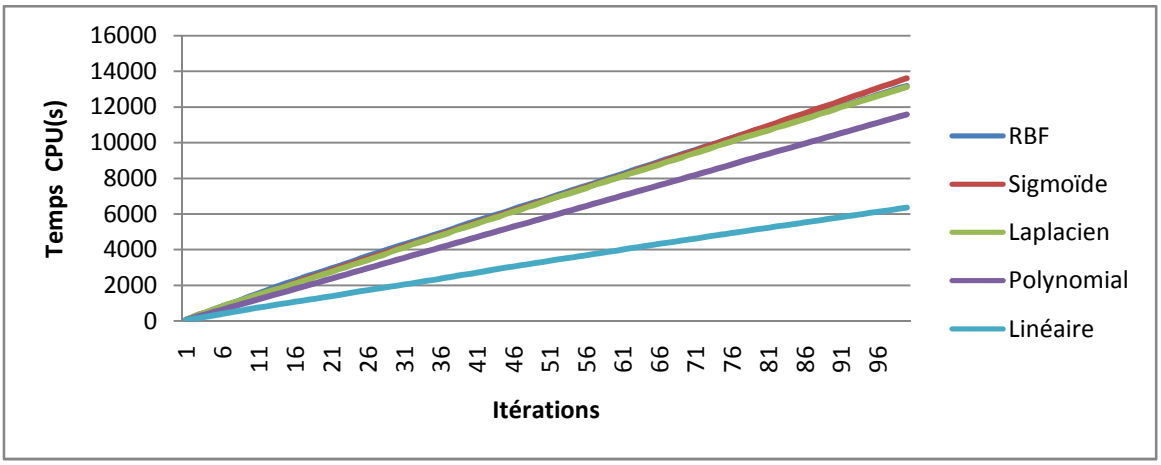


Figure 5.9 : Temps de calcul en secondes de chaque noyau dans l'approche une-contre-une

Les expérimentations réalisées montrent que la sélection de modèle par notre algorithme basé sur la recherche taboue est efficace, mais il nécessite beaucoup de temps comme la procédure grille de recherche, Ceci est dû principalement à la lenteur de la procédure de validation croisée et que dans chaque itération il faut exécuter cette procédure 4 fois pour choisir un mouvement parmi les quatre mouvements existants du noyau RBF par exemple.

Notre algorithme peut produire le meilleur résultat lorsqu'il tombe dans une région concave contenant le minimum global, cela grâce à la réduction automatique de la taille de pas, comme le cas de noyau RBF et Sigmoïde dans l'approche une-contre-une.

Nous remarquons que le noyau linéaire prend beaucoup de temps dans l'approche une-contre-reste bien qu'il y a un seul paramètre à sélectionner, cela parce que l'application SVM light (le moteur SVM utilisé) a besoin beaucoup de temps pour trouver les coefficients de Lagrange optimaux (α^*) pour les valeurs extrêmes de paramètres, plus particulièrement dans cette approche.

5.4.3 Expérimentation3

Dans cette troisième expérience, la sélection des paramètres de modèle SVM ont été effectué par l'utilisation de la méthode PSO, en particulier le modèle gbest [91]. Cinq particules sont utilisées avec les valeurs de coefficient: $\omega = 0{,}7298$ et $C_1 = C_2 = 1{,}4962$, afin d'assurer la convergence [92].

Chaque particule encode les paramètres du modèle SVM dans leur position. Le nombre de ces paramètres varie en fonction du type de noyau. Les valeurs considérées pour chaque paramètre sont : $C \in [10^0, 10^{15}]$, $\gamma \in [10^{-15}, 10^2]$, $coef \in [-10, 10^2]$ et $d \in [1{,}10]$. Si une particule dépasse la limite de l'espace de recherche, alors cette particule sera repositionnée à la limite et sa vitesse est définie à zéro. Les résultats obtenus pour l'approche une-contre-une sont exprimés comme suit :

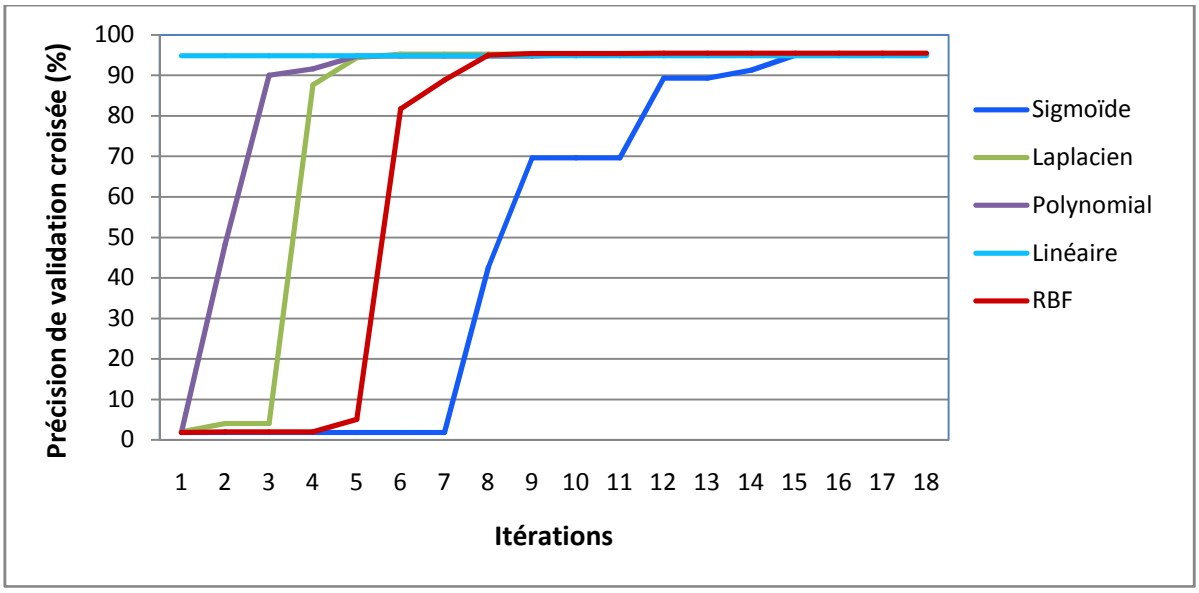


Figure 5.10 : Comparaison des résultats des différents noyaux pour l'approche une-contre-une

Tableau 5.4 : Meilleurs résultats obtenus pour chaque noyau pour l'approche une-contre-une

	Itérations	C	γ	D	$Coef$	Précision (%)
Gaussien (RBF)	14	117405	1,75E-05	/	/	95,49
Sigmoïde	17	65975	9,08E-08	/	-1,97	95,18
Laplacien	12	82771	1,10E-11	/	/	95,26
Polynomial	12	172320	1,02E-10	2	11	95,20
Linéaire	1	13692	/	/	/	94,83

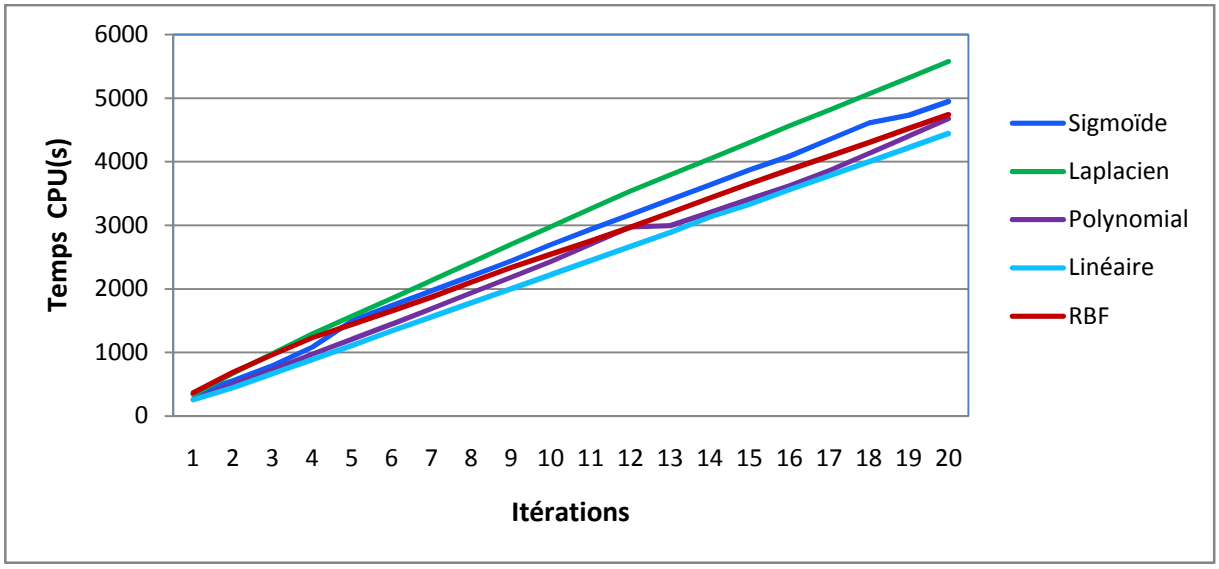


Figure 5.11 : Durée en secondes de chaque noyau pour l'approche une-contre-une

Pour l'approche une-contre-reste, les résultats obtenus sont exprimés comme suit :

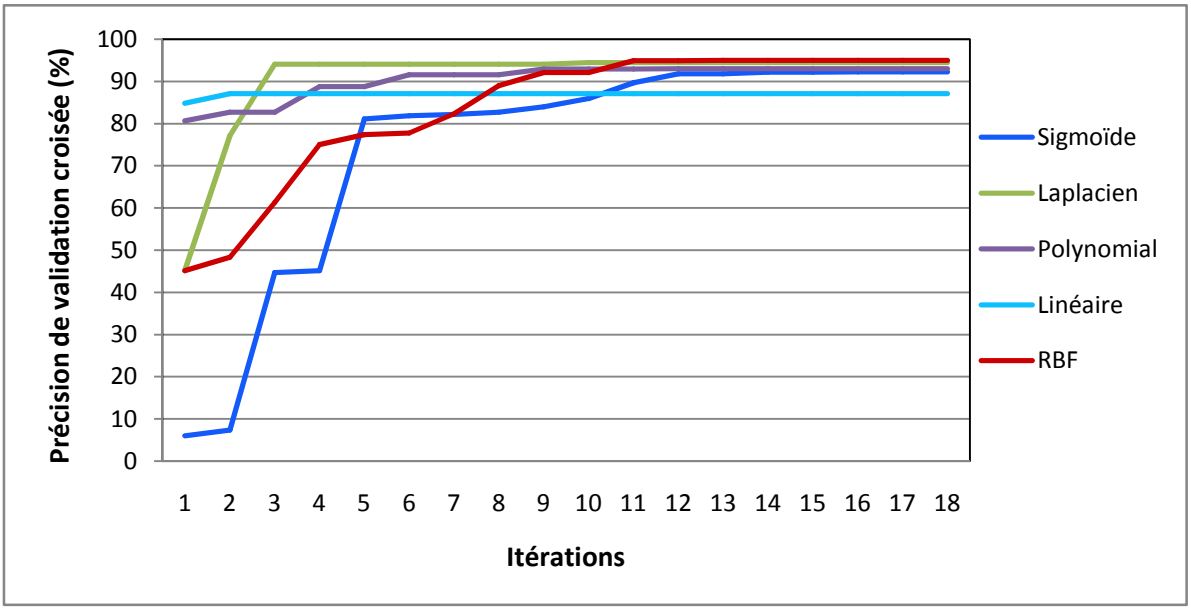


Figure 5.12 : Comparaison des résultats des différents noyaux pour l'approche une-contre-reste

Tableau 5.5 : Meilleurs résultats obtenus pour chaque noyau pour l'approche une-contre-reste

	Itérations	C	γ	D	Coef	Précision (%)
Gaussien (RBF)	15	46651	4,27E-05	/	/	94,99
Sigmoïde	16	85617	1,20E-07	/	-2,03	92,24
Laplacien	10	67577	3,40E-08	/	/	94,48
Polynomial	12	13863	4,70E-03	2	1	93,06
Linéaire	2	2310	/	/	/	87,12

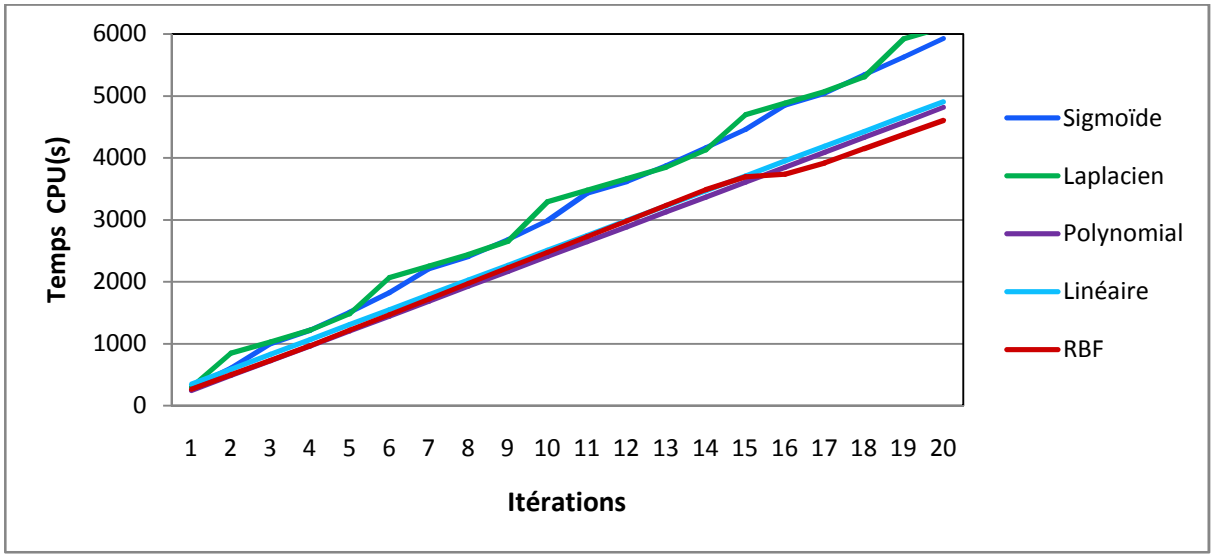


Figure 5.13 : Durée en secondes de chaque noyau pour l'approche une-contre-reste

Selon les résultats obtenus, l'approche SVM une-contre-une conduit à des résultats meilleurs que l'approche SVM une-contre-reste. Le noyau linéaire converge rapidement vers l'optimum, car dans ce cas le modèle SVM n'a qu'un seul paramètre (C), et la valeur de ce paramètre n'affecte pas significativement le comportement du modèle. Par contre, le noyau sigmoïde est le plus lent car dans ce cas, il existe trois paramètres et le modèle est très sensible au changement de ces paramètres. Le meilleur résultat en termes de taux de reconnaissance est pour le noyau RBF dans l'approche une-contre-une.

Enfin, il convient de noter que la méthode grille de recherche est une technique efficace pour fournir une vue globale sur l'espace de recherche (permet d'extraire les régions prometteuses). Pour trouver de bons résultats, il est donc nécessaire d'utiliser une recherche locale ou de lancer la méthode grille de recherche une seconde fois dans la région prometteuse. La recherche Tabou est plus adaptée à la recherche dans une région prometteuse contenant le minimum global pour trouver des résultats plus proches à l'optimum. Ces deux méthodes nécessitent beaucoup de temps pour explorer l'espace de recherche. Par contre, la méthode d'optimisation par essaime de particules (PSO) est la plus efficace car elle produit de bons résultats en un temps réduit.

5.5 Conclusion

Sur la base des résultats expérimentaux obtenus, l'utilisation de la méthode SVM pour la reconnaissance des caractères arabes est fortement recommandée en raison de sa capacité élevée, pour classifier des données de grande dimension, même s'il existe un grand nombre de classes (dans notre cas, 96 classes). Il est également important de noter que le noyau RBF est le plus approprié pour la reconnaissance des caractères arabes manuscrits. En effet, avec ce noyau nous obtenons de meilleurs résultats que les autres noyaux avec une précision de validation croisée de 95,57% (SVM une-contre-une, $\gamma = 1{,}92\text{E} - 05$ et $C = 11$).

Chapitre VI

Contribution à la segmentation de l'écriture arabe manuscrite

6.1 Introduction

La reconnaissance de l'écriture manuscrite est plus complexe que la reconnaissance du texte imprimé en raison de ses extrêmes variabilités, variabilité des formes, l'espacement entre les mots et les caractères, fluctuations des lignes…etc.

La segmentation est une étape très importante dans le processus de reconnaissance ; cette étape est simple dans le cas du texte latin imprimé, mais très difficile dans le cas de l'écriture cursive (écriture arabe). La complexité de la morphologie de l'écriture arabe et sa cursivité rendent plus difficile à segmenter les mots en caractères.

Plusieurs travaux ont été réalisés par les chercheurs pour la reconnaissance du mot entier (approche globale) sans segmentation [93], et d'autres supposent que les caractères sont déjà segmentés pour éviter l'étape de segmentation [94], [95]. En fait, dans ce chapitre nous proposons un nouvel algorithme de segmentation qui effectue une analyse approfondie de l'histogramme de projection verticale, afin d'extraire les points de segmentation.

6.2 Approche proposée

Pour segmenter un mot en caractères, notre approche consiste en plusieurs étapes. Tout d'abord, nous générons l'histogramme de projection verticale. Ensuite, le mot est segmenté en sous-mots, suivi de l'extraction des points de segmentation. Après cela, plusieurs opérations sont effectuées sur ces points de segmentation pour améliorer leur emplacement ou supprimer certains d'autres. Dans ce qui suit, nous présentons une description détaillée de chaque étape de notre processus de segmentation.

6.2.1 Histogramme de projection verticale

La projection verticale représente le nombre de pixels noirs dans chaque colonne de l'image, définie par le vecteur V_j de taille N comme suit :

$$V_j = \sum_j P(i,j) \tag{6.1}$$

Où $P(i,j)$ est un pixel de l'image binaire du mot et vaut 0 ou 1, i et j font référence aux index de la ligne et de la colonne.

6.2.2 Diviser le mot en parties (sous-mots)

Chaque mot peut être composé d'une ou plusieurs parties. Nous avons utilisé le vecteur de projection verticale V pour déterminer les différentes parties d'un mot. L'idée de base est que lorsqu'on trouve une valeur ou une suite de valeurs égales à zéro dans le vecteur de projection verticale ($V_j = 0$), cela signifie que le mot peut être divisé en deux sous-mots dans cette position, voir l'exemple de la Figure 6.1.

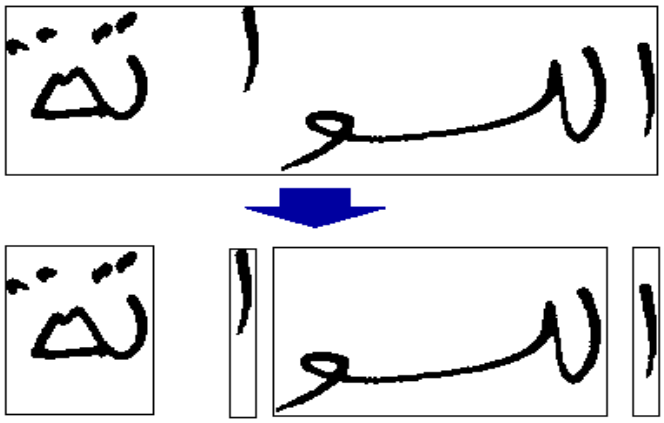


Figure 6.1 : Division du mot en parties (sous-mots)

6.2.3 Algorithme d'extraction des points de segmentation

Nous avons proposé un nouvel algorithme qui utilise le vecteur de projection verticale pour extraire les points de segmentation. L'algorithme a trois paramètres : la taille de bloc (Tb), la taille de pas (Tp) et le seuil (S). La valeur de paramètre Tp doit être inférieure ou égale à la valeur Tb. Si les deux paramètres sont égaux, il n'y a pas de chevauchement entre le bloc et le bloc suivant, comme c'est illustré dans la Figure 6.2.

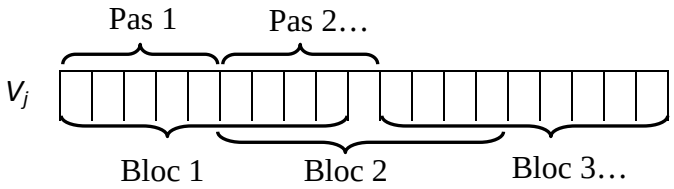


Figure 6.2 : Exemple de vecteur de projection verticale avec Tb = 9 et Tp = 5.

Le concept de notre algorithme est le suivant : chaque fois que la somme des valeurs du bloc augmente, aucun point de segmentation n'est généré. S'il existe une diminution significative entre la somme des valeurs du bloc actuel et le bloc précédent (supérieure au seuil), il existe un point de segmentation. Le pseudo-code suivant décrit notre algorithme.

Algorithme 6.1 : Extraction des points de segmentation

```
Entrée : histogramme de projection verticale : V []
Sortie : vecteur des points de segmentation : Seg []
Spécifier les valeurs des paramètres: Tb,  Tp et  S
SumA = 0; //la somme des valeurs de bloc actuel.
SumP = 0; //la somme des valeurs de bloc précédent.
i = Tp; j = 0; k = 0; // i, j et k sont des entiers
Pour (k = 0; k <= Tb; k ++)
    SumP = SumP + V [k];
Fin pour
Tant que (i <Largeur de l'image −Tb)
    Pour (k = i; k <= i + Tb; k++)
        SumA = SumA + V [k];
    Fin pour
    Si (SumP −  SumA >S) alors
        Seg [j] = i + Tb /2 ; //Gardez l'indice de point de segmentation
        j = j + 1;
        SumP = 0;
    Sinon
        SumP = SumA;
    Fin si
    i = i + Tp;
    SumA = 0;
Fin tant que
```

Dans la Figure 6.3, nous présentons un exemple d'application de notre algorithme, les lignes verticales rouges représentent la position des points de segmentation.

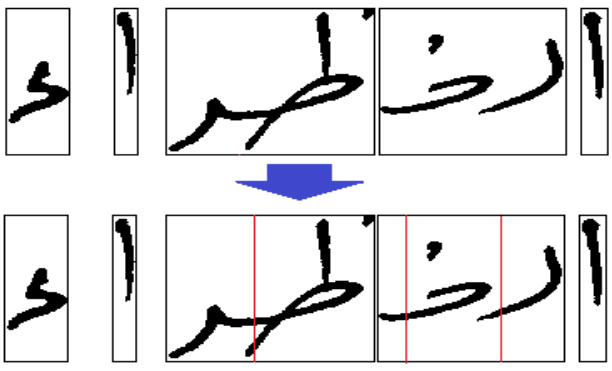


Figure 6.3 : Exemple d'extraction des points de segmentation.

6.2.4 Amélioration de la position des points de segmentation

Cette opération consiste à effectuer une recherche locale autour de chaque point de segmentation afin de trouver sa meilleure position. Le point de segmentation final est le point le plus proche, ayant la plus petite valeur dans le vecteur de projection verticale, autour du point initial, comme c'est illustré dans la Figure 6.4.

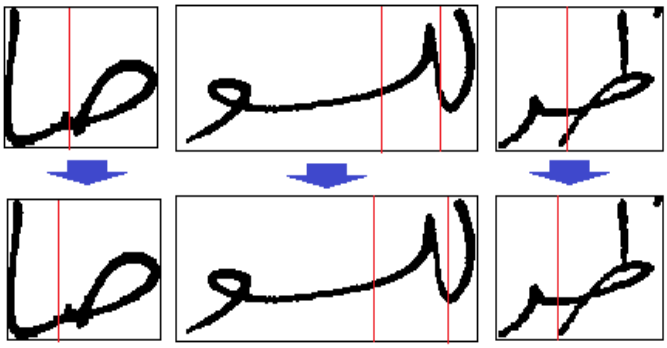


Figure 6.4 : Amélioration des points de segmentation

6.2.5 Détection de la ligne de base

Nous avons effectué la projection horizontale pour déterminer la ligne de base. Cette opération supprime les points de segmentation les plus loin de la ligne de base, comme c'est indiqué dans la Figure 6.5.

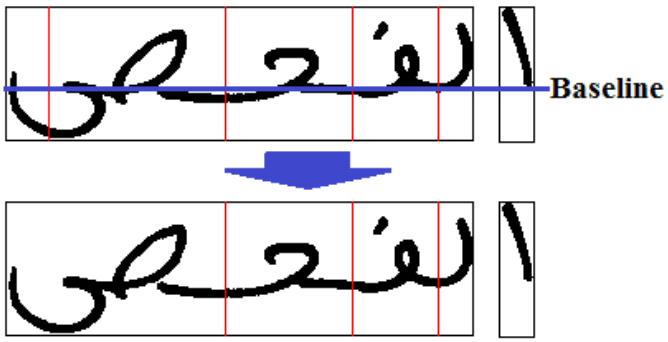


Figure 6.5: Suppression du point de segmentation

6.2.6 Nombre de transitions

On parcourt les pixels de chaque colonne d'un point de segmentation afin de déterminer le nombre de fois où la valeur du pixel est passée de 0 à 1 ou de 1 à 0. Le nombre de transitions correspond au nombre total de fois où l'état du pixel est changé. Un point de

segmentation est supprimé si le nombre de transitions est supérieur à deux (voir l'exemple de la Figure 6.6).

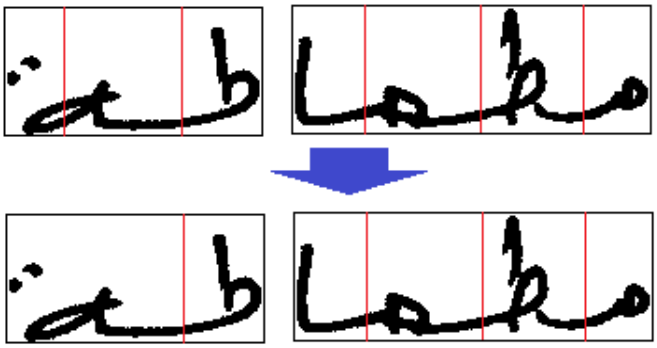


Figure 6.6 : Exemple de nombre de transitions supérieur à deux.

6.3 Résultats expérimentaux

Pour évaluer notre approche, nous avons utilisé la base de données IFN / ENIT [100]. Cette base a été développée en 2002 par l'IFN (Institut des technologies de la communication en Allemagne) et ENIT (École nationale d'ingénieurs de Tunisie). Il s'agit d'une base d'images de noms de villes tunisiennes collectée à l'aide d'une contribution de 411scripteurs. Chacun d'entre eux a écrit 60 noms de villes avec leurs codes postaux correspondants.

L'algorithme développé a été testé sur un ensemble de mots de la base IFN / ENIT. Les mots ont été choisis avec soin pour couvrir toutes les formes de caractères arabes. Les résultats obtenus montrent que 89,5% des points de segmentation ont été extraits correctement, ce qui est un taux appréciable par rapport à l'état de l'art actuel.

Notre algorithme de segmentation a trois paramètres et leurs valeurs affectent les résultats. Dans ce qui suit, nous présentons un ensemble d'expérimentations pour déterminer les valeurs appropriées de ces paramètres.

Dans la première expérimentation, nous avons fixé le seuil S à 50 et les deux autres paramètres sont variés, les résultats obtenus sont présentés dans la Figure 6.7.

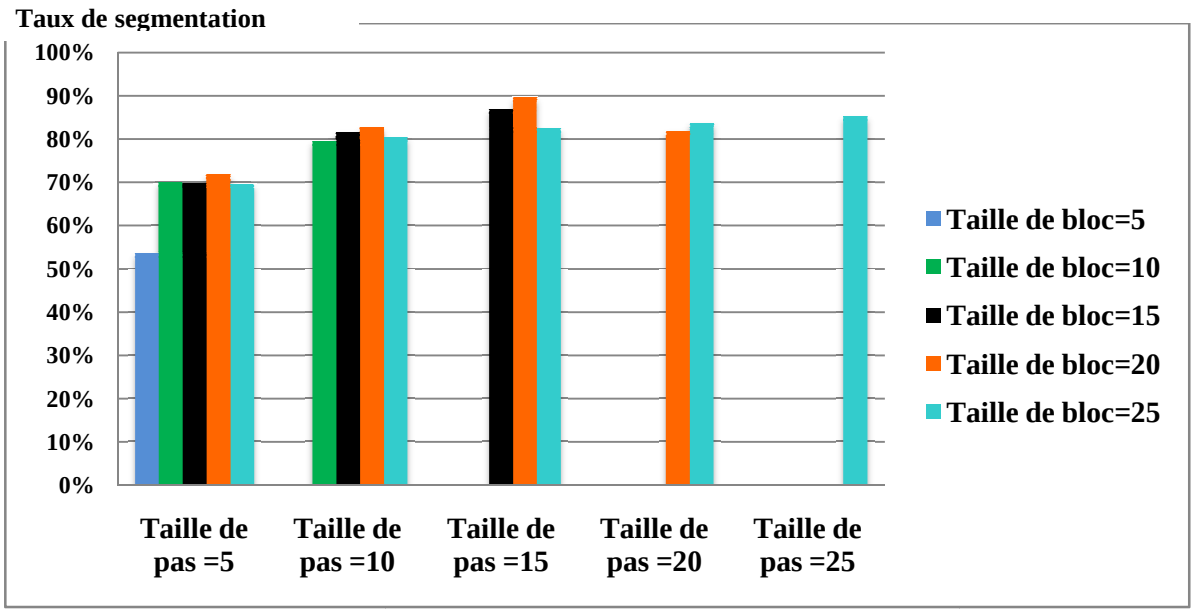


Figure 6.7 : Résultats de segmentation avec le seuil S fixé à 50

Le meilleur résultat obtenu est 89,5% avec Tb = 20 et Tp = 15. Les erreurs de segmentation sont dues au chevauchement ou lorsqu'on tombe dans la sur-segmentation ou la sous-segmentation. Si la valeur des paramètres Tb et Tp est petite, l'erreur de sur-segmentation augmente et vice versa, comme indiqué dans la Figure 6.8.

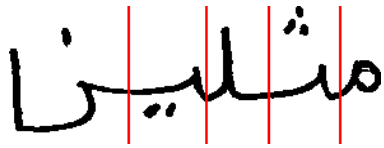
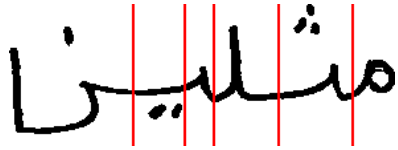
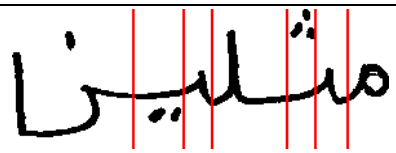
S=50 Tb=20 Tp=15	
S=50 Tb=15 Tp=10	
S=50 Tb=10 Tp=10	

Figure 6.8 : Exemple de sur-segmentation

Dans la deuxième expérimentation, on fixe la taille de bloc et la taille de pas à Tb = 20 et à Tp = 15, et nous varions le seuil S. Les résultats obtenus sont présentés dans la Figure 6.9.

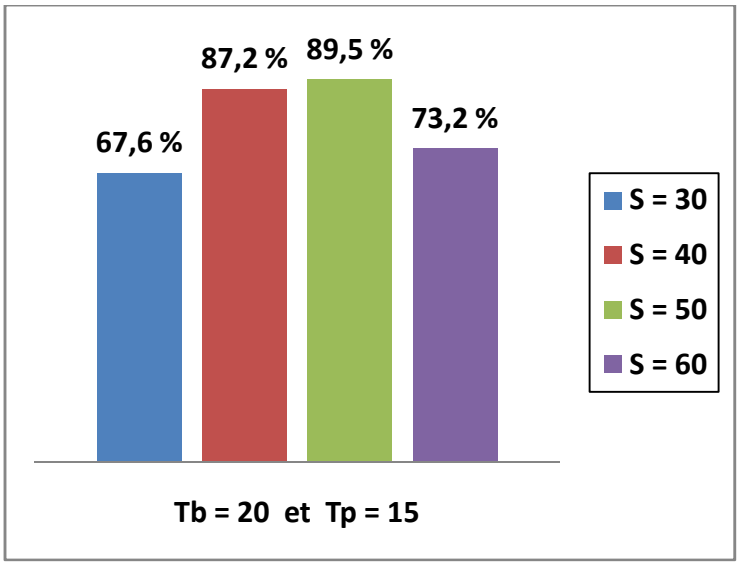


Figure 6.9 : Résultats de la segmentation avec Tb = 20 et Tp = 15.

Les résultats montrent que la meilleure valeur du seuil est 50. L'augmentation du seuil plus de 50 provoque souvent une sous-segmentation et vice-versa, comme le montre la Figure 6.10.

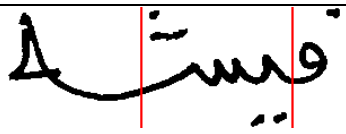
S=50 Tb=20 Tp=15	
S=65 Tb=20 Tp=15	
S=80 Tb=20 Tp=15	

Figure 6.10 : Exemple de sous-segmentation

Enfin, nous présentons dans la Figure 6.11 une comparaison entre le résultat obtenu et les résultats d'autres travaux de la littérature.

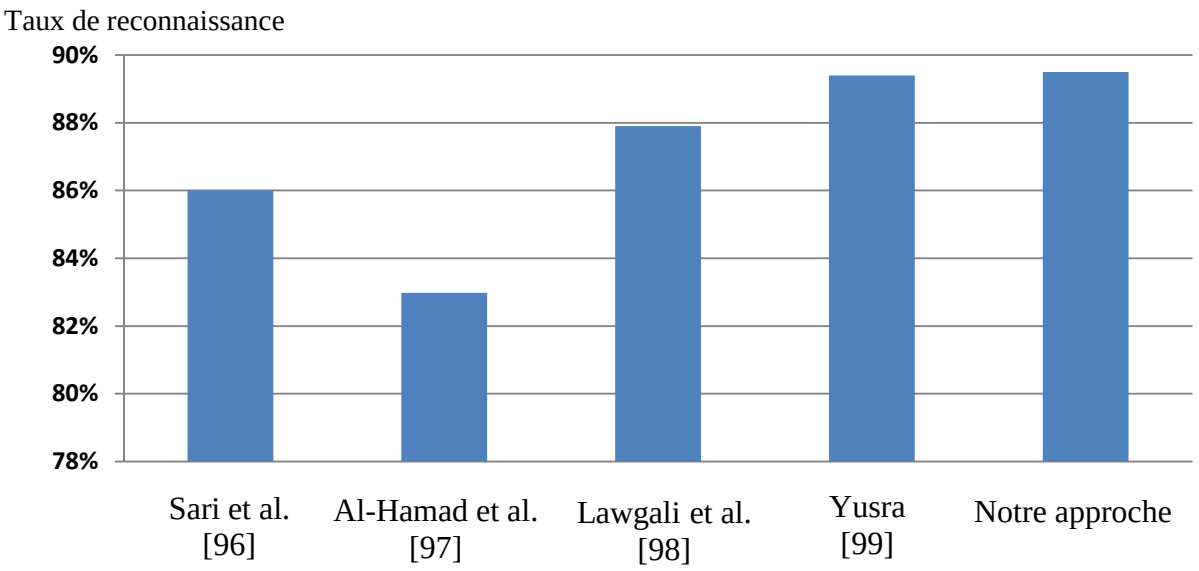


Figure 6.11 : Comparaison avec d'autres travaux de la littérature.

6.4 Conclusion

La reconnaissance des caractères après le processus de segmentation implique plus de défis, car la segmentation est le problème le plus sérieux dans le développement du système de reconnaissance d'écriture arabe.

La tâche de segmentation d'un mot en caractères est plus difficile que la segmentation qui vise à extraire des lignes et des sous-mots. Dans ce chapitre, un nouvel algorithme de segmentation est proposé basé sur l'histogramme de projection verticale.

Notre approche de segmentation donne des résultats encourageants, avec une précision de 89,5%. Le problème majeur qui rend cette tâche cruciale est le problème de chevauchement des caractères.

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, de 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 un taux d'erreurs assez bas.

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 à la théorie de l'apprentissage. Cependant, des limitations restent pour les séparateurs à vaste marge, à savoir la sélection des paramètres libres par des méthodes déterministes et fiables.

L'utilisation des techniques méta-heuristiques est devenue un enjeu important, les différentes caractéristiques de ces techniques (voisinage, mémoires, listes de candidats, populations, fonction objectif, intensification, diversification etc.) sont exploitées pour former de nouveaux procédés, dont l'objectif est de bien explorer l'espace de solutions, afin d'obtenir de très bonnes solutions.

Dans le cadre de cette thèse, nous nous sommes intéressés à l'optimisation des hyper-paramètres du SVM par la sélection automatique du meilleur modèle. Dans ce sens, nous avons utilisé des techniques méta-heuristiques pour la sélection de modèle SVM, en adaptant la méthode de recherche taboue et l'optimisation par essaim de particules. Aussi pour des fins de comparaison, nous avons implémenté la méthode grille de recherche. Un système complet de reconnaissance des caractères arabes manuscrits a été réalisé, avec une étude comparative des différents modèles de SVM en termes de taux de prédiction et de temps CPU.

L'apport de cette thèse a concerné également le développement d'une approche de segmentation des mots manuscrits en caractères. Cette approche est basée sur un algorithme faisant une analyse efficace de l'histogramme de projection verticale.

Les résultats obtenus montrent que la méthode PSO est la plus efficace pour la sélection de modèle SVM en comparaison avec les deux autres méthodes (recherche tabou et grille de recherche), et le noyau RBF a donné le meilleur résultat par rapport aux autres noyaux dans l'approche une-contre-une, avec une précision de 95,49 % ($\gamma = 1{,}75\mathrm{E}-05$ et $\mathrm{C} = 11$). Ainsi, pour la segmentation, des résultats encourageants ont été trouvés, avec une précision de 89,5%; ce qui est un taux appréciable par rapport à la littérature.

Par conséquent, sur la base des résultats prometteurs obtenus, nous proposons comme perspectives, les travaux futurs suivants : L'amélioration de notre algorithme de segmentation par de nouvelle recherche pour la résolution du problème complexe de chevauchent de caractères, et l'utilisation de la base d'image issue de la segmentation du corpus IFN / ENIT dans notre système de reconnaissance de caractères.

Développement d'une hybridation entre la méthode PSO et la recherche tabou, vu que d'après les expérimentations, la méthode PSO est plus efficace dans la diversification de la recherche, car on a plusieurs particules et chaque particule a sa propre position, cela couvre une grande partie de l'espace de recherche; par contre la recherche tabou est plus adaptée à l'intensification de la recherche, ce qui permettra de trouver la meilleure solution dans une région prometteuse.

L'utilisation de l'approche locale au lieu de l'approche globale [30] en vue de l'optimisation de chaque classifieur SVM binaire pour toutes les combinaisons de classes, ce qui permettra à chaque SVM binaire d'avoir ses propres valeurs d'hyper-paramètres.

Bibliographie

[1] Vapnik, V. "The Nature of Statistical Learning Theory". Springer-Verlag, 1998.

[2] Schahrazed H. "Segmentation de textes en caracteres pour la reconnaissance optique de l'écriture arabe", Mémoire, Université El-Hadj Lakhdar Batna, 2007.

[3] Slimane F., Kanoun S., Hennebert J., Alimi A. M., Ingold R. "Modèles de markov cachés et modèle de longueur pour la reconnaissance de l'ecriture arabe à basse résolution", Proceedings de Manifestation des Jeunes Chercheurs en Sciences et Technologies de l'Information et de la Communication (MajecSTIC), Avignon-France ,2009.

[4] Leila C., "Combinaison de classifieurs pour la reconnaissance de mots arabes manuscrits", Thèse de doctorat université Mentouri Constantine, 2013.

[5] Mezghani N., Mitiche A., Cheriet M., "A new representation of shape and its use for high performance in online Arabic character recognition by an associative memory", International Journal on Document Analysis and Recognition, vol. 7, n°. 4, p. 201–210, 2005.

[6] Slimane F., Ingold R., Alimi A.M., Hennebert J., "Duration models for Arabic text recognition using hidden Markov models", Proceedings of IEEE International Conference on Computational Intelligence for Modelling, Control and Automation, Vienne–Austria, IEEE, pp. 838-843, 2008.

[7] Kessentini Y., "Modèles de Markov multi-flux pour la reconnaissance de l'écriture manuscrite multi-scripts", Thèse de Doctorat, Université de Rouen, France, 2009.

[8] Alma'adeed S., "Recognition of off-line handwritten Arabic words", Thèse de Doctorat, Université de Nottingham, England, 2004.

[9] Mustapha K., "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.

[10] Belaïd A., "Reconnaissance de l'Écriture et Analyse de Documents : Numérisation, Prétraitements", LORIA-CNRS Campus scientifique B.P. 239 54506 Vandoeuvre-Lès-nancy, 2006.

[11] Fahmy M.M.M., Al Ali S., "Automatic Recognition Of Handwritten Arabic Characters Using Their Geometrical Feature", Studies in informatics and control journal (SIC journal), vol. 10, No 2, 2001.

[12] Bow S.T. "Pattern Recognition and Image Preprocessing", Marcel Dekker, United States of America (USA), 2002.

[13] Farès M.,"Contributions à la reconnaissance de l'écriture arabe manuscrite", Thèse de doctorat de l'université Paris Descartes, 2008.

[14] Belaïd A., "Analyse et reconnaissance des documents", http://www.loria.fr/~abelaid/Teaching (support de cours) Université de Lorraine, 2006.

[15] Souici Meslati L., "Reconnaissance de formes et écriture arabe manuscrite", support de cours, université Badji Mokhtar d'Annaba, 2008.

[16] Parvez M. T., Mahmoud S. A. "Offline arabic handwritten text recognition: A survey", ACM Comput. Surv. no. 2, p. 23:1–23:35, 2013.

[17] Chevalier S., Lemaitre M., Geoffrois E., Grosicki E., Preteux F. "Etude de primitives spectrales pour la reconnaissance de caractères manuscrits dans le cadre d'une approche markovienne 2d", Actes 15ème Congrès Francophone AFRIF-AFIA Reconnaissance des Formes et Intelligence Artificielle (RFIA'2006), 2006.

[18] Faouzi Z., "Technique basée puzzle/SVM pour l'amélioration de la reconnaissance du texte arabe manuscrit", Thèse de doctorat de l'université Mohamed Khider – BISKRA, 2017.

[19] Ramel J. "Lecture automatique de partitions musicales", Mémoire de DEA ingénierie informatique, LISPI-Equipe de Reconnaissance des Formes et Diagnostics, Université Lyon 1,1993.

[20] Kessentini Y., Paquet T., Burger T. "Comparaison des méthodes probabilistes et évidentielles de fusion de classifieurs pour la reconnaissance de mots manuscrits", CIFED, 2010.

[21] Werbos P. J., Beyond regression: "New Tools for Prediction and Analysis in the Behavioral Sciences". Ph.D. Thesis, Harvard University, Cambridge, 1974.

[22] Rumelhart D. E., Hinton G. E., Williams R. J. "Learning internal representations by error backpropagation". Parallel distributed processing: explorations in the microstructure of cognition, vol. 1 Pages 318-362, 1986.

[23] LeCun Y., Boser B., Denker J. S., Henderson D., Howard R. E., Hubbard W., Jackel L. D. "Backpropagation applied to handwritten zip code recognition". Neural Computation, 1(4):541–551, Winter 1989.

[24] Antoine C., Laurent M., "Apprentissage artificiel : concepts et algorithmes", Préface de Tom Mitchell. Édition EYROLLES deuxième tirage 2003.

[25] Gherghout Y., Souici-Meslati L. “Reconnaissance de caractères arabes manuscrits par les séparateurs à vastes marges (svm)”, Proceedings of the International Conference on Applied Informatics, vol. 3, p. 220–223, 2009.

[26] Luc M., "Reconnaissance de l’écriture manuscrite avec des réseaux récurrents", Thèse de Doctorat de l’Université de Rouen, 2015.

[27] Vapnik V.N. "Estimation of Dependences Based on Empirical Data". springer Verlag, Berlin, 1982.

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

[29] Cortes C., Vapnik V. "Support-vector networks". Machine Learning, 20(3):273 297, 1995.

[30] Ayat N., "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.

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

[32] Milgram J., "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". Thèse ÉTS Montréal, le 29 juin 2007.

[33] Steve R. G., "Support Vector Machines for Classification and Regression" University of Southampton, 10 May 1998.

[34] Bottou L., Chih-jen L. "Support Vector Machine Solvers, in Large Scale Kernel Machines". MIT Press. 2007.

[35] Mercier L., "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, Grenoble, 11 février 2010.

[36] Anis B. I., "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.

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

[38] Abe S. "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.

[39] Knerr S., Personnaz L., Dreyfus G. "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.

[40] Kreßel U. H. G. "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.

[41] Friedman J. H. "Another approach to Polychotomous classification". Technical report, Department of Statistics, Stanford University, 1996.

[42] Platt J. C., Cristianini N., Shawe-Taylor J. "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.

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

[44] Kijsirikul B., Ussivakul N., Meknavin S. "Adaptive directed acyclic graphs for multiclass classification". In PRICAI 2002, pages 158-168, 2002.

[45] Phetkaew T., Kijsirikul B., Rivepiboon W. "Reordering adaptive directed acyclic graphs for multiclass support vector machines". In Proceedings of the Third International Conference on Intelligent Technologies, 2002.

[46] Phetkaew T., Kijsirikul B., Rivepiboon W. "Multiclass Classification of Support Vector Machines by Reordering Adaptive Directed Acyclic Graph". In International Workshop on Intelligent Systems, 2003.

[47] Munteanu D. "A quick survey of text categorization algorithms". Annals of University Dunarea de Jos 1/ of Galati, 1, 2007.

[48] Robert H., Trevor, Tibshirani. "Stanford statistical learning online course SVM lecture. Example and comparison with logistic regression". Web Video, Nov 2013. Accesed 15 Sep 2014.

[49] Chang Y. "Feature ranking using linear svm. Causation and Prediction Challenge" Challenges in Machine Learning, 2:47, 2008.

[50] Joachims T. "Text categorization with support vector machines: Learning with any relevant features". Springer, 1998.

[51] Abdelhamid D., "Utilisation des méthodes Support Vector Machine (SVM) dans l'analyse des bases de données", Thèse de doctorat de l'université Mohamed Khider - Biskra, 2012.

[52] Hsu, C.-W., Chang, C.-C., Lin, "A practical guide to support vector classification". Rapport technique, National Taiwan University, 2009.

[53] Hamel L. "Knowledge discovery with support vector machines". Wiley Edition, 2009.

[54] Efron B., Tibshirani R. "Improvements on cross-validation: The. 632+ bootstrap method". Journal of the American Statistical Association, pages 548–560, 1997.

[55] C.-C., Chang, C.-J., Lin. "LIBSVM: a library for support vector machines". Software Available at http://www.csie.ntu.edu.tw/˜cjlin/libsvm, 2001.

[56] Lee J., Lin C. "Automatic Model Selection for Support Vector Machines". Technical Report. http://www.csie.ntu.edu.tw/˜cjlin/papers/modelselect.ps.gz, 2000.

[57] Vapnik V. N., Chapelle O. "Bounds on error expectation for support vector machines". Neural Computation, 12: 9, 2000.

[58] Chapelle O., Vapnik V. N., Bousquet O., Mukherjee S. "Choosing multiple parameters for support vector machines". Machine Learning, 46(1-3): 131-159, 2002.

[59] Chapelle O. "Support Vector Machines : Principes d'induction, Réglage automatique et Connaissances a priori". PhD thesis, Université Pierre et Marie Curie-Paris VI, avril 2004.

[60] Lebrun G. "Sélection de modèles pour la classification supervisée avec des SVM (Séparateurs à Vaste Marge). Application en traitement et analyse d'images". Thèse doctorat de l'université de CAEN/BASSE-NORMANDIE, 2006.

[61] Kuncheva L., Jain L. C. "Nearest neighbor classifier: Simultaneous editing and feature selection". Pattern Recognition Letters, 20(11-13), pages 1149–1156, 1999.

[62] Cerny V. "Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm". Journal of Optimization Theory and Applications, 45(1), pp 41_51, 1985.

[63] Metropolis N., Rosenbluth A. R, Rosenbluth. M. N, Teller. A & Teller. E. "Equation of state calculations by fast computing machines". The Journal of Chemical Physics, vol. 21, 6, pp 1087-1092, 1953.

[64] Ridha Z., "Contribution à l'analyse et à la classification automatique des défauts mécaniques", Thèse de doctorat de l'université de Ferhat Abbas – Setif 1, 2015.

[65] Aarts E. H., Korst L., van Laarhoven M. "Simulated annealing. Local Search in Combinatorial Optimization", pages 91-120, UK, 1997.

[66] Lin S. W., Lee Z. J., Chen S. C., & Tseng T.Y. "Parameter determination of support vector machine and feature selection using simulated annealing approach". Applied Soft Computing, Vol 8, pp 1505–1512, 2008.

[67] Holland J. "Adaptation in Natural and Artificial Systems". The University of Michigan Press, 1975.

[68] Rennard J. P. "Vie artificielle". Vuibert informatique, 2002.

[69] Zhou X., Xu J. "A SVM Model Selection Method Based on Hybrid Genetic Algorithm and Empirical Error Minimization Criterion" The Sixth International Symposium on Neural Networks (ISNN 2009) pp 245-253, 2009.

[70] Yong M., Xiao-bo Z., Dao-ying P. "Parameters selection in gene selection using Gaussian kernel support vector machines by genetic algorithm". Journal of Zhejiang University Science B, 2005.

[71] Amal B., Salma J., Yassine B. "A multi-objective genetic algorithm for simultaneous model and feature selection for support vector machines". Artificial Intelligence Review, 2018.

[72] Anh V. P., Minh L. N., Lam T. B. "Feature weighting and SVM parameters optimization based on genetic algorithms for classification problems". Applied Intelligence, 2017.

[73] Glover, Taillard, Werra, "guide de Tabu Search", rapport technique N°ORWP, 92101, Département de mathématique, Ecole Polytechnique, Séderal de Lausan, Swidzerlant, 1992.

[74] Fred G., Manuel L., "Tabu Search", Kluwer Academic Pub, 1998.

[75] Boutkhil S., "Contribution à la résolution des problèmes de classification multi-classes par les techniques d'optimisation combinatoire", Thèse de doctorat de l'université d'USTO-MB, 2017.

[76] Battiti R., Tecchiolli G., "The reactive tabu search", ORSA journal on computing, 6(2), 1994.

[77] Fabien P., "Recherche d'information sur Internet par algorithmes évolutionnaires", thèse de doctorat, Ecole doctorale : santé, science et technologie, Université François Rabelais Tours, 2004.

[78] Baptiste A., "les métaheuristiques en optimization combinatoire", mémoire présente en vue d'obtenir l'examen probatoire en informatique, Conservatoire National Des Arts Et Metiers, Paris, 2006.

[79] Kennedy J., Eberhart, R. (1995). Particle swarm optimization. The International Conference on Neural Networks, Perth, WA, Australia, 1942-1948.

[80] Zemmal N., "Techniques d'apprentissage pour la sélection d'attributs: Application à la reconnaissance des formes", Thèse de doctorat de l'université de Badji Mokhtar Annaba, 2018.

[81] Kennedy J., Eberhart R., Shi Y. "Swarm Intelligence". Morgan Kaufman, 2001.

[82] Kennedy J., Mendes R. "Population structure and particle swarm performance". The World on Congress on Computational Intelligence, Honolulu, HI, USA, 1671-1676. 2002.

[83] Zemmal N., Azizi N., Sellami M., Zenakhra D., Dey N., Ashour A. S. "Robust Feature Selection Algorithm based on Transductive SVM Wrapper and Genetic Algorithm: Application on Computer-Aided Glaucoma Classification". International Journal of Intelligent Systems Technologies and Applications, 2016.

[84] Eberhart R., Simpson P., Dobbins R. "Computational Intelligence PC Tools". Academic Press Professional, 1996.

[85] Shi Y., Eberhart R. "A modified particle swarm optimizer". The International Conference on Evolutionary Computation, Washington, DC, USA, 69-73. 1998.

[86] Shi Y., & Eberhart R. C. "Empirical study of particle swarm optimization". The Congress on Evolutionary Computation, Washington, DC, USA, 1945-1949. 1999.

[87] Joachims T., "Making Large-Scale SVM Learning Practical" University of Dortmund, Computer Science Department, 1998.

[88] Robert A., "Application des SVM pour la discrimination de contenus musicaux" Cours d'introduction aux théories de l'apprentissage, Paris, le 31janvier 2005.

[89] Anitha Mary M. O. C., Dhanya P. M. "A comparative study of different feature extraction techniques for offline malayalam character recognition". Computational Intelligence in Data Mining, vol. 2, pp. 9-18, 2015.

[90] Dinesh P. M., Sabenian R. S. "Comparative analysis of zoning approaches for recognition of Indo Aryan language using SVM classifier". Cluster Computing, pp 1-8, 2017.

[91] Frans V. "An Analysis of Particle Swarm Optimizers" (Ph.D. Thesis). University of Pretoria, Afrique du Sud, 2006.

[92] Dang H. N., Luong C. M. "A new model of particle swarm optimization for model selection of support vector machine". New Challenges for Intelligent Information and Database Systems, pp. 167-173, 2011.

[93] Lawgali A. "A survey on Arabic character recognition. International Journal of Signal Processing", Image Processing and Pattern Recognition, vol. 8, no. 2, pp. 401-426, 2015.

[94] Lorigo L. M., Govindaraju V. "Offline Arabic handwriting recognition: a survey". IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 5, pp. 712-724, 2006.

[95] Khorsheed M. S. "Off-line Arabic character recognition – a review". Pattern Analysis & Applications, vol. 5, no. 1, pp. 31-45, 2002.

[96] Sari T., Souici L., Sellami M. "Off-line handwritten Arabic character segmentation algorithm". Proceedings of the IEEE International Workshop on Frontiers in Handwriting Recognition, pp. 452-457, 2002.

[97] Al-Hamad H. A., Abu Zitar R. "Development of an efficient neural-based segmentation technique for Arabic handwriting recognition". Pattern Recognition, vol. 43, no. 8, pp. 2773-2798, 2010.

[98] Lawgali A., Bouridane A., Angelova M., Ghassemlooy Z. "Automatic segmentation for Arabic characters in handwriting documents". Proceedings of the IEEE International Conference on Image Processing, pp. 3529-3532, 2011.

[99] Yusra O. "Segmentation algorithm for Arabic handwritten text based on contour analysis". Proceedings of the IEEE International Conference on Computing, Electrical and Electronic Engineering, pp. 447-452, 2013.

[100] Mario P., Samia S., Volker M., Hamid A. "IFN/ENIT-database of handwritten Arabic words". Francophone International Conference on writing and Document, Tunisia, 2002.

Production scientifique

Publication Internationales

- MAMOUNI El Mamoun, ZENNAKI Mahmoud and SADOUNI Kaddour, 'SVM Model Selection Using PSO for Learning Handwritten Arabic Characters', CMC- COMPUTERS MATERIALS & CONTINUA, Copyright © 2019 Tech Science Press CMC, vol.61, no.3, pp.995-1008, 2019, CMC. doi :10.32604/cmc.2019.08081, www.techscience.com/cmc.
- MAMOUNI El Mamoun, ZENNAKI Mahmoud and SADOUNI Kaddour, 'Efficient Analysis of Vertical Projection Histogram to Segment Arabic Handwritten Characters ', CMC- COMPUTERS MATERIALS & CONTINUA, Copyright © 2019 Tech Science Press CMC, vol.60, no.1, pp.55-66, 2019, CMC. doi:10.32604/cmc.2019, www.techscience.com/cmc.
- Mahmoud ZENNAKI, El Mamoun MAMOUNI, Kaddour SADOUNI 'A Comparative Study of SVM Models for Learning Handwritten Arabic Characters', WSEAS TRANSACTIONS on ADVANCES in ENGINEERING EDUCATION, E-ISSN: 2224-3410, Issue 2, Volume 10, pp 82-90, July 2013. http://www.wseas.org/multimedia/journals/education/2013/085705-237.pdf.
- ZENNAKI Mahmoud, SADOUNI Kaddour and MAMOUNI El Mamoun, 'optimal Sélection of SVM Models by Tabou Search metaheuristic for learning hardwritten Arabic Characters', THE MEDITERRANEAN JOURNAL OF COMPUTERS AND NETWORKS, MedJCN, Vol 9, N°03, pp97-103,2013.

Communications Internationales

- OULED JAAFRI Yamina, MAMOUNI El Mamoun, SADOUNI Kaddour, 'Printed Arabic Optical Character Recognition using Support vector machine', International Conference on Mathematics and information Technology, December 4 – 5, 2017, Adrar, Algeria.
- MAMOUNI El Mamoun, ZENNAKI Mahmoud, SADOUNI Kaddour, 'Sélection optimale de modèles SVM pour la reconnaissance des caractères arabes manuscrits', 5^{ième} Edition de la Conférence Internationale sur les technologies d'Information et de la communication pour l'Amazighe TICAM2012, 26 & 27novembre 2012, Rabat, Maroc.
- MAMOUNI El Mamoun, ZENNAKI Mahmoud, SADOUNI Kaddour**, '**Sélection optimale de modèles SVM pour la reconnaissance des caractères arabes manuscrits', Annual

International Conference on Language, literature & Linguistics, 09 & 10 July 2012, Hotel Fort Canning, Singapore.

- MAMOUNI El Mamoun, ZENNAKI Mahmoud, SADOUNI Kaddour, **'**Sélection optimale de modèles SVM pour la reconnaissance des caractères arabes manuscrits', Conférence Internationale sur le traitement de l'information multimédia, 09-11 Avril 2012, Mascara, Algérie.

Communications nationales

- MAMOUNI El Mamoun, ZENNAKI Mahmoud, SADOUNI Kaddour, **'**Sélection optimale de modèles SVM pour la reconnaissance des caractères arabes manuscrits', Première Conférence Nationale sur les télécommunications « CNT'2012 », 11& 12 Novembre 2012, Guelma, Algérie.