REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MENTOURI-CONSTANTINE
FACULTE DES SCIENCES DE L'INGENIEUR
DEPARTEMENT D'ELECTRONIQUE

THESE

Présentée pour l'obtention du diplôme de

DOCTORAT EN SCIENCES

Spécialité: Electronique

Option: Traitement du signal

Par

BOUKHAROUBA Abdelhak

THEME

Contribution à la segmentation et à la reconnaissance de l'écriture arabe manuscrite

Soutenue le 16/11/2011

Devant le jury :

Président	A. Khamadja	Professeur, Université de Constantine
Rapporteur	A. Bennia	Professeur, Université de Constantine
Examinateurs	A. Charef	Professeur, Université de Constantine
	N. Doghmane	Professeur, Université d'Annaba
	A. Boukrouche	Professeur, Université de Guelma

à toute ma famille,
à mes amis

Remerciements

Tout d'abord je voudrais remercier Professeur Abdelhak Bennia, directeur de thèse, pour m'avoir encadré pendant cette thèse.

J'adresse mes remerciements à tous ceux qui ont contribué de près ou de loin à l'aboutissement de cette thèse.

Je voudrais adresser mes remerciements à Dr Nadir Farah pour la base de données qui m'a permis d'évaluer et valider la première partie de mon travail de thèse.

Je remercie les Professeurs A. Khamadja, A. Charef, N. Doghmane A. Boukrouche d'avoir bien voulu accepter d'être membres du Jury de cette thèse.

ملخص

في هذه الأطروحة ، قدمنا مساهمتنا في الأبحاث على دراسة التعرف على الكتابة اليدوية. المساهمة الرئيسية لهذه الأطروحة هي اقتراح حلول تقنية لإنجاز نظاما للتعرف على المبالغ الحرفية و الرقمية للشيكات.

في الجزء الأول من هذه الأطروحة، ونحن مهتمون في المقام الأول بالتعرف على الكلمات العربية المكتوبة بخط اليد. قدمنا نظاما جديدا يقوم على تقسيم واضح للكلمات للتعرف على المبالغ الحرفية المكتوبة بالعربية. ويتم تقييم التعرف باستخدام نموذج هجين متكون من perceptron متعدد الطبقات (MLP) ونماذج ماركوف المخفية (H MM). في التطبيق الخاص بهذه الدراسة، اعتمدنا نموذجا ذا مسار مميز الذي يحتوي على نموذج واحد لجميع أصناف الكلمات، من خلال إيجاد المسار الأمثل. يستخدم نموذج واحد (H MM) لنمذجة الكلمات المقسمة إلى (graphemes).

في البداية ، يتم استخدام النموذج (MLP) باعتباره كمصنف للتعرف على الحروف و أجزاء الحروف (graphemes) الناجمة عن تجزئة الكلمة إلى قطع أساسية. إذا لم ينجح النموذج (MLP) على التعرف على واحدة من الكلمات المرشحة، ينبغي اللجوء إلى نماذج ماركوف (H MM). يتم حساب احتمالات الإصدارات بواسطة perceptron متعددة الطبقات، ويتم الحصول على احتمالات الانتقالات عن طريق الإحصاء و الاحتمالات المسبقة عن طريق حساب عدد الكلمات في كل (فئة) في بنك المعلومات المخصصة لتدريب النماذج. أخيرا، يتم استخدام خوارزمية فيتربي (Viterbi) للحصول على المسار الأمثل الذي يمثل الكلمة المتعرف عليها.

الجزء الثاني يصف نظاما جديدا قصد التعرف على الأرقام الفارسية التي يمكن استخدامها للتعرف على المبالغ الرقمية للشيكات بطريقة أوتوماتيكية. المساهمة الرئيسية لهذا الجزء هو تطوير تقنيات جديدة لاستخراج ميزات قصد تحسين أداء نظم التعرف. لإثبات فعالية هذه التقنية، تم تقييم ومقارنة ثلاثة أنواع من المصنفات (MLP، SVM، Neuro-fuzzy) حيث تم تحقيق أفضل النتائج باستعمال المصنف SVM مع هذه المجموعة من الميزات الجديدة.

Abstract

In this thesis, we presented our contribution to researches on the study of handwriting recognition. The main contribution of this thesis is the proposal of technical solutions allowing the realization of a system for recognizing numerical and literal amounts of checks.

In the first part of this thesis, we are mainly interested in the recognition of handwritten Arabic words. We have presented a new system based on an explicit segmentation for the recognition of Arabic literal amounts. Recognition is achieved using a hybrid model based on Multilayer Perceptron (MLP) and Hidden Markov Models (HMM). In our application, we adopted a path discriminant continuous HMM. Only one HMM is used for all the word classes, and different paths in the model distinguish one word class from the others. The HMM is used to model explicitly the words segmented into graphemes.

First, the MLP is used as a classifier and the grapheme labels (classes) are arranged from right to left according to their appearance in the word. Then we match the grapheme and diacritic sequence against each candidate vocabulary word. If this matching cannot succeed to recognize one of candidate words, the MLP outputs are used as observation (posterior) probabilities of graphemes and the neuro-markovian system is then carried out. The transition probabilities are estimated by counting, and the priori probabilities are obtained by calculating the number of occurrences of each state (class) in the entire training database. Finally, the Viterbi algorithm is used to find the optimal path representing the recognized word.

The second part describes a system for recognizing handwritten Farsi digits that may be used in automatic recognition of the numerical amounts.

The main contribution of this part is the synthesis of new feature extraction techniques to improve the performances of recognition systems. To demonstrate the effectiveness of this technique, three types of classifiers (MLP, SVM and neuro-fuzzy) are evaluated and compared when the SVM with this new feature set yielded the best performance.

Résumé

Dans cette thèse, nous avons présenté notre contribution aux recherches sur l'étude de la reconnaissance de l'écriture manuscrite. La contribution principale de cette thèse est la proposition de solutions techniques permettant la réalisation d'un système de reconnaissance des montants littéraux et numériques des chèques.

Dans la première partie de ce travail de thèse, nous nous sommes principalement intéressés à la reconnaissance des mots arabes manuscrits. Nous avons présenté un nouveau système basé sur une segmentation explicite pour la reconnaissance des montants littéraux arabes. La reconnaissance est évaluée à l'aide d'un modèle hybride à base de Réseau de Neurones de type Perceptron Multi-Couches (PMC) et de Modèles de Markov Cachés (MMC). Dans notre application, nous avons adopté une modélisation (Path Discriminant) qui dispose d'un seul modèle pour toutes les classes des mots, par recherche du chemin optimal. Un seul MMC est utilisé pour modéliser explicitement les mots segmentés en graphèmes.

Au début, le PMC est utilisé comme un classifieur pour reconnaître les graphèmes provenant de la segmentation du mot à reconnaître. Si le PMC n'arrive pas à reconnaître un des mots candidats, le système neuro-markovien est ensuite intervenu. Les probabilités d'émission des observations sont calculées par un perceptron multi-couches, les probabilités de transition sont estimées par comptage, et les probabilités a priori sont obtenues en calculant le nombre d'occurrences de chaque état (classe) dans la base d'apprentissage. Enfin, l'algorithme de Viterbi est utilisé afin de trouver le chemin optimal représentant le mot reconnu.

La deuxième partie décrit un système de reconnaissance de chiffres farsis qui pourrait être utilisé à la lecture automatique des montants numériques de chèques. L'apport principal de cette partie est la synthèse des nouvelles techniques d'extraction de caractéristiques permettant une amélioration des performances des systèmes de reconnaissance. Pour montrer l'efficacité de cette technique, trois types de classifieurs (PMC, SVM et neuro-flou) sont évalués et comparés où le SVM avec ce nouveau jeu de caractéristiques a permis d'obtenir les meilleures performances.

Tables des matières

Introduction générale 1

Chapitre 1 - Méthodes de Classification 4

1.1 Apprentissage 4
1.2 Méthodes statistiques bayesiennes 7
1.3 Les Classifieurs paramétriques 8
1.3.1 Le Classificateur Euclidien 9
1.3.2 Le Classificateur Quadratique 9
1.3.3 Le Classifieur Gaussien 10
1.4 Méthodes basées sur le concept de similarité 11
1.4.1 Distance d'un point à une classe 12
1.4.2 Méthode de k plus proches voisins 13
1.4.3 La classification C-moyenne floue 14
1.5 Réseaux de neurones 15
1.5.1 Neurone formel 15
1.5.2 Perceptron 17
1.5.3 L'Adaline 18
1.5.4 Perceptron multicouches (PMC) 18
1.5.5 Les réseaux RBF (Radial Basis Function) 21
1.5.6 Réseau hybride neuro-flou 22
1.6 Comparaison dynamique 23
1.6.1 Distance d'édition 24
1.7 Méthodes stochastiques à base de modèles de Markov 26
1.7.1 Les modèles de Markov cachés 27
1.7.2 Les fonctionnalités d'un HMM 28
1.7.3 Type de modèles de Markov cachés 31
1.8 Les systèmes hybrides RN-MMC 34
1.9 Machines à vecteurs de support (SVM) 35

Chapitre 2 - Reconnaissance de l'Ecriture et du Document 36

2.1 Supports et formes de documents 36

2.1.1 Formes images 36
2.1.2 Formes électroniques codées 37
2.1.3 Formes électroniques codées et structurées 37
2.2 Analyse d'image de documents 37
2.2.1 Segmentation de documents 39
2.3 Reconnaissance de documents 40
2.3.1 Acquisition des images de documents 40
2.3.2 Pré-traitement 41
2.3.3 Redressement 41
2.3.4 Binarisation 42
2.4 Reconnaissance de l'écriture 43
2.4.1 Types d'écriture 44
2.4.2 Classes des approches de reconnaissance de l'écriture 46
2.5 Architecture générale d'un système OCR (Optical Character Recognition) 47
2.5.1 Acquisition et pré-traitement 49
2.5.2 Segmentation 49
2.5.3 Interprétation de l'image du mot : reconnaissance de mots 49
2.5.4 Extraction des caractéristiques 52
2.5.5 Apprentissage 53
2.5.6 Reconnaissance 54
2.5.7 Post-traitement 55
2.6 Sélection des primitives 56
2.6.1 Analyse en Composantes Principales 56
2.6.2 Analyse Discriminante 58
2.6.3 Estimation de la puissance de discrimination : critère de Fisher 59

Chapitre 3 - Reconnaissance de l'Ecriture Arabe 61

3.1 Caractéristiques de l'écriture arabe 61
3.1.1 Les signes secondaires 63
3.1.2 Ligatures et chevauchements verticaux 64
3.2 Différentes approches et systèmes existants 65
3.3 Prétraitements 67
3.3.1 Correction de l'inclinaison et ligne de base 67
3.4 Segmentation 69

3.4.1 Segmentation en lignes ... 69
3.4.2 Segmentation en mots et en pseudo-mots ... 70
3.4.3 Segmentation en caractères et en graphèmes ... 71
3.5 Reconnaissance de mots ... 72
3.5.1 Reconnaissance en ligne l'écriture manuscrite ... 73
3.5.2 Reconnaissance hors-ligne de l'écriture manuscrite ... 73

Chapitre 4 - Reconnaissance des montants littéraux arabes ... 79

4.1 Architecture et description du système de reconnaissance utilisé ... 79
4.2 Segmentation des mots ... 80
4.2.1 Segmentation en composantes connexes ... 81
4.2.2 Segmentation en graphèmes ... 82
4.3 Description en graphèmes ... 85
4.3.1 Notre alphabet ... 85
4.4 Extraction de caractéristiques : description des graphèmes ... 86
4.4.1 Description d'un graphème à partir de ses contours. ... 86
4.4.2 Description d'un graphème à partir des transitions ... 88
4.4.3 Description d'un graphème à partir des profils ... 89
4.5 Modélisation statistique du système de reconnaissance ... 89
4.5.1 Modélisation par un MMC ... 89
4.5.2 Le perceptron multicouches et estimation des paramètres du réseau neuro-markovien: ... 91
4.6 Application ... 93
4.7 Résultats expérimentaux ... 96
4.8 Conclusion ... 100

Chapitre 5 - Reconnaissance de chiffres farsis manuscrits ... 103

5.1 Base de données : HODA farsi digits ... 103
5.2 Prétraitement ... 104
5.3 Extraction de caractéristiques ... 104
5.4 Reconnaissance ... 105
5.4.1 Classification par un PMC ... 105
5.4.2 Classification par les SVMs ... 106
5.4.3 Système neuro-flou ... 108
5.5 Comparaison des résultas ... 112

5.6 Sélection des caractéristiques 114
5.7 Conclusion 115

Conclusion Générale 117

A. La Logique floue 119

A.1 Sous-ensemble flou 119
A.2 Propositions et règles floues 120
A.3 Fonctionnement d'un système flou 121
A.4 Exemple d'application 122

B. Réseau hybride neuro-flou 124

B.1 Architecture du réseau neuro-flou 124
B. 2 Apprentissage du réseau neuro-flou 125

C. Evaluation de la vraisemblance d'une observation/MMC 127

C.1 Variable "forward" 128
C.2 Variable " backward " 129

D. Machines à vecteurs de supports (MVS) 130

Bibliographie 134

Table des figures

Fig 1. 1 – Exemple pour un problème à trois classes dans l'espace R^3 6
Fig 1. 2 – Deux types d'approche de classification : (a) par séparation, (b) par modélisation. . 7
Fig 1. 3 – Exemple de classification bi-classes avec un kpv : (a) k= 1, (b) k=3. 13
Fig 1. 4 – Modèle d'un neurone formel. 15
Fig 1. 5 – Un réseau de neurones peut être vu comme un graphe : les nœuds du graphe s'appellent neurones, et les arrêtes des liens synaptiques. 17
Fig 1. 6 – Un exemple de perceptron. 17
Fig 1. 7 – Perceptron multicouche avec une seule couche caché. 19
Fig 1. 8 – Réseau RBF à deux entrées, M cellules cachées et deux sorties. 21
Fig 1. 9 – Exemple d'un HMM de 4 états et avec une typologie gauche-droite. 32
Fig 1. 10 – Exemple d'un HMM à densités continues monogaussiennes de 3 états. 33
Fig 2. 1 – Reconnaissance de structures physiques et logiques d'un document : (a) page d'un document, (b) Structure physique, (c) Structure logique. 39
Fig 2. 2 – Choix du seuil sur l'histogramme de l'image à niveaux de gris. 42
Fig 2. 3 – Exemples de l'écriture manuscrite d'après [67]. 45
Fig 2. 4 – Graphe de complexité des systèmes de RAED d'après BELAID [68]. 46
Fig 2. 5 – Système de reconnaissance de caractères (OCR). 48
Fig 2. 6 – Reconnaissance analytique de mots basée graphème, (a) dissection, (b) schéma de la reconnaissance du mot wilson par appariement dynamique des graphèmes ([79]). 51
Fig 2. 7 – Reconnaissance analytique de mots basée sur la reconnaissance de lettres ([79]). . 51
Fig 2. 8 – Exemple de reconnaissance basée sur la lecture humaine ([84]). 52
Fig 2. 9 – Analyse en composantes principales : axes de représentation des données avant (x_1, x_2) et après (z_1, z_2). 57
Fig 2. 10 – Illustration du problème de la réduction adéquate de la dimension de représentation: l'axe z1 est celui qui offre la meilleure qualité de représentation des variables, mais c'est l'axe z2 qui en permet la classification. 58
Fig 2. 11 – La différence entre la LDA et la PCA. 59
Fig 3. 1 – Un mot peut être composé de plusieurs composantes connexes (pseudo-mots). 63
Fig 3. 2 – Signes diacritiques: (a) différentes formes des diacritiques, (b) variabilité des styles des points diacritiques. 64

Fig 3. 3 – Chevauchements et ligatures. 65
Fig 3. 4 – Deux mots appartenant à la même classe [122]. 70
Fig 3. 5 – Exemples de sous-segmentations en pseudo-mots (figure extraite de [125]). 71
Fig 3. 6 – Exemples de sur-segmentations en pseudo-mots (figure extraite de [125]). 71
Fig 3. 7 – Exemple de segmentation en graphèmes (figure extraite de [127]). 71
Fig 3. 8 – Liste des 34 symboles qui constituent l'alphabet. 72
Fig 3. 9 – (a) HMM associé au caractère "س " et (b) Séquence radiale, d'après [134]. 73
Fig 3. 10 – Modèle de mot d'après [135]. 74
Fig 3. 11 – Segmentation en graphèmes : (a) mauvaise segmentation, (b) détection du chevauchement, (c) correction, d'après [139]. 75
Fig 3. 12 – Modèle de caractère (a) et de mot (b), d'après [137]. 75
Fig 3. 13 – Modèle final associé au mot "الحصان ", d'après [139]. 76
Fig 3. 14 – Segmentation en bandes uniformes et non-uniformes (figure extraite de [144]). 77
Fig 3. 15 – (a) Extraction des primitives d'une ligne de texte, (b) HMM à 14 états associé aux caractères, d'après [146]. 77
Fig 3. 16 – PHMM : Découpage en bandes. Un HMM-1D par bande horizontale, et un HMM vertical de 'super-états' (figure extraite de [147]). 78
Fig 4. 1 – Schéma block du système de reconnaissance proposé: les flèches en tiret et continues représentent respectivement les flèches de contrôle et de données. 80
Fig 4. 2 – Différents types de composants connexes formants le mots quatre-vingts dix '' تسعون''. 81
Fig 4. 3 – Les étapes de base pour extraire les (PSPs) : (a) le tracé principal du mot cinq "خمسة ", (b) résultat de localisation verticale de transitions blanc/noir, (c) les points de segmentation primaires (PSPs). 83
Fig 4. 4 – p_2 et p_3 sont à enlever et seulement p1 est retenu. 83
Fig 4. 5 – Filtrage des PSPs redondants: (a et b) les PSPs sont à enlever pour éviter la sur-segmentation des caractères, (c) les PSPs sont maintenus. 84
Fig 4. 6 – Segmentation du mot "خمسة" de la (Fig.4.3-a) en graphèmes : les PSRs finaux sont dessinés par des segments blancs verticaux. 84
Fig 4. 7 – Principe de nettoyage de pixels redondants : le pixel 2 est redondant, et doit être supprimé. 87
Fig 4. 8 – Extraction des chaînes de codes: (a) codage de Freeman à 4 directions, (b) contour du graphème "ى" devisé en 4 régions. 88
Fig 4. 9 – Schéma du système hybride. 93

Fig 4. 10 – Séquence d'observations du mot " مائة ". 94
Fig 4. 11 – Quelques échantillons extraits de la base de données. 96
Fig 4. 12 – Types de segmentations incorrectes: (a) Sur-segmentation: p1 et p2 sont des RPSs et tous les autres segments doivent être enlevés, (b) Exemple de sous-segmentation (ligature verticale), (c) les segments verticaux de "سـ" sont omis 97
Fig 5. 1 – Exemples de chiffres farsi manuscrits. 103
Fig 5. 2 – (a) différents styles des chiffres farsi manuscrits, (b) échantillons des chiffres de différentes qualités. 104
Fig 5. 3 – Système de reconnaissance: les poids des flèches représentent le nombre de chiffres non reconnus de chaque classe. 107
Fig 5. 4 – Activation normalisée entre 0 et 1 des 20 règles obtenues par apprentissage. 110
Fig 5. 5 – Les 10 sorties correspondant aux 10 classes des chiffres farsis. 110
Fig 5. 6 – les dix premières fonctions d'appartenance de la règle 12 : les degrés d'appartenance sont marqués par des lignes verticales. 111
Fig A. 1 – Différentes formes de fonctions d'appartenance. 119
Fig A. 2 – Partition floue de la variable linguistique « température ». 120
Fig A. 3 – Déplacement d'un robot le long d'un mur. 122
Fig A. 4 - Exemple de moteur d'inférence flou. (a) partition des univers du discours de la distance d. (b) partition des univers du discours de l'angle α. (c) résultat de l'implication floue de Mamdani définie par l'opérateur MIN. (d) Résultat d'Agrégation des conclusions définie par l'opérateur MAX et défuzzification. 123
Fig B. 1 – Architecture d'un réseau neuo-flou. 125
Fig C. 1 – Treillis d'états complètement connectés. 127
Fig C. 2 – Opération de base du calcul direct. 128
Fig C. 3 – Induction des variables inverses. 129
Fig D. 1 – Critère de la maximisation de la marge. 130
Fig D. 2 – Projection dans un espace de grande dimension. 133

Liste des tableaux

Tableau 3. 1– Les 28 lettres arabes avec leurs différentes formes d'apparitions dans un mot. 62
Tableau 4. 1 – La base des données consiste de 48 classes différentes de mots. 96
Tableau 4. 2 – Taux de reconnaissance et d'erreur pour les deux classifieurs. 99
Tableau 5. 1 – Matrice de confusion pour la reconnaissance par le PMC. Les colonnes correspondent aux valeurs reconnues, les lignes aux valeurs vraies. 106
Tableau 5. 2 – Chiffres mal classés et taux de reconnaissance pour chaque chiffre. 107
Tableau 5. 3 – Matrice de confusion pour la reconnaissance par le réseau hybride neuro-flou. Les colonnes correspondent aux valeurs reconnues, les lignes aux valeurs vraies. 109
Tableau 5. 4 – Performances des trois classifieurs sur la base 'Hoda'. 112
Tableau 5. 5 – Comparaison avec les plus excellents travaux existants. 113
Tableau 5. 6 – Puissances de discrimination des 46 caractéristiques utilisées. 114
Tableau 5. 7 – Résultats de sélection de primitives. 115

Introduction générale

La reconnaissance de l'écriture arabe remonte aux années 70. Cependant, la reconnaissance du manuscrit arabe reste relativement mal servie malgré l'importance de cette dernière dans les différents domaines tels que la lecture automatique des chèques, des formulaires administratifs et des adresses postales. La majorité des solutions proposées a été testée sur l'écriture latine puis appliquée telle quelle pour la reconnaissance de l'écriture arabe imprimée. Ces méthodes supposent généralement que les caractères peuvent être isolés par une étape de segmentation. Cette étape de segmentation est possible dans le cas d'un texte latin imprimé, mais très difficile dans le cas de l'écriture cursive ou semi-cursive, le cas de l'écriture arabe.

Les deux principales méthodes qui ont été utilisées sont l'approche globale et l'approche analytique. La première modélise le mot dans sa globalité. Elle présente l'avantage de garder le caractère dans son contexte de voisinage mais son inconvénient est qu'elle est limitée à la reconnaissance de vocabulaire très réduit et statique. L'approche analytique quand à elle se base sur la modélisation de l'alphabet de la langue et la segmentation du mot en entités représentant un caractère ou un pseudo-caractère. La reconnaissance du mot consiste à identifier ces entités et à proposer des hypothèses de mots. Cette approche est très liée aux résultats de la segmentation et elle présente l'avantage de pouvoir manipuler un vocabulaire ouvert.

Plusieurs approches ont été testées, mais toujours avec un succès limité. Pour s'affranchir des problèmes de segmentation dus à la complexité de l'écriture arabe, d'autres approches spécifiques ont été proposées à savoir l'approche pseudo-globale fondée sur la notion de pseudo-mot, les approches analytiques qui intègrent la structure morphologique du vocabulaire en post-traitement pour valider les hypothèses de mots et proposer des scénarios de correction, l'approche affixale modélisant les segments linguistiques du mot à préfixe, suffixe, infixe et racine et l'approche neuro-linguistique, modélisant les concepts linguistiques du vocabulaire en terme de réseaux de neurones.

Face aux limites des approches existantes pour la reconnaissance de l'écriture arabe et aux difficultés de segmentation des mots arabes, de nombreuses recherches se sont orientés vers l'utilisation des méthodes stochastiques et en particulier vers les Modèles de Markov Cachés (MMCs) pour la reconnaissance de mots et de textes.

L'objectif de cette thèse est de proposer un système de reconnaissance de l'écriture manuscrite arabe hors-ligne. Nous présentons un nouveau système basé sur une segmentation explicite pour la reconnaissance des mots arabes manuscrits. La reconnaissance est évaluée à l'aide d'un modèle hybride à base de Modèles de Markov Cachés et de Réseau de Neurones de type Perceptron Multi-Couches).

La reconnaissance peut être effectuée de deux façons différentes soit dans le cas d'un modèle par classe, par recherche du modèle discriminant (Model Discriminant), soit dans le cas d'un seul modèle pour toutes les classes, par recherche du chemin optimal qui fournira la classe (Path Discriminant).

Dans notre application, nous avons adopté une modélisation (Path Discriminant) qui dispose d'un seul modèle pour toutes les classes des mots, par recherche du chemin optimal.

Ce mémoire est organisé en deux parties. La première partie présente un état de l'art et composée de trois chapitres. La deuxième partie est dédiée à notre contribution et composée de deux chapitres.

Nous posons au chapitre 1 le problème de la reconnaissance statistique de formes, et présentons divers modèles de classification. Leurs propriétés, architectures et algorithmes d'apprentissages sont également décrits. Des études approfondies sur les méthodes de classification que nous utiliserons dans nos applications sont bien détaillées.

Le chapitre 2 s'intéresse aux problématiques de la reconnaissance automatique de l'écriture et du document.

Dans le chapitre 3, nous présentons les caractéristiques de l'écriture arabes et les travaux effectués dans le domaine de la reconnaissance de l'écriture arabe. Nous basons beaucoup plus sur les approches de type markovien et nous présentons un état sur l'emploi des MMCs de différents types.

Dans le chapitre 4, nous présentons nos travaux sur la reconnaissance des montants arabes manuscrits. L'originalité de notre approche basée sur la segmentation en graphèmes, par rapport aux méthodes classiques, réside dans les points suivants :

Une nouvelle méthode de segmentation qui est adaptée à la particularité de l'écriture Arabe.

Un nouvel alphabet réduit qui exploite un certain nombre de spécificités de l'écriture arabe.

Un nouveau modèle hybride de type neuro-Markovien qui permet d'incorporer plus de contexte en attribuant les signes diacritiques à leurs plus proches graphèmes. Ainsi, un seul modèle qui comprend les états (classes) de graphèmes et les états (classes) des signes diacritiques est construit pour représenter la totalité de l'alphabet.

Dans la section 4.1, nous présentons l'architecture et description du système de reconnaissance utilisé. Dans la section 4.2, nous décrivons les détails de l'algorithme de segmentation proposé. Dans la section 4.3, nous proposons un nouvel alphabet de graphèmes, qui permet de mieux exploiter certaines spécificités de l'écriture arabe telles que les redondances de corps de graphèmes issus de la phase de segmentation. Dans la section 4.4, nous présentons les techniques d'extraction de caractéristiques qui apparaissent les mieux appropriées à notre application. La section 4.5 décrit la modélisation statistique du système de reconnaissance des mots manuscrits. Dans la section 4.6, nous détaillons l'application du système hybride à la reconnaissance des montants littéraux arabes. Dans la section 4.7, nous présentons la partie d'expérimentation avec discussion des résultats et nous terminons par une conclusion.

Dans le chapitre 5, nous proposons un système de reconnaissance de chiffres farsis/persans qui pourrait être utilisé à la lecture automatique des montants numériques de chèques. Dans ce travail, nous présentons un nouveau jeu de primitives pour la caractérisation des chiffres manuscrits. Dans une étude comparative de trois classifieurs (PMC, SVM et neuro-flou), l'utilisation conjointe de SVM et du nouveau jeu de primitives a permis d'obtenir le meilleur résultat.

Enfin nous finirons ce travail de thèse par une conclusion en essayant de mettre en évidence les avantages et les limitations des systèmes présentés. Nous terminerons en donnant quelques perspectives à ces travaux.

Chapitre 1 - Méthodes de Classification

En reconnaissance de formes, les phases d'apprentissage et de classification constituent des étapes fondamentales qui conditionnent en grande partie les performances du système.
Classifier des formes ou individus (par exemple des objets, des images, des phonèmes,…) décrits par un ensemble de grandeurs caractéristiques (taille ou masse de l'objet, pixels de l'image numérisée, spectre acoustique du phonèmes,…), c'est les ranger en un certain nombre de catégories ou classes définies à l'avance.
Citons quelques exemples de classification :
Un exemple courant d'application de la classification est le tri automatique du courrier en fonction du code postal ou de l'adresse manuscrite. Pour un système d'interprétation du code postal, on dispose au départ d'une base de données (un ensemble d'apprentissage) constituée de quelques milliers de chiffres, provenant de scripteurs différents, et identifiés, par un être humain, comme appartenant à une classe donnée, c'est-à-dire comme étant un 0, un 1,…, un 9. A partir de ces exemples munis d'une classe, ou étiquette, l'objectif est de calculer une fonction capable d'associer automatiquement une classe à un chiffre manuscrit ne provenant pas de l'ensemble d'apprentissage.
La classification est l'élaboration d'une règle de décision qui transforme les attributs caractérisant les formes en appartenance à une classe; passage de l'espace de représentation vers l'espace de décision Richard [1]. La classification consiste alors à identifier les classes auxquelles appartiennent les formes à partir des caractéristiques préalablement choisies et calculés. L'algorithme ou la procédure qui réalise cette application est appelé classifieur.
Avant qu'un modèle de décision ne soit intégré dans un système de reconnaissance, il faut avoir procédé auparavant à une étape d'apprentissage.

1.1 Apprentissage

L'apprentissage consiste à caractériser les classes de formes de manière à bien distinguer les familles homogènes de formes. C'est une étape clé dans le système de reconnaissance. On

distingue deux types d'apprentissage : apprentissage *supervisé* et apprentissage *non supervisé* Richard [1].

Dans le cas de l'apprentissage supervisé, les classes sont connues et que l'on dispose d'exemples de chaque classe. Chaque exemple est étiqueté par un opérateur appelé professeur (superviseur), cette étiquette permet d'indiquer au module d'apprentissage la classe dans laquelle le professeur souhaite que la forme soit rangée. Les méthodes de classification supervisée tentent alors d'apprendre, à partir d'objets étiquetés, constituant un échantillon d'apprentissage, une fonction de classification. Cette fonction permettra d'associer une classe à chaque nouvel objet non étiquetée.

Dans le cas de l'apprentissage non supervisé, le problème consiste à regrouper un ensemble de formes non étiquetées en clusters (groupes) significatifs dont le nombre est également inconnu. Les méthodes non supervisées ne nécessitent aucune intervention humaine (classification automatique).

L'algorithme de classification décide d'affecter un objet à une classe. Cet algorithme peut fournir soit une réponse binaire à valeurs discrètes (appartenance ou non à une classe) soit une réponse probabiliste à valeurs continues (l'image à 90% de chance de représenter le chiffre 3). Dans le cas où la règle de décision peut être expliquée de manière linguistique par l'expert, la solution consiste en une suite d'opérations logiques. Lorsque l'expert ne peut pas expliciter son processus de classification, il faut se tourner vers des systèmes de classification qui « apprennent par l'exemple ». Après un lot d'individus déjà classés par l'expert, le système peut apprendre à classer comme l'expert. Après apprentissage, le système est capable de classer de nouveaux individus. Dans ce cas, la règle de décision ne peut être formalisée en termes linguistiques, et l'on a alors recours à une méthode statistique comportant un apprentissage supervisé à partir d'exemples.

Résoudre un problème de classification, c'est trouver une application de l'ensemble des objets à classer, décrits par les variables descriptives choisies, dans l'ensemble des classes. Les variables descriptives considérées ici peuvent être le résultat d'un prétraitement des variables initiales.

Une formulation classique du processus de décision consiste à représenter chaque observation (mesure) $X = [x_1, x_2, x_3, ..., x_d]^T$ par un point dans un espace vectoriel (Euclidien) R^d appelé espace d'observation ou espace des caractéristiques. En raison du bruit, des imperfections de mesure, des distorsions de forme, etc., les observations possibles de chaque classe se traduisent par des points un peu loin les uns des autres. L'ensemble des observations possibles

associées à une classe donnée forme ainsi un amas ou nuage de points groupés autour du point typique. Ces observations peuvent alors être regroupées en groupes, chacun de ces groupes étant associé à une classe particulière. Un exemple pour un problème à trois classes dans un espace à 3-dimensions (R^3) est illustré à la figure 1.1. La règle de décision est obtenue en subdivisant l'espace d'observation en autant de régions distinctes que de classes, de manière à minimiser les risques de fausses décisions. Un problème théorique essentiel est donc celui du mode de partition (choix des frontières entre régions). Un taux élevé de décisions correctes n'est évidemment obtenu que si les amas de poins sont nettement séparés. Pour que cette condition soit satisfaite, il faut, d'une part, chercher à limiter la dispersion des points autour de la valeur typique, c'est-à-dire réduire le bruit et les autres causes d'incertitudes. On doit, d'autre part, vouer une attention particulière au choix des paramètres utilisés, de manière à éloigner le plus possible les amas les uns des autres (maximisation de la distance entre classes) et choisir pour mesurer cette distance une définition approprie.

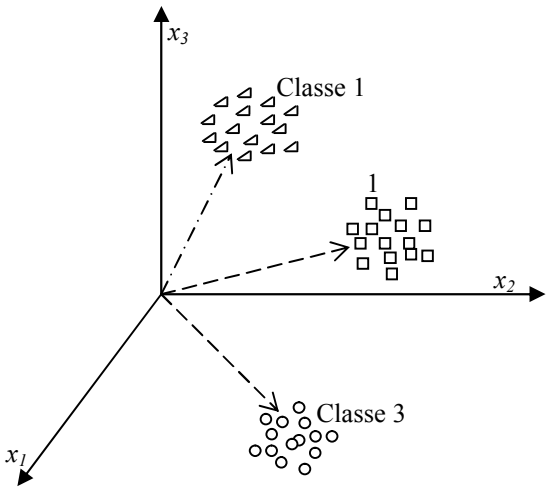
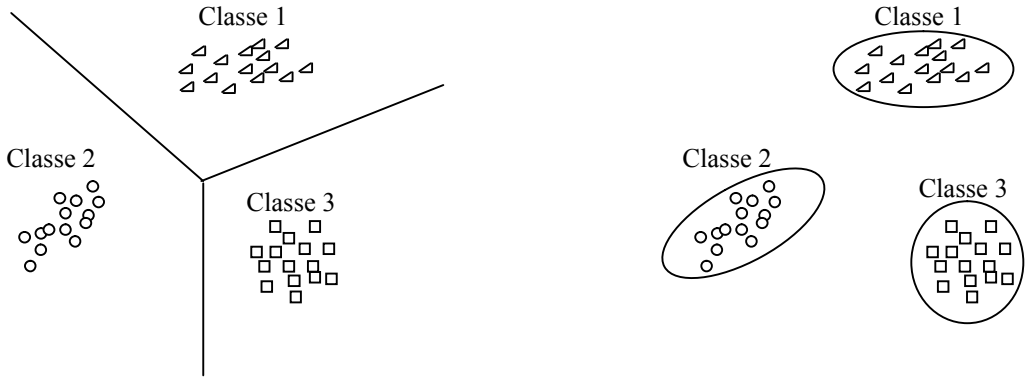


Fig 1. 1 – Exemple pour un problème à trois classes dans l'espace R^3.

Parmi l'ensemble des techniques de classification, il est possible de distinguer deux catégories d'approches, celles agissant par séparation et celles agissant par modélisation Milgram et al [2]. L'objectif du premier type d'approche (figure 1.2-a) est d'optimiser des frontières de décision de manière à séparer au mieux les classes, alors que le second type (figure 1.2-b) cherche à déterminer un modèle le plus fidèle possible de chacune des classes. La décision est alors prise dans le premier cas en se basant sur la position de l'exemple par rapport aux frontières et dans le second cas en utilisant une mesure de similarité pour comparer la donnée à classifier à chacun des modèles.

Ainsi, comme il est montré expérimentalement dans [3], de part leur nature discriminante, les approches par séparation sont plus performantes pour traiter les données ambiguës, mais peu aptes à gérer les « outliers ». Par contre, les approches par modélisation s'avèrent plus efficace pour détecter ces données aberrantes, mais sont généralement peu discriminantes.



(a) par séparation (b) par modélisation

Fig 1. 2 – Deux types d'approche de classification : (a) par séparation, (b) par modélisation.

Un classifieur doit être capable de modéliser au mieux les frontières qui séparent les classes les unes des autres. Cette modélisation fait appel à la notion de *fonction discriminante*, qui permet d'exprimer le critère de classification de la manière suivante:

« Assigner la classe w_i à l'objet représenté par le vecteur X si, et seulement si, la valeur de la fonction discriminante de la classe w_i est supérieure à celle de la fonction discriminante de n'importe quelle autre classe w_j ».

Ou encore, sous forme mathématique:

$$X \in w_i \Leftrightarrow \Phi i(X) \geq \Phi j(X) \quad \forall\, j = 1,2,...C \;;\; j \neq i. \qquad (1.1)$$

où $\Phi i(X)$ est appelé *fonction discriminante* de la classe w_i, et C est le nombre total de classes.

1.2 Méthodes statistiques bayesiennes

L'application des méthodes statistiques bayésienne à la reconnaissance de forme a été formalisée par Chow [4]. Cet ensemble de méthodes est fondé sur un modèle probabiliste qui définit l'appartenance d'une forme à une classe avec un taux d'erreur (risque) minimum. A partir des caractéristiques mesurées, les probabilités d'appartenance des formes aux classes prédéfinies sont déduites, ce qui permet de faire le meilleur choix en cherchant la classe qui maximise la probabilité d'appartenance parmi l'ensemble des classes. On définit par Ω

l'ensemble des objets, et $W=\{w_k\}$ une partition réalisée sur Ω qui représentent les classes possibles d'appartenance.

Soient :

$p(X)$ désigne la probabilité de chaque mesure ou observation X ; celles-ci sont considérées comme équiprobables.

$p(w_k)$ est la probabilité a priori d'avoir la classe w_k.

$p(X/w_k)$ est la probabilité d'une mesure X sachant qu'on a une classe w_k.

$p(w_k/X)$ est la probabilité à posteriori que la classe correcte soit w_k connaissant la mesure (l'observation) X.

Le classifieur optimal, également appelé « Bayesien », est alors celui qui minimise la probabilité d'erreur, c'est-à-dire la probabilité qu'une classe incorrecte soit assignée à un objet. Le critère de classification devient ainsi [4]:

$$X \in w_i \quad \Leftrightarrow \quad p(w_i/X) > p(w_k/X) \qquad \forall k = 1,2,.....,C; k \neq i \tag{1.2}$$

La classe attribuée à l'objet représentée par l'observation X est alors celle dont la probabilité étant donné X est supérieure à la probabilité de n'importe quelle autre classe, étant donné X.

Tandis que $p(w_k/X)$ est inaccessible directement, on peut utiliser le théorème de Bayes :

$$p(w_k/X) = p(X/w_k) * p(w_k)/p(X) \tag{1.3}$$

Comme les $p(X)$ sont équiprobables, trouver le maximum de $p(w_k/X)$ revient à trouver celui du produit $p(X/w_k) * p(w_k)$. Ces quantités peuvent être déterminées par apprentissage sur un corpus d'apprentissage et le critère de Bayes se résume de la manière suivante :

$$X \in w_i \quad \Leftrightarrow \quad p(X/w_i)p(w_i) > p(X/w_k)p(w_k) \qquad \forall k = 1,2,.....,C; k \neq i \tag{1.4}$$

Le problème est qu'il faut connaître les lois $p(X/w_k)$ et $p(w_k)$. Il est généralement possible d'estimer $p(w_k)$, à partir des données expérimentales. Par exemple, pour un problème de reconnaissance de caractères manuscrits, on peut mesurer la fréquence d'apparition des différents caractères. Par contre, la connaissance de $p(X/w_k)$ n'est généralement pas possible, sauf dans des cas très simples. On peut supposer que $p(X/w_k)$ a une forme gaussienne. Les coefficients caractérisant la gaussienne peuvent alors être estimés à partir des données expérimentales.

1.3 Les Classifieurs paramétriques

Ce type d'approche est basé sur le développement d'un modèle pour chacune des classes et l'utilisation d'une mesure de similarité entre l'exemple à classer et les différents modèles.

1.3.1 Le Classificateur Euclidien

La classe dont le vecteur de caractéristiques moyen est le plus proche, au sens de la distance Euclidienne, du vecteur de caractéristiques de l'objet à classifier est assignée à ce dernier. Les fonctions discriminantes utilisées sont donc de la forme suivante:

$$\Phi_i(X) = -\frac{1}{2}(X-\mu_i)^T(X-\mu_i) \qquad (1.5)$$

où $\mu_i = E[X/w_i]$ est le vecteur de caractéristiques moyen des éléments qui appartiennent à la classe w_i , $E[.]$ désignant l'opérateur d'espérance mathématique, et $(\cdot)^T$ celui de transposition.
Le terme quadratique X^TX est indépendant de la classe de l'objet, et les fonctions discriminantes peuvent également s'écrire:

$$\Phi_i(X) = \mu_i{}^T X - \frac{1}{2}\mu_i^T\mu_i \qquad (1.6)$$

Les frontières qui séparent les classes dans l'espace R^d sont ici linéaires.

1.3.2 Le Classificateur Quadratique

Comme le nom l'indique, les frontières de décision fournies par ce type de classifieurs sont décrites par des fonctions discriminantes quadratiques. Les fonctions discriminantes s'expriment:

$$\Phi_i(X) = -\frac{1}{2}(X-\mu_i)^T\Sigma_i^{-1}(X-\mu_i) \qquad (1.7)$$

où $\Sigma_i = E[(X-\mu_i)(X-\mu_i)^T/w_i]$ est la matrice de covariance des vecteurs de caractéristiques de classe w_i.
En pratique, les vecteurs de caractéristiques moyens et les matrices de covariance ne peuvent qu'être estimées à partir des objets disponibles. Une estimation non biaisée à partir d'un ensemble fini de prototypes de chaque classe est donnée par [5]:

$$\tilde{\mu}_i = \frac{1}{N_i}\sum_{X_k\in w_i} X_k \quad \text{et} \quad \tilde{\Sigma}_i = \frac{1}{N_i-1}\sum_{X_k\in w_i}(X_k-\tilde{\mu}_i)(X_k-\tilde{\mu}_i)^T \qquad (1.8)$$

où Ni est le nombre total d'objets de classe w_i et X_k les vecteurs de caractéristiques qui représentent ces objets.
Dans le cas particulier où les composantes des vecteurs de caractéristiques ne sont pas corrélées entre elles, les matrices de covariances expérimentales sont diagonales. L'expression des fonctions discriminantes se réduit alors à:

$$\Phi_i(x) = -\frac{1}{2}\sum_{j=1}^{d}\frac{(x_j - \widetilde{\mu}_{ij})^2}{\widetilde{\sigma}_{ij}^2} \tag{1.9}$$

où $\widetilde{\mu}_{ij}$ et $\widetilde{\sigma}_{ij}^2$ représentent respectivement la moyenne et la variance expérimentales de la $j^{ème}$ composante du vecteur X, calculées sur les éléments de la classe w_i.

1.3.3 Le Classifieur Gaussien

Les fonctions discriminantes utilisées ici sont basées sur une estimation paramétrique des fonctions de répartition des vecteurs de caractéristiques. Ce classifieur suppose que les éléments de chaque classe possèdent une distribution Gaussienne multivariable. Dans la mesure où cette hypothèse s'avère exacte, le classifieur Gaussien permet d'obtenir les frontières optimales de décision de Bayes.

Lorsque les vecteurs de caractéristiques suivent une distribution Gaussienne, les vraisemblances sont estimées par:

$$p(X/w_i) = \frac{1}{\sqrt{(2\pi)^n |\Sigma_i|}}\exp\left\{-\frac{1}{2}(X-\mu_i)^T \Sigma_i^{-1}(X-\mu_i)\right\} \tag{1.10}$$

Le terme $(2\pi)^{-n/2}$ constant, peut être omis pour la classification. En prenant le logarithme, les fonctions discriminantes du classificateur Gaussien s'écrivent:

$$\Phi_i(X) = -\frac{1}{2}(X-\mu_i)^T \Sigma_i^{-1}(X-\mu_i) - \frac{1}{2}|\Sigma_i| + \ln(p(w_i)) \tag{1.11}$$

Les fonctions discriminantes du classificateur Gaussien ne diffèrent de celles du classificateur quadratique que par un biais, spécifique à chaque classe. Les frontières de décision entre les classes sont ici aussi de formes quadratiques. Lorsque les composantes des vecteurs de caractéristiques sont non corrélées, les matrices de covariances sont diagonales. L'expression (3.10) se réduit alors à un produit de n Gaussiennes indépendantes:

$$p(X/w_i) = \prod_{j=1}^{n}\frac{1}{\sqrt{2\pi\sigma_{ij}^2}}\exp\left[-\frac{(x_j-\mu_{ij})^2}{2\sigma_{ij}^2}\right] \tag{1.12}$$

Les fonctions discriminantes du classificateur Gaussien deviennent ainsi:

$$\Phi_i(x) = -\frac{1}{2}\sum_{j=1}^{n}\frac{(x_j-\mu_{ij})^2}{\sigma_{ij}^2} - \frac{1}{2}\sum_{j=1}^{n}\sigma_{ij}^2 + \ln(p(w_i)) \tag{1.13}$$

L'hypothèse de non-corrélation des composantes des vecteurs de caractéristiques permet donc de simplifier fortement les calculs.

Une variante du classificateur Gaussien consiste à estimer la fonction de répartition par une somme pondérée de plusieurs Gaussiennes:

$$p(X/w_i) = \sum_{s=1}^{S} \alpha_{si} \frac{1}{\sqrt{(2\pi)^n |\Sigma_i|}} \exp\left\{ -\frac{1}{2}(X-\mu_{si})^T \Sigma_{si}^{-1} (X-\mu_{si}) \right\} \qquad (1.14)$$

où S est le nombre total de Gaussiennes, et α_{si} des coefficients de pondération devant être déterminés à partir des données, sous les contraintes:

$$\sum_{s=1}^{S} \alpha_{si} = 1, \quad \forall i = 1,....,C; \quad \alpha_{si} \geq 0 \qquad (1.15)$$

Ce classificateur offre ainsi des possibilités plus complexes d'estimation des vraisemblances. La recherche de la valeur des paramètres (importance relative α_{qi} de chacune des Gaussiennes ainsi que leur vecteur de moyennes et matrice de covariance respectifs) doit cependant s'effectuer de manière itérative, dans la mesure où chaque Gaussienne n'est pas associée *à priori* à un groupe bien déterminé de vecteurs de caractéristiques.

1.4 Méthodes basées sur le concept de similarité

Le moyen le plus simple de faire de la classification est de définir une fonction de distance entre les vecteurs caractéristiques, et d'affecter chaque objet d'entrée inconnue à la classe dont le barycentre est le plus proche de l'objet requête, selon la fonction de distance définie. La décision par la distance constitue une des approches les plus simples et les plus intuitives de la reconnaissance des formes. La motivation première d'une telle approche est qu'il est naturel de considérer qu'un élément appartient à une classe s'il est plus proche de cette classe que de toutes les autres.

Distance et similarité sont des notions de ressemblance entre individus. Ce sont des applications particulières du carré de l'espace de représentation des individus dans R^+. Supposons que trois formes a, b et c soient décrites dans un espace de représentation E. Une distance d est une application : $E \times E \rightarrow R^+$ telle que :

1. $\forall (a,b) \in E^2, \quad a \neq b \Rightarrow d(a,b) > 0$
2. $\forall\, a \in E, \quad d(a,a) = 0$
3. $\forall (a,b) \in E^2, \quad d(a,b) = d(b,a)$ d est symétrique.
4. $\forall (a,b,c) \in E^3, \quad d(a,c) \leq d(a,b) + d(b,c)$, d remplit l'inégalité triangulaire.

Le couple (E, d) est alors appelé espace métrique.

La notion de distance correspond à ce qui sépare, celle de similarité est relative à ce qui rapproche les individus. Mathématiquement toute distance correspond à plusieurs indices de similarité et réciproquement, ces deux notions sont symétriques. Ainsi une similarité *s* est une application $s : E \times E \rightarrow [0, s_{max}]$ telle que :

1. $s(a, b) = s(b, a)$, *s* est symétrique
2. $s(a, b) = s_{max}$ si et seulement si a = b.

Si *d* est une distance sur $E \times E$ alors une similarité est définie par :

$s_1(a,b) = \frac{s_{\max}}{d(a,b)+1}$ ou bien par $s_2(a,b) = s_{\max} - \frac{d(a,b)}{\max\limits_{i,j \in E \times E} d(i,j)}$, si la distance est bornée.

On peut définir une distance classique

Par exemple, la distance $d_n(X,Y) = (\sum_{i=1}^{N} |x_i - y_i|^n)^{\frac{1}{n}}$ entre deux vecteurs *X={xi}* et *Y={yi}* correspond à une définition générale de la distance. Où, avec $n = 1$, on retrouve la distance de Hamming, avec $n = 2$, on retrouve la distance euclidienne, et avec $n = \infty$, on retrouve la distance du maximum.

1.4.1 Distance d'un point à une classe

Cette notion est importante dans un système de reconnaissance. En effet, pour attribuer un élément *x* à une classe C_k, il faut définir un critère de proximité comme suit :

$$x \in C_k \Leftrightarrow C_k = Arg\min_{C_i} d(x, C_i), \qquad (1.16)$$

Où $Arg\min(f(C_i))$, est la fonction qui donne la classe C_i minimisant la fonction f. Il est donc nécessaire de définir la distance d'un point à une classe. La définition de cette distance n'est pas unique et dépend des formes traitées. On considère un espace métrique *E*, on peut définir deux exemples de distance.

- La distance d_1 d'un point *p* de *E* à une classe *C* de *E* est définie par:

$$d_1(p, C_i) = \inf\{d(p,m);\ \ m \in C\}.$$

- La distance d_2 entre deux classes C_1 et C_2 est définie par :

$$d_2(C_1, C_2) = \inf\{d(p,m);\ \ p \in C_1 \ et\ \ m \in C_2\}.$$

On remarque que plus la distance considérée est petite, plus on admet que la ressemblance, au point de vue de la reconnaissance des formes, est grande.

1.4.2 Méthode de k plus proches voisins

Une légère variation des méthodes précédentes est l'algorithme des *k* plus proches voisins (*kpv*) ou en anglais (nearest neighbor) : plusieurs représentants sont pris pour chaque classe et, pour chaque forme d'entrée, l'ensemble des *k* plus proches voisins est construit. La forme est affectée à la classe qui a le plus de représentants dans cet ensemble.

Chercher des formes identiques ou similaires dans un dictionnaire est un problème classique et peut être défini pour tout espace métrique. Soit *(E, d)* un espace métrique quelconque et $D \subset E$ un ensemble fini quelconque, $x \in E$ est un élément de E et $s \in R^+$ un réel positif. L'objectif est de trouver le sous-ensemble $D'(x, s) \subset D$ des voisins les plus proches de x tels que :

$$D'(x,s) = \{y \in D / d(x,y) \leq s\} \qquad (1.17)$$

Afin de déterminer les voisins de *x*, une méthode simple consiste à estimer toutes les distances entre *x* et les éléments de *D*. Le coût de cette méthode est proportionnel au nombre d'éléments de *D* et à la complexité du calcul de la distance. Lorsque l'espace métrique est vectoriel, la recherche des plus proches voisins est facilitée car il est possible d'utiliser les coordonnées des éléments. Ces coordonnées permettent également d'obtenir des résultats théoriques plus avancés en ce qui concerne le coût de cette recherche, [6].

Afin d'illustrer cet algorithme, la figure 1.3-a représente un problème de classification à 2 classes avec un algorithme *kpv* où $k = 1$. Les données étiquetées appartenant à la classe 1 et la classe 2 sont respectivement représentées par des carrés blancs et noirs. Une observation non étiquetée A (donnée à classer), est représentée par un carré gris. Afin de classer la donnée A dans un voisinage de k= 1, on recherche le plus proche voisin de A. Le cercle entoure le point à classer et son plus proche voisin. Le plus proche voisin de A est un point de la classe 1, d'où A sera donc affecté à la classe 1.

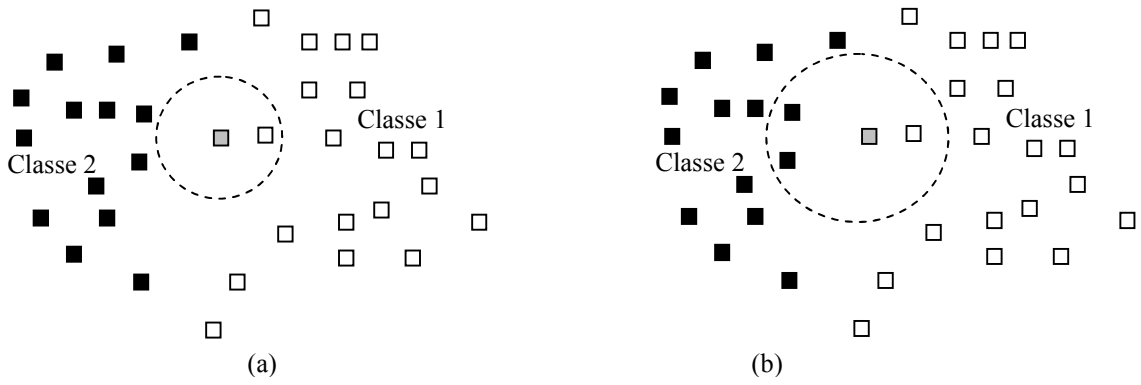


Fig 1. 3 – Exemple de classification bi-classes avec un kpv : (a) *k*= 1, (b) *k*=3.

Considérons maintenant le même problème, mais avec un voisinage de k= 3 points (figure 1.3-b). Afin de classer le point A, on recherche les 3 points les plus proches de A. Le cercle entoure le point à classer et ses trois plus proches voisins. Parmi les 3 points les plus proches de A, il y en a 2 de la classe 2. Un majoritaire est effectué et le point A sera donc affecté à la classe 2.

L'avantage principal de cette méthode est sa simplicité et le fait qu'elle ne nécessite pas d'apprentissage. C'est l'échantillon d'apprentissage, associé à une fonction de distance et d'une fonction de choix de la classe en fonction des classes des voisins les plus proches, qui constitue le modèle. Les knn rentrent alors dans la catégorie des modèles non paramétriques.

L'introduction de nouvelles données permet d'améliorer la qualité de la méthode sans nécessiter la reconstruction d'un modèle. C'est une différence majeure avec des méthodes telles que les réseaux de neurones.

1.4.3 La classification C-moyenne floue

La méthode C-moyenne floue utilise une partition floue tel que chaque point de l'espace de caractéristiques peut appartenir à toutes les classes (clusters) avec différents degrés d'appartenance entre 0 et 1. Cette méthode [7] est fréquemment utilisée en reconnaissance de formes. L'objectif de C-moyenne floue est de trouver les centres des groupes (classes) qui minimisent la fonction coût (objective) suivante :

$$J_m = \sum_{i=1}^{c}\sum_{j=1}^{n} {u_{ij}}^m \left\| x_i - c_j \right\|^2$$
$$s.t \quad \sum_{i=1}^{c} u_{ij} = 1, \forall j = 1,...,n \qquad (1.18)$$

Où m est un nombre réel (exposant flou) supérieur à 1, u_{ij} est le degré d'appartenance de x_i au cluster j, x_i est le $i^{ème}$ donnée, c_j est le centre du cluster j, et ||*|| est la norme Euclidienne.

La partition floue est exécutée à travers une optimisation itérative de la fonction coût (1.18) en ajustant les coefficients d'appartenance u_{ij} et les centres des clusters c_j par :

$$c_i = \frac{\sum_{j=1}^{n} {u_{ij}}^m x_j}{\sum_{j=1}^{n} {u_{ij}}^m} \quad \text{et} \quad u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\left\| x_i - c_j \right\|}{\left\| x_i - c_k \right\|} \right)^{2/(m-1)}}$$

Les itérations seront répétées tant que ($\max_{ij}\left\{\left|u_{ij}^{(k+1)} - u_{ij}^{(k)}\right|\right\} > \varepsilon$), où ε est un critère d'arrêt entre 0 et 1.

Cet algorithme a comme paramètre d'entrée le nombre de classes c. Ils divisent l'ensemble d'objets en c classes. D'abord ils proposent les centres des groupes et ensuite ils assignent chaque objet au centre le plus proche. Dans ces méthodes, le premier choix de centres est aléatoire, et les résultats sont sensibles à l'initialisation.

1.5 Réseaux de neurones

L'idée de base derrière les réseaux de neurones est de s'inspirer des propriétés du cerveau pour construire des systèmes de calcul capable de mieux résoudre le type de problèmes que les êtres vivants savent résoudre.

Les réseaux de neurones ressemblent au cerveau en deux points :

- la connaissance est acquise au travers d'un processus d'apprentissage.
- Les poids des connections entre les neurones sont utilisés pour mémoriser la connaissance.

1.5.1 Neurone formel

Le modèle du neurone formel utilisé aujourd'hui dans toutes les études des machines neuronales date des années 40.

Cette modélisation est inspirée du neurone biologique (figure 1.4).

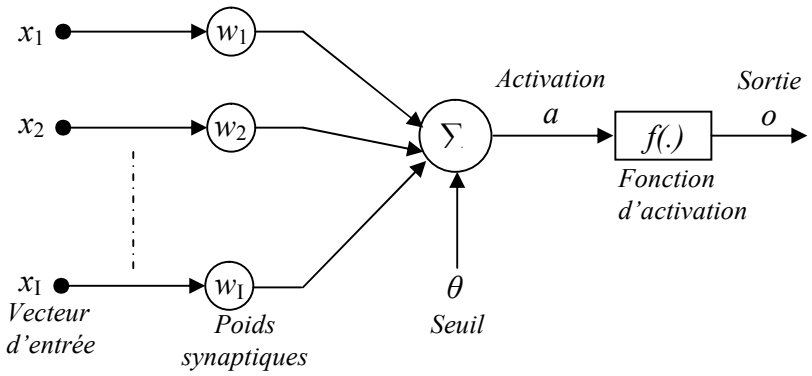


Fig 1. 4 – Modèle d'un neurone formel.

Le neurone formel recalcule son état à chaque instant en fonction de l'influence globale du réseau. Il multiplie la valeur de l'état des neurones en entrée par l'efficacité synaptique correspondante, et additionne le tout (sommateur). Enfin, il compare le résultat à son seuil interne et déduit son nouvel état en utilisant une fonction appelée une fonction d'activation ou de transfert :

$$o = f\left(\sum_{i=1}^{I} x_i w_i - \theta\right) \tag{1.21}$$

- o est appelée la sortie du neurone.
- f : fonction d'activation ou de transfert
- $a = \sum_{i=1}^{I} x_i w_i - \theta$: activation du neurone.
- x_i : Valeur de sortie de la $i^{ème}$ cellule de la rétine.
- w_i : Intensité de la connexion entre la $i^{ème}$ cellule d'entrée et la cellule de sortie.
- θ : le seuil

Le fait d'utiliser un seuil θ est équivalent à avoir une cellule d'entrée, notée généralement x_0=1, toujours active. Dans ce cas, il est facile de voir que w_0 est égal à $-\theta$.

L'activation peut donc se réécrire comme :

$$o = f\left(\sum_{i=0}^{I} x_i w_i\right) \tag{1.22}$$

Dans le modèle original de (Mc Culloch & Pitts) en 1943, la fonction d'activation est la fonction seuil de Heaviside, définie par :

$$f(a) = \begin{cases} 1 & si\ a > 0 \\ 0 & si\ non \end{cases} \tag{1.23}$$

Par conséquent, la règle de décision devient :

$$o = \begin{cases} 1 & pour\ \ a > 0 \\ 0 & pour\ \ a \le 0 \end{cases} \tag{1.24}$$

En général, les neurones sont animés par d'autres types de fonctions d'activations : linéaire ou non linéaires telle que la fonction signe, exponentielle, tangentielle :

Ce neurone formel est l'unité de base des modèles connexionnistes (Réseaux de neurones artificiels).

Formellement, un réseau de neurones est un graphe dont les nœuds sont des unités de calcul (neurones formels), et les arêtes, orientées et pondérées, se nomment liens synaptiques (figure 1.5).

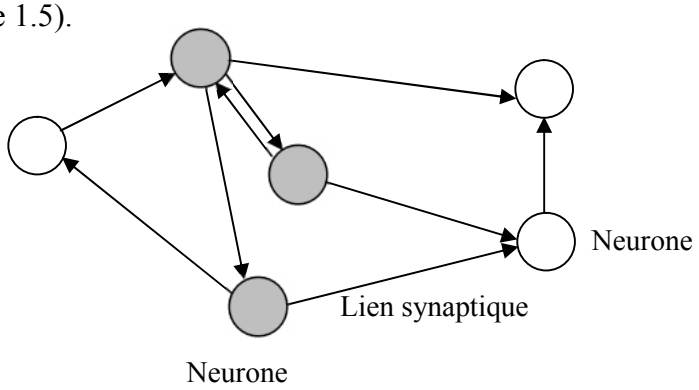


Fig 1. 5 – Un réseau de neurones peut être vu comme un graphe : les nœuds du graphe s'appellent neurones, et les arrêtes des liens synaptiques.

Il y a plusieurs types de modèles de bases de réseaux de neurones, on se restreint ici aux réseaux à couches. Les modèles de réseaux de neurones que nous avons choisis dans le cadre de notre étude sont les Perceptrons Multi-Couches (PMC) et les réseaux de fonctions à base radiale (RBF) : ces modèles sont très utilisés en classification et en reconnaissance de formes.

1.5.2 Perceptron

Le perceptron est considéré comme le premier modèle des réseaux de neurones, il fut mis au point dans les années cinquante par Rosenblatt (1957-1961)[8],[9].

Le perceptron se compose de deux couches de neurones la rétine (n'est pas compté d'où le nom de perception mono-couche) et la couche de sortie [8]. La fonction seuil de Heaviside est utilisée comme fonction d'activation des neurones de la couche de sortie. La figure 1.6 montre un exemple de perceptron.

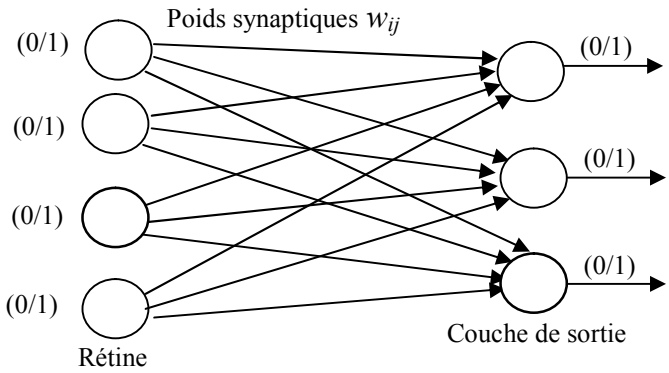


Fig 1. 6 – Un exemple de perceptron.

Les cellules de la première couche sont binaires, répondent en oui / non (0/1).

Les cellules d'entrée sont reliées aux neurones de sortie grâce à des liens synaptiques w_{ij} d'intensité variable.

La règle d'apprentissage du perceptron est la règle de widrow hoff [10]:

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \eta(t_j - o_j)x_i = w_{ij}^{(t)} + \Delta w_{ij} \qquad (1.25)$$

avec :

- Δw_{ij} : Changement à effectuer pour la valeur w_{ij} .
- x_i : Valeur de sortie (0 ou 1) de la i^{eme} cellule de la rétine.
- o_j : Réponse de la j^{eme} cellule de sortie (0 ou 1).
- t_j : Réponse théorique ou (désirée) de la j^{eme} cellule de sortie (0 ou 1).
- $w_{ij}^{(t)}$: Intensité de la connexion entre la i^{eme} cellule d'entrée et la j^{eme} cellule de sortie, au temps t (les valeurs $w_{ij}^{(0)}$ sont généralement choisies au hasard).
- η : Une constante positive généralement comprise entre 0 et 1, sa valeur influe, en effet, sur la vitesse d'apprentissage.

1.5.3 L'Adaline

Le perceptron est souvent considéré comme l'ancêtre des réseaux actuels, mais durant la même période apparut un autre modèle guidé par la théorie des filtres adaptatifs et qui était assez proche du perceptron : (L'ADAptive LINear Element ou ADALINE) [8].

La principale différence entre les deux modèles (perceptron et Adaline) se situe au niveau de la règle d'apprentissage car l'erreur dans le premier est calculée sur la sortie seuillée alors que dans le deuxième, elle est évaluée sur la sortie linéaire, avant le seuillage. L'Adaline est un neurone qui possède des valeurs d'activation continues et une fonction d'activation linéaire : La règle d'apprentissage du perceptron est la règle de widrow hoff [10]:

1.5.4 Perceptron multicouches (PMC)

Les réseaux à deux couches peuvent être entraînés avec des règles d'apprentissage relativement simples. En contrepartie les réseaux à deux couches sont limités au calcul de fonctions très simples. De là, l'intérêt d'utiliser des réseaux plus évolués, contenant en outre des neurones cachés.

Le perceptron multi-couches [8], [11] peut se voir comme une généralisation du perceptron, il se compose d'une première couche de cellules d'entrée (équivalente à la rétine du perceptron), d'une ou plusieurs couches intermédiaires (dites couches cachées), et d'une couche de sortie qui est la seule en contact avec le monde extérieur (ou le superviseur). La connectivité de ce réseau est restreinte: un neurone d'une couche inférieur ne peut être relié qu'a des neurones de couche suivante.

Le calcul de l'activation dans le réseau s'effectue en propageant l'activation initiale de la couche d'entrée jusqu'à la couche de sortie. L'erreur est calculée dans le sens inverse.

En général, les neurones du PMC sont animés par une fonction d'activation sigmoïde, soit exponentielle, soit tangentielle. La non linéarité de ces fonctions est nécessaire pour que les couches cachées du réseau soient utiles. De plus, leur dérivabilité est nécessaire à l'apprentissage du réseau.

La figure 1.7 montre un Perceptron multicouche composé d'une couche d'entrée, une seule couche caché et une couche de sortie.

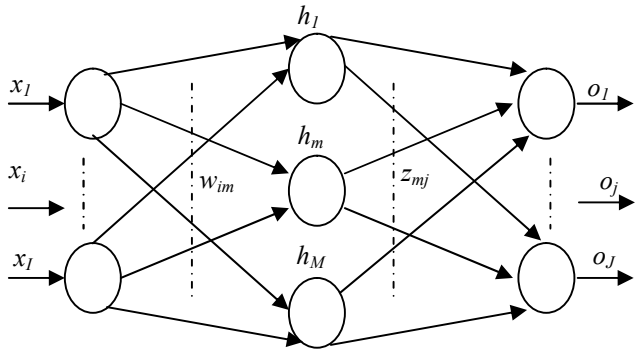


Fig 1. 7 – Perceptron multicouche avec une seule couche caché.

Pour répondre à un stimulus, le signal est propagé de la couche d'entrée à la couche de sortie en passant par la couche cachée, ainsi lorsque un stimulus est présenté en entrée, la réponse du $m^{ième}$ neurone de la couche cachée h_m est donné par :

$h_m = f(w_{0m} + \sum_{i=1}^{I} x_i w_{im}) = f(\sum_{i=0}^{I} x_i w_{im})$, où $x_0 = 1$ est le neurone d'entrée correspondant au biais w_{0m}.

Puis la réponse du $j^{ème}$ neurone de la couche de sortie o_j est donnée par :

$o_j = f(z_{0j} + \sum_{m=1}^{M} h_m z_{mj}) = f(\sum_{m=0}^{M} h_m z_{mj})$, où $h_0 = 1$ est le neurone de la couche cachée correspondant au biais z_{0j}.

f est la fonction d'activation sigmoïde : $f(a) = \dfrac{1}{1+e^{-a}}$ où $f'(a) = f(a)(1-f(a))$

La règle d'apprentissage utilisée est la rétro-propagation de l'erreur [12], qui est une technique de descente du gradient minimisant l'erreur instantanée entre la sortie du réseau o^k et celle désirée t^k lorsque le $k^{ième}$ stimulus est présenté en entrée :

$$E_k = \frac{1}{2}\sum_{j=1}^{J}(t_j^k - o_j^k)^2 \qquad (1.26)$$

L'adaptation des poids synaptiques se calculent de manière itérative pour tous les K stimulus jusqu'au moment où un minimum de l'erreur est trouver. La fin de l'apprentissage peut être décidée lorsque l'erreur totale $E_T = \sum_{k=1}^{K} E_k$ observée à la sortie du réseau devient inférieure à un seuil prédéfini.

La règle d'adaptation avec l'ajout du moment de Rumelhart est définie comme suit [12].

Calculer le signal d'erreur pour les neurones de la couche de sortie et la couche cachée respectivement :

$$\delta_j^k = (t_j^k - o_j^k)o_j^k(1-o_j^k) \qquad (1.27)$$

$$\delta_m^k = h_m^k(1-h_m^k)\sum_{j=0}^{J}\delta_j^k z_{mj} \qquad (1.28)$$

Corriger l'ensemble des poids synaptique z_{mj} et w_{im} suivant les formules suivantes :

$$z_{mj}(t+1) = z_{mj}(t) + \eta\delta_j^k h_m^k + \mu(z_{mj}(t) - z_{mj}(t-1)) \qquad (1.29)$$

$$w_{im}(t+1) = w_{im}(t) + \eta\delta_m^k x_i^k + \mu(w_{im}(t) - w_{im}(t-1)) \qquad (1.30)$$

Avec η et μ sont respectivement le pas d'apprentissage et le moment.

Pour une tache de classification, la couche de sortie comporte un neurone par classe et les valeurs désirées sont : 1 pour le neurone qui correspond à la classe de l'objet présenté à l'entrée et 0 pour tous les autres neurones. De la définition de la fonction d'erreur, l'apprentissage tend à augmenter la valeur de sortie du neurone qui correspond à la classe correcte, tout en cherchant à diminuer les valeurs des sorties des autres neurones. Il en résulte donc un apprentissage discriminant.

Alors, les sorties du perceptron multicouches peuvent être interprétées comme de bonnes approximations, au sens des moindres carrés, des probabilités à posteriori des classes.

Lorsque l'on désire obtenir, en couche de sortie, une image la plus fidèle possible des probabilités à posteriori, il est souhaitable d'assure que la somme des valeurs des sorties soit

égale à l'unité, afin de respecter la définition de la probabilité. A cet effet, la fonction d'activation sigmoïde peut être remplacée par la fonction softmax, définie comme suit [14] :

$$\varphi(u_i) = \frac{e^{u_i}}{\sum_{k=1}^{h_L} e^{u_i}} \qquad (1.31)$$

En pratique cependant, même lorsque seule la fonction sigmoïde est utilisée, la sommes des sorties des neurones possède souvent une tendance naturelle à s'approcher aussi nettement de l'unité [14, 15, 16].

1.5.5 Les réseaux RBF (Radial Basis Function)

Les réseaux à fonctions de base radiales (RBF) sont des modèles connexionnistes simples à mettre en oeuvre et assez intelligibles, et sont très utilisés pour la régression et la discrimination. Les réseaux RBF ont comme origine une technique d'interpolation nommée méthode d'interpolation RBF, employée pour la première fois dans le contexte des réseaux neuronaux par B. Head et Lowe (B. Head et Lowe, 1988) [17].

Les réseaux RBF sont des réseaux à trois couches, figure 1.8 :

Une couche d'entrée, une couche cachée composée de fonctions-noyaux, et une couche de sortie dont les neurones sont généralement animés par une fonction d'activation linéaire:

Chaque neurone de la couche cachée réalise donc une fonction noyau, définie par deux paramètres : la position du centre w_j et la taille σ_j du champs récepteur.

Le but de la méthode est d'approximer un comportement désiré par un ensemble de fonctions noyaux « Kernel Functions ». Les fonctions noyaux sont assemblées pour couvrir de leurs champs récepteurs l'ensemble des données d'entrée, ces fonctions sont ensuite pondérées, et leurs valeurs sont sommées pour produire une valeur de sortie.

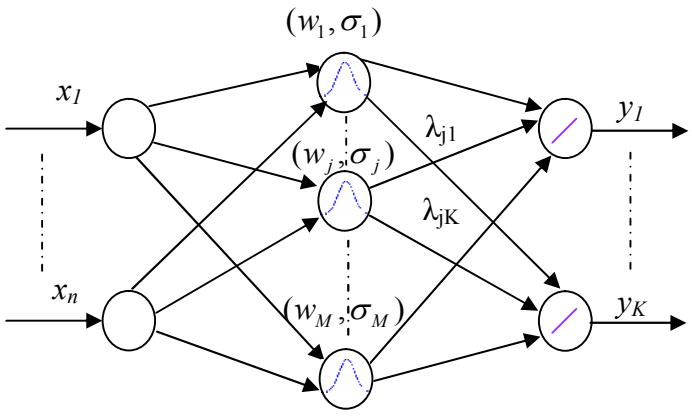


Fig 1. 8 – Réseau RBF à deux entrées, M cellules cachées et deux sorties.

Plus formellement, pour un réseau avec M unités cachées. La sortie y_k du réseau est donnée pour la sortie k et l'entrée x par :

$$y_k = \sum_{j=1}^{M} \lambda_{jk} \times e^{\frac{-d(x,w_j)^2}{2\sigma_j^2}} \qquad (1.32)$$

où w_j et σ_j représentent respectivement le centre et le rayon associés à la cellule cachée j. λ_j est le poids synaptique entre la cellule cachée j et la cellule de sortie k.

Dans les réseaux de neurones de type RBF (Radial Basis Function) la réponse de la gaussienne est donnée par le calcul de la distance entre le vecteur d'entrée et un vecteur prototype stocké dans un neurone de la couche cachée.

L'apprentissage du réseau RBF est hybride puisqu'il est non-supervisé pour la couche cachée et supervisé pour la couche de sortie. Le but de l'apprentissage non-supervisé est de trouver les vecteurs poids des neurones de la couche cachée, qui définissent les centres des régions de sensibilité associées.

1.5.6 Réseau hybride neuro-flou

Les principaux avantages des techniques floues sont l'approche naturelle de la modélisation et la bonne interprétabilité de la description, en employant des règles linguistiques. Cependant, comme il n'y a aucune méthode formelle pour déterminer ses paramètres (ensembles et règles floues), l'exécution d'un système flou peut prendre beaucoup de temps. Dans ce sens, il serait intéressant de disposer d'algorithmes permettant l'apprentissage automatique de ces paramètres. L'une des méthodes qui permet de répondre à ces exigences est la théorie des réseaux de neurones qui emploie des échantillons pour l'apprentissage. La combinaison des deux techniques nous donne les systèmes neuro-flou.

Les systèmes neuro-flous sont des systèmes flous formés par un algorithme d'apprentissage inspiré de la théorie des réseaux de neurones où les règles floues sont codées dans le système neuro-flou.

L'utilisation conjointe des réseaux de neurones et de la logique floue, permet de tirer les avantages des deux méthodes : les capacités d'apprentissage de la première et la lisibilité et la souplesse de la seconde.

De cette manière, les algorithmes d'apprentissage peuvent être employés pour déterminer les paramètres des systèmes flous. Ceci revient à créer ou améliorer un système flou de manière automatique, au moyen des méthodes spécifiques aux réseaux neuronaux.

Par ailleurs, en utilisant des fonctions d'appartenance gaussiennes, on peut faire le rapprochement entre le classifieur de Sugeno et les réseaux de fonctions à base radiale (RBF) voir annexe B.
Un aspect important est que le système reste toujours interprétable en termes de règles floues, vu qu'il est basé sur un système flou.

1.6 Comparaison dynamique

Les classifieurs présentés jusqu' à présent sont statiques, dans le sens où le vecteur de caractéristiques qui leur est fourni est de dimension constante. Cependant certaines formes peuvent être représentées, non plus par un vecteur de caractéristiques unique, mais par une séquence temporelle de vecteurs de caractéristiques où la longueur des séquences de vecteurs est variable. De nouvelles techniques doivent être envisagées, afin de pouvoir tenir compte de cette variabilité temporelle
La comparaison dynamique ou l'alignement temporel (en anglais, *Dynamic Time Warping ou DTW*) est en fait une application au domaine de la reconnaissance de la parole [18] de la méthode plus générale de la programmation dynamique [19]. Elle peut ainsi être vue comme un problème de cheminement dans un graphe [20].

La programmation dynamique est une méthode de recherche d'optimum fondée sur le principe d'optimalité locale de Bellman : « Dans une séquence optimale de décisions, quelle que soit la première décision prise, les décisions subséquentes forment une sous-séquence optimale, compte tenu des résultats de la première décision », Bel57 [19]. Ce principe indique que tout chemin optimal est constitué de portions de chemins elles-mêmes optimales. En effet, si on considère un chemin optimal reliant A à B et un point quelconque M de ce chemin, les chemins de A à M et de M à B sont tous deux optimaux.
Le principe de base est, en fait, de chercher l'alignement qui permet de minimiser la distance ou la distorsion accumulée sur l'ensemble d'une séquence.
La programmation dynamique consiste à calculer récursivement la distance accumulée minimale pour chaque point (i,j) suivant des contraintes locales et d'autres globale (Rabiner et al., 1978) [21].
L'idée de base de la programmation dynamique est : qu'à un point (i,j) on continue juste avec le chemin de la plus petite distance des points suivants ($i-1,j-1$), ($i-1,j$) ou ($i,j-1$).
Si nous désignons par $D(i,j)$ et $d(i,j)$ les distances globale et locale respectivement , donc nous exprimons la formulation mathématique de la PD par la relation récurrente suivante :

$$D(i, j) = min[D(i - 1, j), D(i - 1, j - 1), D(i, j - 1)] + d(i, j) \quad (1.33)$$

Avec la condition initiale : $D(1,1) = d(1,1)$

1.6.1 Distance d'édition

Les distances d'édition permettent de comparer deux mots entre eux ou plus généralement deux séquences de symboles entre elles. L'usage le plus simple est de trouver, pour un mot mal orthographié, le mot le plus proche dans un dictionnaire, c'est une option proposée dans la plupart des traitements de texte.
La distance entre chaînes peut être calculée par minimisation de distances locales entre les composantes des chaînes et optimisation donc de la ressemblance [22].

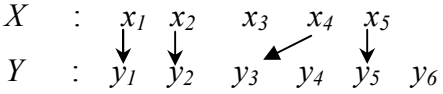
Si l'on prend l'exemple de deux chaînes de caractères {abc} et {abde}, on s'aperçoit que la comparaison peut se faire en termes de transformations pouvant faire passer d'une chaîne à l'autre. Par exemple, pour passer de la première chaîne à la deuxième, il faut conserver les deux premiers caractères « ab », changer le troisième, remplacer la lettre « c » par la lettre « d » et ajouter un « e ».

On remarque que ces opérations peuvent être effectuées à l'aide des trois transformations suivantes :

- L'insertion d'une lettre y_i, de coût $c(\lambda, y_i)$,
- La suppression d'une lettre x_i, de coût $c(x_i, \lambda)$,
- La substitution d'une lettre x_i par une lettre y_i, de coût $c(x_i, y_i)$.

Où λ est le caractère vide. A chacune de ces transformations est associé un coût $c(x_i, y_i)$ pour substitution, $c(x_i, \lambda)$ pour suppression et $c(\lambda, y_i)$ pour insertion. La distance entre deux chaînes est égale au coût total des transformations les moins coûteuses de passage d'une chaîne à l'autre. Elle est appelée distance d'édition $D(X, Y)$. Le schéma de transformation, appelé trace, reproduit par des flèches les transformations concernées.

Soient $X=\{x_1, x_2, x_3, x_4, x_5\}$ et $Y=\{y_1, y_2, y_3, y_4, y_5, y_6\}$. Supposons que la trace de la transformation de X en Y soit la suivante :



Elle peut aussi s'écrire comme suit : substitution (x_1, y_1) ; substitution (x_2, y_2) ; suppression (x_3, λ) ; substitution (x_4, y_3) ; substitution (x_5, y_5) ; insertion (λ, y_6). Chaque opération « trace » est affectée d'un coût et la distance est associée au coût minimal.

De manière générale, soient $X(n)=\{x_1, x_2, ..., x_n\}$ et $Y(m) = \{y_1, y_2, ..., y_m\}$, deux chaînes de longueurs respectives n et m. Il existe trois manières de progresser dans les deux chaînes pour aboutir à (x_r, y_k) :

- Venir de (x_{r-1}, y_{k-1}) et faire substitution (x_r, y_k) ;
- Venir de (x_r, y_{k-1}) et faire insertion (λ, y_k) ;
- Venir de (x_{r-1}, y_k) et faire suppression (x_r, λ).

L'algorithme de *WAGNER* et *FISHER* calcule la distance d'édition avec une complexité $O(m.n)$ par programmation dynamique :

On note n et m les longueurs respectives de X et Y.

$\delta(0,0) = 0$;

Pour i variant de 1 à n effectuer :

$\delta(i,0) = \delta(i-1,0) + \delta(x_i, \lambda)$;

Pour j variant de 1 à m effectuer :

$\delta(0,j) = \delta(0,j-1) + \delta(\lambda, y_i)$;

Pour i variant de 1 à n effectuer :

Pour j variant de 1 à m effectuer :

$m_1 = \delta(i-1, j-1) + \gamma(x_i, y_j)$;

$m_2 = \delta(i-1, j-1) + \gamma(x_i, \lambda)$;

$m_3 = \delta(i-1, j-1) + \gamma(\lambda, y_j)$;

$\delta(i,j) = \min(m_1, m_2, m_3)$

Fin

La distance d'édition $D(X,Y)$ est alors le résultat final de l'algorithme $\delta(m,n)$.

La matrice δij, $1 \leq i,j \leq n$, correspond aux distances cumulées.

Certaines méthodes permettent de réduire ce temps de calcul à l'utilisation par apprentissage a priori de coefficients qui permettent de compacter la connaissance présente dans la base de référence qui devient ainsi un corpus d'apprentissage. Une première méthode mettant en oeuvre ce principe de compactage de la connaissance est le modèle de Markov.

1.7 Méthodes stochastiques à base de modèles de Markov

Les modèles stochastiques ont été mis au point pour décrire des processus qui évoluent au cours du temps. Ils peuvent servir également à modéliser des successions de mesures obtenues en progressant le long d'un axe, et sont donc aussi utilisés pour la reconnaissance de la parole [23] et de l'écriture, manuscrite comme imprimée [24,25]. Pour des raisons de simplicité, nous commençons par le cas discret. Néanmoins, par la suite, les équations présentées sont facilement étendues dans le cas continu.

Une chaîne de Markov discrète est un processus stochastique discret avec des variables aléatoires discrètes (dont les réalisations sont appelées états) :

$X = (x_t)\ t>0$ est une chaîne de Markov, à valeurs dans un espace d'états fini E= ($e_1, e_2, ...e_N$), de cardinal N . Un modèle de Markov discret est un automate stochastique à nombre d'états fini satisfaisant la propriété suivante: "la probabilité d'être dans un état à un instant donné ne dépend que des états visités avant", un modèle de Markov est alors paramétrisé en termes d'un ensemble de probabilités de transition :

$$p\,(x_t = e_j \,/\, x_{t-1} = e_i, x_{t-2} = e_n, \ldots\ldots) \qquad (1.34)$$

Elle est définie par :

1- La distribution de probabilités des états initiaux :

$$\Pi = \{\pi_i\} = \{p(x_1 = e_i)\}, \quad e_i \in E$$

où $\pi_i = p(x_1 = e_i)$ est la probabilité d'être dans l'état e_i à l'instant t = 1.

2- la matrice de probabilités de transition :

$$A = \{a_{ij}\} = \{p(x_t = e_j \,/\, x_{t-1} = e_i)\} \qquad 1 \le i, j \le N$$

Les transitions entre les états se produisent entre deux instants discrets consécutifs, selon une certaine loi de probabilité.

Les probabilités a_{ij} doivent évidemment vérifier les propriétés suivantes:

$$0 \le a_{ij} \le 1 \quad \text{et} \quad \sum_{j=1}^{N} a_{ij} = 1$$

Maintenant on fait les deux hypothèses simplificatrices suivantes:

1. On suppose que le modèle de Markov est d'ordre 1, c'est-à-dire que la probabilité (1.34) de passer à un état particulier e_j à l'instant t ne dépend que de l'état à l'instant $t-1$. Ceci revient alors a supposer que

$$\mathrm{P}\,(\mathrm{x_t} = \mathrm{e_j} \,/\, \mathrm{x_{t-1}} = \mathrm{e_i}, \mathrm{x_{t-2}} = \mathrm{e_n}, \ldots\ldots) = p(x_t = \mathrm{e_j} / \mathrm{x_{t-1}} = \mathrm{e_i}) \qquad (1.35)$$

Avec $p(x_t = e_j / x_{t-1} = e_i) = a_{ij}$ pour $e_i, e_j \in E$, *et* $\forall t \geq 1$

2. On suppose également que ces probabilités de transition sont indépendantes du temps (modèle stationnaire). Cela veut dire que :

$$p(x_{t+1} = e_j / x_t = e_i) = a_{ij} \quad \forall t \geq 1 \tag{1.36}$$

Un problème intéressant associé aux modèles de Markov est le calcul de la durée pendant laquelle le système reste dans un état donné, et qui se formule de la façon suivante : quelle est la probabilité qu'un système se trouvant dans un état e_i y demeure pour une durée d ? Il s'agit simplement de la probabilité de la séquence d'observation suivante :

$$X = \underbrace{\{e_i, e_i, ,, \ldots, e_i}_{d}, e_j \neq e_i \} \tag{1.37}$$

Donc, la probabilité qu'un système se trouve dans un état e_i pendant une durée d est :

$$p_i(d) = p(X / x_t = e_i) = (a_{ii})^{d-1}(1 - a_{ii}) \tag{1.38}$$

1.7.1 Les modèles de Markov cachés

Un modèle de Markov caché (Hidden Markov Model ou HMM) est une chaîne de Markov stationnaire où l'observation est une fonction probabiliste de l'état. On a aussi une notion de séquence d'observations qui apparaît, c'est-à-dire qu'à chaque instant donné on observe une réalisation d'une variable aléatoire suivant la loi de probabilité associée à l'état visité à cet instant. Ces lois sont donc appelées les lois d'émission.

Soit $O = (O_t)\ t \geq 1$ le processus des observations associé à chaque état à valeurs dans l'espace mesurable $V=\{v_1, v_2, \ldots, v_M\}$ à alphabet fini, dont le nombre d'observations possibles est M.

Ayant une séquence d'observations $O=(o_1, o_2, \ldots, o_T)$ et une séquence d'états correspondantes, dans un MMC, chaque élément de séquence d'observations est supposé ne dépendre que de l'état correspondant. L'hypothèse d'indépendance se traduit par :

$$\begin{aligned} &p(o_1 = v_k, o_2 = v_1, \ldots, o_T = v_m / x_1 = e_i, x_2 = e_j, \ldots, x_T = e_n) = \\ &p(o_1 = vk / x_1 = e_i) p(o_2 = v_l / x_2 = e_j) \ldots p(o_T = v_m / x_T = e_n) \end{aligned} \tag{1.39}$$

La probabilité $b_j(k) = p(o_t = v_k / x_t = e_j)$ désigne la probabilité d'observer le symbole v_k à l'instant t, sachant que l'on est dans l'état e_j. Cette probabilité est généralement appelée probabilité d'émission. La matrice $B = \{b_j(k)\}$ est stochastique, i .e.

$$0 \leq b_j(k) \leq 1 \quad \text{et} \quad \sum_{j=1}^{M} b_j(k) = 1 \quad \forall j \in [1.. \ldots N].$$

Par conséquent, un modèle de Markov caché est un processus doublement stochastique, dans lequel les observations sont une fonction aléatoire de l'état, et dont l'état change à chaque instant en fonction des probabilités de transition issues de l'état antérieur. Le fait que les états soient non directement observables, d'où leur nom de cachés.

Les paramètres qui définissent un HMM discret sont alors les suivants :

– Le nombre d'états cachés N ; $e_1, \ldots, e_N$.

– Le nombre d'observations possibles M. Ces observations appartiennent à un alphabet fini $\{v_1, v_2, \ldots, v_M\}$.

– La matrice donnant la distribution initiale des états $\pi = (\pi_i)$ $i=1, \ldots, N$ où $\pi_i = p(x_1 = e_i)$ est la probabilité d'être dans l'état e_i à l'instant $t = 1$.

– La matrice des probabilités de transition $A = \{a_{ij}\}$ où a_{ij} est la probabilité de passer de l'état ei à l'état e j :

– La matrice des probabilités d'émission des observations B = $\{b_j(k)\}$ où $b_j(k)$ est la probabilité d'être à l'état e_j et d'émettre l'observation o_k à cet état.

– Un HMM est donc entièrement défini par le triplet (A, B, π).

1.7.2 Les fonctionnalités d'un HMM

Les fonctionnalités d'un HMM sont l'évaluation de la probabilité d'observation d'une séquence O, l'apprentissage à partir d'un ensemble d'échantillons (séquences d'observations) et la reconnaissance d'une séquence. La description de ces fonctionnalités se base en grande partie sur le travail de Rabiner [23], adoptant parfois une présentation légèrement différente [26].

- **l'évaluation**

Etant donnée une CMC λ et une observation O, on cherche à calculer la vraisemblance de l'observation O avec la CMC λ, c'est à dire avec quelle probabilité la CMC λ engendre l'observation O. Cette valeur est notée P(O / λ). Un calcul direct de cette valeur est bien sûr possible, mais très coûteux en nombre d'opérations car il énumérerait tous les chemins possibles [27]. Une évaluation optimale de cette probabilité est obtenue par les fonctions forward-backward [28] :

$$p(O/\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{1.40}$$

$$p(O/\lambda) = \sum_{i=1}^{N} \pi_i . b_i(o_1) . \beta_1(i) \tag{1.41}$$

Où : la variable "forward" $\alpha_t(i) = p(o_1, o_2, ..., o_t, x_t = e_i / \lambda)$ représente la probabilité de générer l'observation partielle $(o_1, \ldots, o_t)$ sachant que le modèle λ se trouve dans l'état e_i à l'instant t. et la variable "backward" $\beta_t(i) = p(o_{t+1}, o_{t+2}, ..., o_T, x_t = e_i / \lambda)$ représente la probabilité d'observer la séquence partielle $(o_{t+1}, \ldots, o_T)$ sachant qu'on est à l'état e_i à l'instant t, pour le modèle λ. Le calcul de l'évaluation est détaillé dans l'annexe C.

- **Apprentissage**

Cette étape est très importante car la qualité du décodage est étroitement liée à la qualité des modèles en sortie de l'apprentissage. Le but de l'apprentissage est de déterminer les paramètres (A, B, π) qui maximisent la fonction de vraisemblance $P(O|\lambda)$ d'une séquence donnée d'observations des échantillons d'apprentissage.

Cela pose un problème relatif à l'absence de critères d'optimisation globaux. Les solutions utilisées ne présentent que des optimisations locales tel que l'algorithme de Baum-Welch qui garantit l'atteinte d'un maximum local de la fonction de vraisemblance par ré-estimation des paramètres A, B, π.

L'algorithme de Baum-Welch [29] est un algorithme d'apprentissage qui estime itérativement les paramètres des modèles de manière à maximiser la vraisemblance de génération de la séquence d'observation. Cet algorithme n'est qu'une forme particulière de l'algorithme EM (Expectation-Maximisation) [30] dans le cas discret et se déroule en trois étapes :

1. Le calcul des deux variables $\alpha_t(i)$ et $\beta_t(i)$ avec l'algorithme Forward-backward décrit précédemment.

2. La probabilité d'occuper l'état e_i à l'instant t et l'état e_j à l'instant $t+1$, étant donnée le modèle λ et la séquence d'observation O; cette probabilité est fonction de $\alpha_t(i)$ et $\beta_t(i)$ [31]. On a alors :

$$\xi_t(i,j) = p(x_t = e_i, x_{t+1} = e_j / O, \lambda) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)} \quad (1.42)$$

la probabilité de se trouver dans l'état e_i à l'instant t.

$$\gamma_t(i) = p(x_t = e_i / O, \lambda) = \sum_{j=1}^{N} \xi_t(i,j) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \quad (1.43)$$

Cette phase correspond à l'étape de l'espérance "E" dans l'algorithme EM.

3. Les formules de ré-estimation des paramètres (A, B, π) s'écrivent à l'aide de ces quantités

$\overline{\pi}_i = nb$ de fois d'être dans e_i à l'instant (t = 1) $= \gamma_1(i)$ (1.45)

$$\overline{a}_{ij} = \frac{nb.transitions\, e_i \rightarrow e_j}{nb.transitions\, à\, partir\, de\, s_i} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \quad (1.45)$$

$$\overline{b}_j(l) = \frac{nb.de\ fois\ d'etre\, dans\, e_j\ et\, d'observer\, v_l}{nb.de\ fois\ d'etre\, dans\, e_j} = \frac{\sum_{\substack{t=1 \\ o\ =v_l}}^{T-1} \gamma_t(j)}{\sum_{t=1}^{T-1} \gamma_t(j)} \quad (1.46)$$

Cette phase correspond à l'étape de maximisation "M" de l'algorithme EM.

Nous pouvons noter que l'initialisation de cet algorithme conditionne l'optimalité des paramètres du modèle.

- **La reconnaissance (décodage)**

La reconnaissance peut être effectuée de deux façons différentes soit dans le cas d'un modèle par classe, par recherche du modèle discriminant (Model Discriminant), soit dans le cas d'un seul modèle pour toutes les classes, par recherche du chemin optimal qui fournira la classe (Path Discriminant) [32].

Dans le premier cas, la reconnaissance peut se faire simplement par le calcul des probabilités d'émission de la forme par les modèles que l'on suppose a priori équiprobables. La forme à reconnaître est affectée à la classe dont le modèle fournit la probabilité la plus importante :

$$\lambda^* = \arg\max_{\lambda \in \Lambda} p(O/\lambda) \quad (1.47)$$

où Λ désigne l'ensemble des modèles .

Dans le deuxième cas, la reconnaissance consiste à déterminer le chemin correspondant à l'observation, c'est-à-dire à trouver dans le modèle, la meilleure suite d'états, appelée suite d'états de Viterbi, qui maximise la quantité $p(X/O,\lambda)$. Ceci revient à trouver le meilleur chemin dans un graphe. La structure de ce graphe se prête aux techniques de la programmation dynamique.

Pour cela, l'algorithme de Viterbi [33] définit $\delta_t(i)$ qui est la probabilité du meilleur chemin amenant à l'état e_i à l'instant t, en étant guidé par les t premières observations :

$$\delta_t(i) = \max_{x_1,....,x_{t-1}} p(x_1, x_2, ..., x_t = e_i, o_1, o_2, ..., o_t / \lambda) \quad (1.48)$$

C'est-à-dire que $\delta_t(i)$ est la meilleure correspondance entre la suite $x_1, x_2 \ldots x_t$ et la suite o_1 $o_2, \ldots o_t$ avec la contrainte $x_t = e_i$. Par induction, on calcule :

1. Initialisation : $\delta_1(j) = \pi_j b_j(o_1)$, $1 \le j \le N$

2. Récurrence croissante : $\delta_t(j) = \max_{1 \le i \le N} \left[\delta_{t-1}(i) a_{ij}\right] b_j(o_t)$,

$\Psi_t(e_j) = \arg\max_{e_i \in E} \left[\delta_{t-1}(i) a_{ij}\right]$, $1 \le j \le N$, t=2, 3,….,T

3. Terminaison : $p^*(O/\lambda) = \max_{1 \le i \le N} \delta_T(i)$

$x_T = \arg\max_{e_i \in E} \left[\delta_T(i)\right]$

4. Séquence d'états $X^* = \left\{x_t^*\right\}_{1 \le t \le T}$ obtenue par récurrence décroissante : $x_{t-1}^* = \Psi_t(x_t^*)$

$t = T \ldots 2$

On garde trace, lors du calcul, de la suite d'états qui donne le meilleur chemin amenant à l'état e_i à l'instant t.

1.7.3 Type de modèles de Markov cachés

- ***Représentation d'une chaîne de Markov sous forme de graphe***

Il est possible de modéliser une chaîne de Markov sous forme de graphe où les noeuds sont les états et les transitions les arcs. Une probabilité non nulle de passer d'un état e_i à un autre état e_j peut être envisagée comme un lien unidirectionnel entre ces deux états dont le poids est la probabilité de transition de l'état e_i vers l'état e_j. L'ensemble des probabilités non nulles d'un modèle définit un ensemble de liens entre les états qui peut être décrit par un graphe. Un modèle entièrement connecté de N états contient $N^2 + 2N$ connexions. Une structure de graphe permet de diminuer ce nombre de connexions en ne tenant compte que des connexions non nulles (les modèles utilisés pour la reconnaissance de l'écriture contiennent en général une grande part de connexions nulles.). Un exemple de HMM avec 4 états est présenté dans la figure 1.9.

Il existe plusieurs topologies de HMM. La plus répondue est celle de Bakis [34]. C'est un modèle gauche-droite et il représente bien l'évolution temporelle. Autrement dit, on ne peut avoir que des transitions d'un état e_i à un état e_j sachant que $j \ge i$.

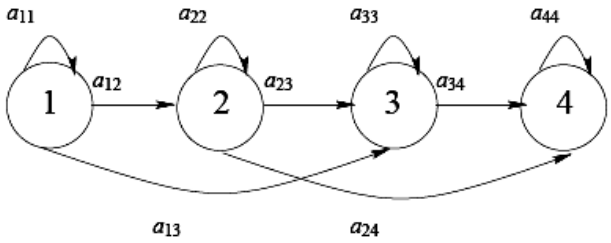


Fig 1. 9 – Exemple d'un HMM de 4 états et avec une typologie gauche-droite.

- ***MMC continus***

Dans le cadre du formalisme initial des MMC discrets, les vraisemblances des observations $p(o_t \mid x_t = e_i, \lambda) = b_i(o_t)$ sont estimées pour chaque état par des distributions discrètes de probabilité. Ceci nécessite que le nombre de formes soit limité. Le problème est que les observations sont en général décrites par des vecteurs continus : l'utilisation de modèles de distributions discrètes implique donc une phase préalable de quantification de ces vecteurs, avec les dégradations qui en résultent. On fait alors une segmentation de l'espace de représentation des formes en un nombre fini M de régions par exemple par une quantification vectorielle (QV). Il faut alors faire un compromis entre un nombre important de régions, pour modéliser plus finement l'espace des observations, et le nombre de probabilités d'observation à évaluer, pour chaque état. Il est dès lors intéressant d'inclure des densités d'observations continues dans les modèles de Markov (HMM à densités continues) qui sont utilisées plus souvent. On a donc une densité continue sur R^N associée à chaque état e_i notée $f_i(.)$. Maintenant l'observation o_t peut prendre n'importe quelle valeur dans R^N, mais on ne peut plus calculer la probabilité d'émission, puisqu'elle est toujours nulle pour les lois continues. Donc on calcule la vraisemblance $p(o_t / x_t = e_i) \equiv f_i(o_t)$, qui est appelée par analogie vraisemblance d'émission.

Ensuite on se pose la question du choix des densités continues pour la modélisation. Afin de limiter le nombre de paramètres de ces distributions, en première approximation, on choisit souvent, une somme finie de densités gaussiennes, ou multi-gaussienne :

$$b_j(O) = \sum_{m=1}^{M} c_{jm} . N(O, \mu_{jm}, \Sigma_{jm}) \qquad (1.49)$$

Les paramètres du modèle à ajuster sont alors, en outre des coefficients a_{ij} et π_i, les coefficients de pondération c_{jm}, les vecteurs de moyennes μ_{jm}, et les matrices de covariance $\sum_{jm}$.

Pour donner une idée, un HMM à trois états est représenté dans la figure 1.10.

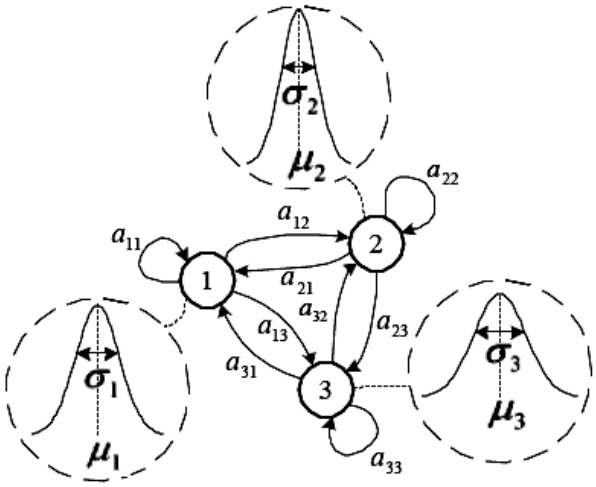


Fig 1. 10 – Exemple d'un HMM à densités continues monogaussiennes de 3 états.

Les formules de re-estimation nécessitent alors la définition de la quantité suivante :

$$\gamma_t(j,m) = \frac{\alpha_t(j)\beta_t(j)}{\sum_{t=1}^{T}\alpha_t(j)\beta_t(j)} \frac{c_{jm}.f(O_t,\mu_{jm},\Sigma_{jm})}{\sum_{n=1}^{M}c_{jn}.f(O_t,\mu_{jn},\Sigma_{jn})} \tag{1.50}$$

Ce paramètre généralise la notion de $\gamma_t(j)$ au cas continu : il s'agit en effet de la probabilité de se trouver dans l'état e_j au temps t, en ne prenant en compte que la $m^{ème}$ composante de la multigaussienne de l'état considéré, en o_t.

Les formules de re-estimation des paramètres a_{ij} et π_i restent inchangées, alors que les formules de re-estimation des nouveaux paramètres deviennent :

$$\bar{c}_{jm} = \frac{\sum_{t=1}^{T}\gamma_t(j,m)}{\sum_{t=1}^{T}\sum_{n=1}^{M}\gamma_t(j,n)} \tag{1.51}$$

$$\bar{\mu}_{jm} = \frac{\sum_{t=1}^{T}\gamma_t(j,m).o_t}{\sum_{t=1}^{T}\gamma_t(j,m)} \tag{1.52}$$

$$\overline{\Sigma}_{jm} = \frac{\sum_{t=1}^{T} \gamma_t(j,m).(o_t - \mu_{jm}).(o_t - \mu_{jm})'}{\sum_{t=1}^{T} \gamma_t(j,m)} \quad (1.53)$$

1.8 Les systèmes hybrides RN-MMC

La stratégie la plus courante dans les systèmes hybrides RN-MMCs est d'utiliser les RNs pour estimer les probabilités d'observation des MMCs dans leur formalisme habituel.

Les RNs évaluent des vraisemblances des observations pour les états $p(o_t|x_t=e_i)$ comme les distributions discrètes pour les MMCs discrets ou les sommes pondérées de fonctions de densité de probabilité pour les MMCs continus et semi-continus. Les MMCs, avec des probabilités de transition attachées à leurs états, permettent de faire le décodage d'une séquence d'observations.

Le système hybride RN-MMC le plus courant a été formalisé et détaillé dans [35,36, 37]. Il consiste à estimer par un Perceptron Multicouches (PMC), les probabilités a posteriori des états, qui sont utilisées pour calculer les vraisemblances normalisées (scaled likelihood) pour les probabilités d'observation des MMCs. La section 4.5 du chapitre 4 détaille le formalisme mathématique de l'hybridation des Perceptron Multicouches et modèle de Markov caché.

Les avantages des RNs pour estimer les probabilités d'observation sont les suivants :

1. Les RNs n'imposent pas d'hypothèse sur la forme des distributions de probabilité des objets (phonèmes ou graphèmes) dans l'espace de représentation (même si la fonction de sortie Softmax correspond à une décision bayesienne sur des distributions gaussiennes).
2. De plus, les RNs modélisent les frontières de décision et non la distribution des formes, ce qui nécessite moins de paramètres.
3. Les RNs sont soumis à un apprentissage discriminant, sur toutes les formes possibles. Cela permet de diminuer le manque de pouvoir de discrimination des MMCs.
4. Les RNs sont capables de traiter des données fortement corrélées, dans un espace de grande dimension. Dans le cas de la lecture du manuscrit, les graphèmes, définis comme des sous-unités de lettres, appartiennent à plusieurs lettres simultanément et les lettres de l'écriture cursive se ressemblent. Il est alors difficile d'attribuer un graphème à une seule classe, et il est plus précis de distribuer la probabilité sur plusieurs sorties.

1.9 Machines à vecteurs de support (SVM)

Récemment, par le biais d'une étude comparative des principales techniques utilisées en reconnaissance de chiffres manuscrits, il a été montré dans [38] que les machines à vecteurs de support permettent une meilleure généralisation que les classifieurs neuronaux classiques tel que les MLP ou les réseaux RBF (Radial Basis Function). Par ailleurs, grâce à l'augmentation de la puissance de calcul et aux développements de nouveaux algorithmes d'apprentissage, il est aujourd'hui possible d'entraîner des SVM à résoudre des problèmes réels de grandes dimensions.

Ainsi, étant donné un exemple de test x et un ensemble de données d'apprentissage étiquetées $\{(x_k, y_k) : k = 1,\ldots,n\}$, où $x_k \in R^d$ et $y_k \in \{1,-1\}$, la sortie du SVM correspondant est :

$$f(x) = sign\left(\sum_{\text{sup } port\ vectors} y_k \alpha_k K(x_k, x) + \beta\right) \tag{1.54}$$

où les données d'apprentissage, dont les multiplicateurs de Lagrange α_k sont différents de 0, sont nommées vecteurs de support. La formulation mathématique est décrite dans l'annexe D.

Malheureusement, la sortie d'un SVM n'est pas nécessairement calibrée. Mais une solution est proposée dans [39] pour estimer facilement des probabilités a posteriori à partir des sorties d'un SVM.

Un SVM est un classifieur binaire, il est donc nécessaire de combiner plusieurs SVM pour résoudre un problème multi-classe. La stratégie la plus classique est le « un contre tous » qui consiste à construire un SVM par classe.

Chaque classifieur est alors entraîné à distinguer les exemples de sa classe de ceux de toutes les autres classes. Une autre stratégie classique est le « un contre un » qui consiste à construire un SVM par paire de classes. Pour un problème à c classes, cette stratégie revient donc à entraîner $c(c-1)/2$ classifieurs binaires, mais comme il est montré dans [40], étant donné que chaque sous-problème est beaucoup moins complexe à résoudre, il est plus rapide d'entraîner l'ensemble des SVM de la stratégie « un contre un » que les c SVM de l'approche « un contre tous ». Par ailleurs, dans notre cas, il semble préférable d'utiliser la seconde stratégie qui est plus modulaire et qui permettra de se focaliser uniquement sur les p classes en conflit. En effet, l'utilisation de la première stratégie conduirait à calculer des distances aux vecteurs de support de classes improbables, ce qui aurait pour effet d'augmenter inutilement le coût de classification.

Chapitre 2 - Reconnaissance de l'Ecriture et du Document

Un document est le support physique pour conserver et transmettre de l'information. Selon le support choisi, un document peut être textuel, graphique, multimédia (sonore, vidéo). Avec l'apport de l'informatique, un grand nombre de documents actuels est en format numérique. Ces documents sont soit conçus et réalisés, dés le départ, par ordinateur, soit numérisés au moyen d'un scanner.

Dans le monde entier, il existe un nombre important de documents papiers. Ces derniers sont de différents types tels que les journaux, les revues, les livres, les encyclopédies, etc…

Afin de préserver ces documents de tout genre de détérioration ou d'éventuelle décomposition, qui pourraient survenir, on peut les conserver (stocker) sous forme numérique. Le coût de stockage et de duplication des documents électroniques est inférieur au coût de stockage et de duplication des mêmes documents au format papier.

2.1 Supports et formes de documents

La notion de document recouvre aujourd'hui plusieurs supports et formes (ou formats) qui coexistent. Nous distinguerons :

– les documents imprimés sur support papier, que nous appellerons forme papier d'un document : livres, journaux, revues, notes techniques, imprimés divers... qui sont de nature majoritairement textuelle.

– les documents électroniques, ou numériques, stockés sur support informatique, qui se divisent eux-mêmes en :

• documents électroniques en mode image, que nous appellerons forme image,

• documents électroniques codés, par exemple en ASCII, dits forme codée,

• documents électroniques codés et structurés (avec des marques de titre, paragraphe...), ou forme structurée.

2.1.1 Formes images

La gestion électronique de documents (GED) a jusqu'ici principalement utilisé la forme image, ou image numérique, obtenue après numérisation des documents à l'aide d'un scanner.

Celle-ci est une représentation de la page par une succession de pixels. Plusieurs codages des images sont utilisés. En GED, on utilise essentiellement un codage biniveau : chaque pixel est représenté par un seul bit qui prend deux valeurs : 1 ou 0 (noir ou blanc). Cependant, la reproduction fidèle des photos et des documents de qualité dégradée nécessite un codage en niveaux de gris, voire en couleurs.

2.1.2 Formes électroniques codées

La forme électronique codée d'un document est une simple séquence de caractères. Ces caractères peuvent être représentés dans différents espaces de codage, le plus classique étant le codage ASCII permettant 256 caractères différents. Afin d'avoir une représentation plus riche, notamment en ce qui concerne la représentation de différentes langues comme le chinois ou l'arabe, différentes normes de codages sont largement utilisées comme la norme UNICODE qui permet la représentation de 65536 caractères. Il y a d'autres codes décrivant la typographie et la mise en page.

2.1.3 Formes électroniques codées et structurées

La forme codée et structurée décrit en plus la structure logique du document, c'est-à-dire qu'elle inclut les éléments de texte dans une organisation logique : titres, chapitres, paragraphes, notes... La norme SGML (standard generalised markup language) [41] permet le codage des documents structurés.

En résumé, seule la représentation codée supporte des documents « vivants » et permet d'accéder véritablement à leur contenu. Cela tient au fait que pour les ordinateurs, elle donne accès aux informations signifiantes, alors que la forme image n'est interprétable que par les humains.

2.2 Analyse d'image de documents

L'analyse automatique de documents est un champ vaste d'applications potentielles compte tenu de la quantité énorme de documents produits par l'activité humaine. L'Analyse d'Image de Documents est un thème de recherche du domaine du traitement d'images numériques qui avait pour objectif principal de convertir des images de documents en vue de la modification, l'archivage, la réutilisation et la transmission de l'information que ces images contiennent. Le document papier, une fois converti sous forme électronique, permet une recherche par le contenu et une gestion beaucoup plus aisée. Ainsi Nagy [42] propose la définition suivante : ‘‘L'Analyse d'Images de Documents est une théorie et une pratique de reconstruction de la

structure symbolique des images numériques directement produites par l'ordinateur ou simplement numérisées à partir du papier''.

L'utilisation effective de système automatique d'analyse de documents papiers dépend évidemment des capacités à localiser les zones informatives et à les reconnaître. Or beaucoup d'applications potentielles sortent de ces limites : orientation variables des lignes, fond incontrôlé, écriture manuscrite, mélange de textes, de graphiques, de photographies. C'est le cas des documents techniques, des cartes et plans, des magazines où la mise en page peut être très complexe, ou encore des documents postaux où l'adresse peut n'être qu'une faible fraction de l'information présente.

Parallèlement, le document électronique passe par différentes formes : structure logique, structure physique, image et papier. La structure logique reflète le point de vue de l'auteur; elle permet de représenter l'organisation du document en entités telles que chapitres, sections, paragraphes, etc… La structure physique permet de représenter la structuration du document en vue de son impression; de ce fait certains critères, comme par exemple la découpe en page et la répartition des espaces, sont nécessaires.

La reconnaissance de documents aura comme entrée une image numérisée épurée et elle est composée de deux étapes successives, l'une pour la reconnaissance de structures physiques (ou segmentation) et l'autre pour la reconnaissance de structures logiques. La figure 2.1 illustre la reconnaissance de structures physiques et logiques d'un document.

Dès lors une étape essentielle d'un système complet d'analyse, en amont de la lecture elle-même, consiste à localiser et extraire le plus correctement possible toutes les entités à traiter par le système de reconnaissance.

La reconnaissance de structures physiques comprend la détection et la classification des différentes zones de l'image. Elle a pour objectif de délimiter toutes les régions d'intérêt de l'image. L'objectif de cette délimitation est de regrouper les régions en des zones homogènes : textes, graphiques et images. La finesse de cette décomposition de régions pour le cas du texte, par exemple, permet d'obtenir les lignes de texte, les mots et éventuellement les caractères.

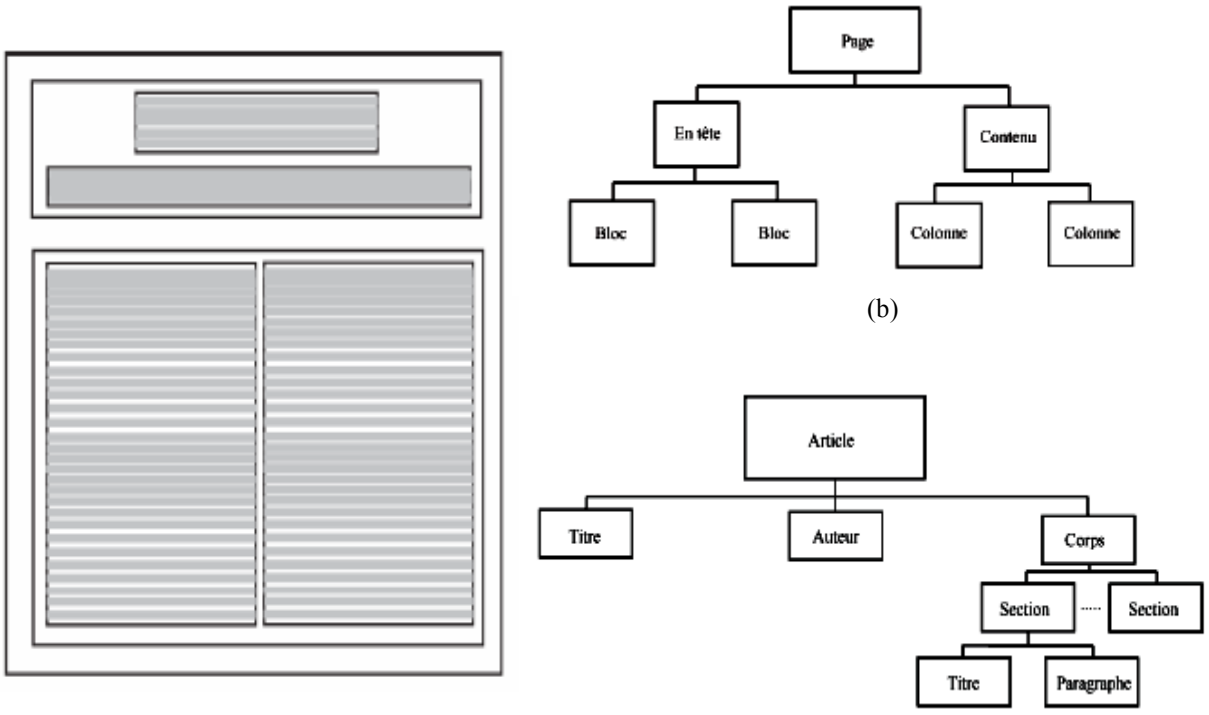


Fig 2. 1 – Reconnaissance de structures physiques et logiques d'un document : (a) page d'un document, (b) Structure physique, (c) Structure logique.

2.2.1 Segmentation de documents

Avant de passer à une analyse plus fine, il est nécessaire de donner au système l'équivalent de la vision globale d'un lecteur humain. Au niveau des pages, on appelle segmentation l'opération d'analyse globale qui consiste à découper l'image du document en régions ou blocs homogènes (texte, graphique, photographies...). À l'issue de cette opération, on soumet les différentes zones à des traitements spécifiques. En particulier, les blocs contenant du texte sont envoyés à un module de reconnaissance de caractères (OCR).

Les différentes approches de décomposition en régions sont soit descendantes, soit montantes voire hybrides.

- ***Approches descendantes***

Les approches descendantes divisent de manière itérative l'image du document en régions de plus en plus petites, et stoppent ce processus lorsqu'un critère d'arrêt est vérifié. Nous évoquons les méthodes de remplissage (RLSA) [43] et l'algorithme de découpage X-Y [44]. Wang [45] propose une méthode basée sur l'utilisation conjointe de l'algorithme RLSA et de l'algorithme de découpage X-Y récursif pour segmenter l'image en blocs.

- ***Les approches montantes***

Les approches montantes démarrent au niveau pixel de l'image en regroupant les pixels en composantes connexes, telles que des caractères, qui sont ensuite regroupées pour former des mots, des lignes ou des régions [46, 47].

L'avantage de cette approche est qu'elle s'adapte à toute mise en page, même incluant des blocs non rectangulaires ou imbriqués, et qu'elle tolère une certaine inclinaison des pages. Son inconvénient tient à la nécessité d'une connaissance a priori de la taille des caractères, à une complexité de mise en oeuvre et au besoin d'utiliser en complément une méthode projective (descendante) pour éviter d'agglomérer des colonnes de texte proches. On aboutit ainsi aux méthodes hybrides.

- ***Les approches hybrides***

Les approches hybrides peuvent être vues comme un mélange des deux approches précédentes. Pavlidis [48] a proposé une approche hybride utilisant une stratégie de type « split and merge». Un état de l'art sur les algorithmes de segmentation de page peut être trouvé dans [49] et dans [50].

2.3 Reconnaissance de documents

On appelle reconnaissance de documents le passage de la forme papier ou d'une forme image d'un document, à une forme électronique codée ou structurée. L'image d'un document est composée de régions homogènes telles que du texte, du graphique, des formules... Chacune d'entre-elles est soumise à un traitement spécialisé dépendant de son contenu. Toutes ces régions s'organisent de manière hiérarchique pour composer une structure.

La dernière étape de la reconnaissance est la reconnaissance des structures logiques. Son objectif est de déterminer l'organisation logique des entités retrouvées au niveau de la reconnaissance de structures physiques, en effectuant un étiquetage. Les étiquettes utilisées dans cette étape sont dépendantes de l'application visée et peuvent correspondre à titre, auteur, paragraphe et article. La reconnaissance de structures logiques comprend aussi le recouvrement de l'ordre de lecture. Pour un document composé d'une seule colonne, cet ordre est de haut en bas et de gauche à droite. Cependant, cet ordre est fortement dépendant de la langue dans laquelle le document est écrit.

2.3.1 Acquisition des images de documents

La première opération consiste à acquérir les images numériques de documents qui s'effectue généralement à l'aide d'un scanner, périphérique devenu très courant depuis l'explosion de la

bureautique dans les années 80. Les résolutions utilisées pour l'OCR sont au minimum de 300 dpi (dots per inch), voire 400 dpi pour les petits caractères.

Le choix d'utiliser des images binaires s'applique souvent à des images contenant des symboles graphiques ou du texte. Les photographies doivent être produites en niveau de gris. La couleur peut également être utilisée pour des images de magazines par exemple. Mais tous les pré-traitements de ces images pour les rendre interprétables automatiquement sont généralement indépendants du type de document.

2.3.2 Pré-traitement

Les pré-traitements consistent en des transformations d'images. L'objectif des pré-traitements est de faciliter la caractérisation de l'image à reconnaître soit en éliminant le bruit ou en réduisant la quantité d'information à traiter pour ne garder que les informations les plus significatives. La réduction de la quantité d'information à traiter peut être obtenue à partir des opérations visant à ramener l'épaisseur du trait à un seul pixel (soit par squelettisation [51], soit par suivi de trait [52]) ou à partir d'extracteurs de contours. Notons que certaines formes (caractères, chiffres, mots) sont inclinées ou penchées donc il est nécessaire de normaliser en pente cette forme afin de segmenter la forme (par exemple segmentation d'un mot en lettres). Cette normalisation consiste à corriger la pente d'un mot ou à redresser l'inclinaison des lettres dans un mot afin de faciliter la segmentation.

2.3.3 Redressement

L'un des problèmes le plus fréquent dans un système d'analyse et de reconnaissance de documents est la détermination de l'inclinaison du document. En général, l'inclinaison des images est provoquée essentiellement soit par un mauvais positionnement des pages lors de la saisie optique, soit par une mise en page fantaisiste et irrégulière de l'auteur. Pour une bonne exploitation des documents, il faut détecter cet angle et redresser les images, de telle sorte que les lignes de texte soient parallèles aux bords.

Parmi les techniques de détection de l'angle d'inclinaison les plus utilisées sont les suivantes. Belaid [53] utilise la méthode des moindres carrés pour évaluer l'inclinaison du texte. La méthode de projection [53, 54, 55] est basée sur le calcul de l'histogramme horizontal de l'image du document pour chaque angle appartenant à l'intervalle de détection.

La transformée de Hough [56, 57, 58, 59] est une technique de détection des lignes et des courbes dans une image. Elle est utilisée aussi pour détecter l'angle d'inclinaison avec un intervalle de détection compris entre 0° et 180°. La transformée de Hough a été testée sur des

documents arabes imprimés. Mais, le problème majeur était la lenteur des calculs. Ainsi et afin de palier à ce problème, la réduction de points à traiter est nécessaire, et ce sans altérer la précision dans la détection de l'angle d'inclinaison recherché pour la totalité des points de l'image.

La méthode des k-plus proches voisins [60, 61, 62] consiste à calculer l'orientation de la ligne qui relie chaque deux composantes connexes voisines, les orientations sont représentées dans un histogramme. L'angle d'inclinaison du document est déterminé à partir du pic de l'histogramme. Cette méthode est sensible au bruit et à l'écriture cursive particulièrement les documents arabes.

2.3.4 Binarisation

La binarisation permet de convertir l'image numérique en niveau de gris issue du scanner en une image bi-modale dans laquelle l'objet et le fond sont séparés. Lorsque les documents papier sont de bonne qualité et de fond blanc, un simple seuillage global suffit après analyse de l'histogramme des niveaux de gris. Tous les pixels plus lumineux que le seuil sont pris comme blancs, les autres comme noirs. Mais lorsque le fond est texturé ou si le document est dégradé (pliures, taches ...), une analyse plus fine est nécessaire. Pour une analyse et comparaison de différentes approches de seuillage, on pourra se référer aux travaux de synthèse dans [63, 64, 65].

- Seuillage global

Le seuillage global consiste à prendre un seuil identique pour toute l'image. Chaque pixel de l'image est comparé à ce seuil : ceux de niveaux de gris inférieurs sont mis à 0 «noir », ceux supérieurs mis à 1 «blanc».

Pour le seuillage automatique d'une image à niveaux de gris [66], une méthode consiste à tracer un histogramme de ces niveaux de gris, et choisir le seuil au fond de la vallée qui sépare le pic correspondant au niveau de gris du fond et le premier pic suivant (voir figure 2.2, image en polarité inverse).

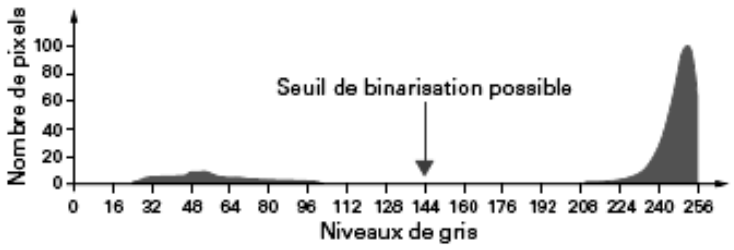


Fig 2. 2 – Choix du seuil sur l'histogramme de l'image à niveaux de gris.

Les limites de ce seuillage sont les suivantes :

— lorsque la qualité d'impression du texte n'est pas constante dans toute la page, des caractères peuvent être partiellement perdus

— lorsque le fond est bruité ou non homogène, des taches parasites peuvent apparaître ;

— lorsque la page contient des encarts de couleur (texte imprimé sur fond coloré), tout le texte de ces blocs risque d'être perdu ;

— les photos traitées de cette manière ne sont plus reconnaissables : elles sont rendues par une mosaïque de zones noires et blanches.

La solution à cet ensemble de problèmes est donnée par le seuillage adaptatif d'une part et la séparation des photos d'autre part

- Seuillage adaptatif

Le seuillage adaptatif consiste à faire varier le seuil localement, en fonction des niveaux de gris des pixels de document. Plusieurs solutions de seuillage adaptatif ont été proposées pour restituer correctement le texte dans tous les cas. On distingue des méthodes basées sur un découpage de l'image en régions et des méthodes locales proches du traitement du signal.

- Découpage de l'image en régions

Une première approche consiste à découper l'image en grille de petites zones rectangulaires, à tracer un histogramme des niveaux de gris dans chaque rectangle et à calculer un seuil adapté à chacun. Parmi les inconvénients de cette méthode : difficulté d'ajustement du seuil aux frontières des zones, et surtout non-adéquation a priori du quadrillage à l'information de la page.

- Méthodes locales proches du traitement du signal

Un autre type de solution consiste à déterminer le seuil de binarisation de chaque pixel à partir de propriétés locales du voisinage de ce pixel. Cela revient à utiliser une fenêtre glissante, et appliquer des opérateurs locaux, linéaires de type convolution ou non linéaires.

Cette solution pallie certains manques de l'approche précédente comme la fixité de la grille.

2.4 Reconnaissance de l'écriture

La reconnaissance de l'écriture a eu un grand essor de la part des chercheurs dès les années 1970 ; 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.

Il y a longtemps que ce thème se développe depuis l'apparition des premiers systèmes de lecture optique de caractères. Les premiers systèmes de lecture des adresses postales furent installés en 1965. La lecture automatique de l'écriture manuscrite et contrainte dans les formulaires commença à fonctionner dans les années 80. Ceci fût fortement conditionné par les progrès importants effectués en matière de possibilité de traitement par ordinateur, la progression spectaculaire des capacités de stockage de documents numériques et la réduction de leur coût.

L'archivage et la recherche d'information dans les bases de documents sont des problèmes centraux pour l'utilisateur. Les systèmes de traitement de l'information qui appréhendent ces problèmes nécessitent l'acquisition de l'information dans une représentation adéquate pour les bases de données. Cet aspect du domaine de la GED, appelé rétroconversion, a donc pour objectif de lire et classer partiellement ou en détail les informations figurant sur le support papier.

Pour les informations imprimées, dans un format simple ou connu à l'avance, les techniques OCR (Optical Character Recognition) ont apporté une solution satisfaisante.

L'analyse et l'interprétation d'un format quelconque de document fait toujours l'objet d'actives recherches en analyse de documents.

La lecture automatique des mentions manuscrites, appelée ICR (Intelligent Character Recognition) est l'autre difficulté de la rétroconversion. Le tri postal et la lecture des montants de chèques sont les applications les plus abouties de l'ICR et sont industrialisées depuis plusieurs années avec des améliorations successives, permettant d'interpréter un nombre plus grand de mentions, et avec une fiabilité croissante. On est ainsi passé de la lecture du code postal et de la ville, à celle de la ligne du numéro et du nom de rue. Dans le cas des chèques, les premiers systèmes se contentaient de lire le montant en chiffres, et les systèmes plus récents donnent des résultats plus fiables avec la lecture du montant en toutes lettres. Dans le cas de la lecture automatique de formulaires, l'OCR est utilisé depuis de nombreuses années pour la saisie des cases à cocher, des nombres puis des lettres pré-casées. La lecture des mots en majuscules jointives et même des mots cursifs commence à être utilisée.

2.4.1 Types d'écriture

Les premiers travaux sur l'écriture manuscrite se sont d'abord intéressés à la reconnaissance de chiffres et de caractères isolés. C'est le cas des formulaires pré-casés. Tappert et al. [67]

proposent de distinguer 5 classes d'écriture, de difficultés croissantes pour la reconnaissance, comme présenté dans la figure 2.3 :

— écriture bâton (en majuscules bien séparées) ;

— écriture scripte en lettres isolées ;

— écriture scripte jointive avec des lettres mal segmentées ;

— écriture cursive en lettres liées ;

— écriture mixte cursive et scripte.

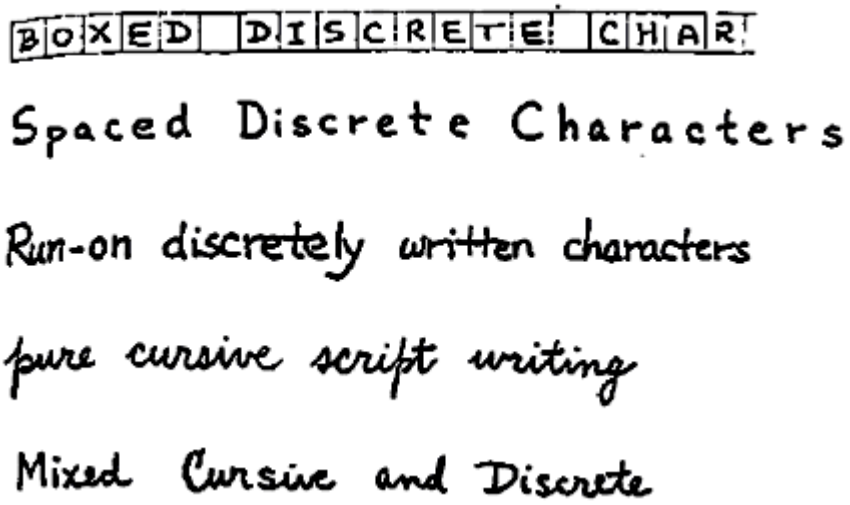


Fig 2. 3 – Exemples de l'écriture manuscrite d'après [67].

L'écriture cursive rend impossible une segmentation en caractères sans reconnaissance.

L'écriture mixte cursive et scripte rend plus aléatoire la segmentation en mots si l'espacement entre les mots n'est pas suffisamment marqué par rapport aux espaces entre les lettres.

En plus des qualités d'écriture, on classe usuellement les systèmes selon la variabilité et le nombre de scripteurs traités. Les premières recherches sur la lecture de l'écriture manuscrite se sont d'abord penchées sur des systèmes dédiés à un scripteur, puis à un groupe de scripteurs. Les applications dites industrielles traitent une écriture quelconque. Ces systèmes sont omniscripteurs, mais s'aident en général d'un vocabulaire, et d'une syntaxe bien définis, comme dans le cas très contraint des chèques et celui plus large des adresse postales.

D'une manière générale la complexité d'un système de reconnaissance de l'écriture s'évalue suivant trois critères orthogonaux [68] :

– *Disposition spatiale du texte :* la présentation d'un texte varie globalement entre deux formats : l'écriture *contrainte* correspondant à une écriture guidée par des cadres (les formulaires par exemple) et l'écriture *non-contrainte* correspondant à une écriture guidée

exclusivement par le scripteur donc extrêmement variable. Les écritures externes ou internes détachées (écriture en bâtons) sont, bien entendu, les plus aisées à traiter du fait de la séparation plus ou moins immédiate des lettres.

– *Nombre de scripteurs :* la difficulté de traitement croît avec le nombre de scripteurs. Trois catégories d'écritures se distinguent : les écritures *monoscripteurs*, *multiscripteurs* et *omniscripteurs*. En mode multiscripteurs, le système doit être capable de reconnaître l'écriture de plusieurs personnes prédéfinies, alors qu'en omniscripteur il doit s'adapter à n'importe qui.

– *Taille du vocabulaire :* Il existe deux types d'applications, celles qui sont à vocabulaire limité (<100 mots) et celles qui sont à vocabulaire très étendu (>10000 mots). La reconnaissance sera bien plus aisée dans le premier cas puisqu'il sera possible de comparer un mot inconnu avec la totalité des mots du dictionnaire. Dans le second cas il faudra bien souvent mettre en oeuvre des décisions hiérarchiques pour réduire le temps de calcul et l'encombrement mémoire.

La figure 2.4 présente un schéma synthétique de la complexité des systèmes RAED [68]. Plus l'application s'éloigne du centre du repère, plus la reconnaissance devient difficile.

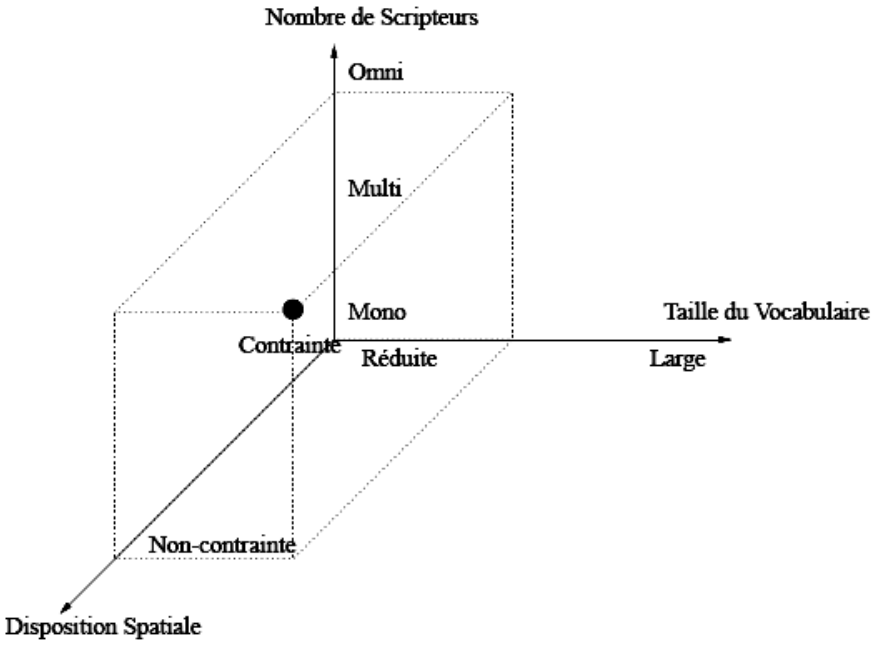


Fig 2. 4 – Graphe de complexité des systèmes de RAED d'après BELAID [68].

2.4.2 Classes des approches de reconnaissance de l'écriture

Le problème de la reconnaissance de l'écriture peut être abordé par deux approches différentes, approche hors-ligne et approche en-ligne, ces deux approches se diffèrent par la façon d'acquérir les tracés de l'écriture (en-ligne ou hors-ligne), et par la présence ou non des

informations décrivant l'ordre de construction de chaque tracé; donc les méthodes de reconnaissance se diffèrent d'une approche à une autre.

- ***Approche hors-ligne (off-line)***

Les systèmes de reconnaissance (hors-ligne) [69, 70] ou statiques sont des approches « images 2-D » et destinés à des applications qui ne nécessitent pas un traitement en temps réel telles que les applications bancaires ou postales.

Il s'agit de lire le montant manuscrit d'un chèque ou une adresse postale sur une enveloppe dans un centre de tri.

Les informations transmises au système seront les pixels d'une image acquise a l'aide d'un scanner ou camera (l'ordre d'arrivée des points n'a plus d'importance) on a aussi l'épaisseur du tracé qui est de plusieurs pixels. La recherche de primitives pour certaines techniques de reconnaissance peut alors nécessiter la détermination d'un tracé moyen (squelettisation).

- ***Approche en-ligne (on-line)***

La reconnaissance d'écriture en-ligne [71, 72] correspond à une approche signal et destinée à la bureautique, il s'agit de saisir un texte avec un stylo et une table à digitaliser pour le transmettre à un traitement de texte.
La reconnaissance en-ligne se fait en temps réel c'est à dire pendant l'écriture (le temps de reconnaissance dépend de la technique utilisée et de la vitesse de l'ordinateur).

Les informations se présentent sous forme d'une suite ordonnée de points définis par leurs coordonnées, l'ordre d'arrivée étant important pour la classification, notons toutefois que la connaissance de cet ordre peut être instable: deux scripteurs différents peuvent utiliser des règles différentes et un même scripteur peut revenir surcharger un tracé qu'il juge incomplet ou incorrect.

2.5 Architecture générale d'un système OCR (Optical Character Recognition)

C'est la partie la plus étudiée des systèmes de reconnaissance d'écrits car la plus ancienne et la plus simple quand il s'agit de caractères numériques ou latins (vocabulaire limité). Le dictionnaire est constitué d'images de caractères étiquetées. Une image de caractère inconnu pourra être étiquetée comme l'élément du dictionnaire dont elle est la plus similaire. La notion de similarité implique une description des caractères dans un espace de représentation muni d'une métrique.

Un système OCR se présente sous forme d'un ensemble de modules en partant de l'acquisition des caractères à reconnaître pour arriver à leur reconnaissance effective (figure 2.5). Dans ce qui suit, nous nous intéressons aux systèmes de reconnaissance hors-ligne.

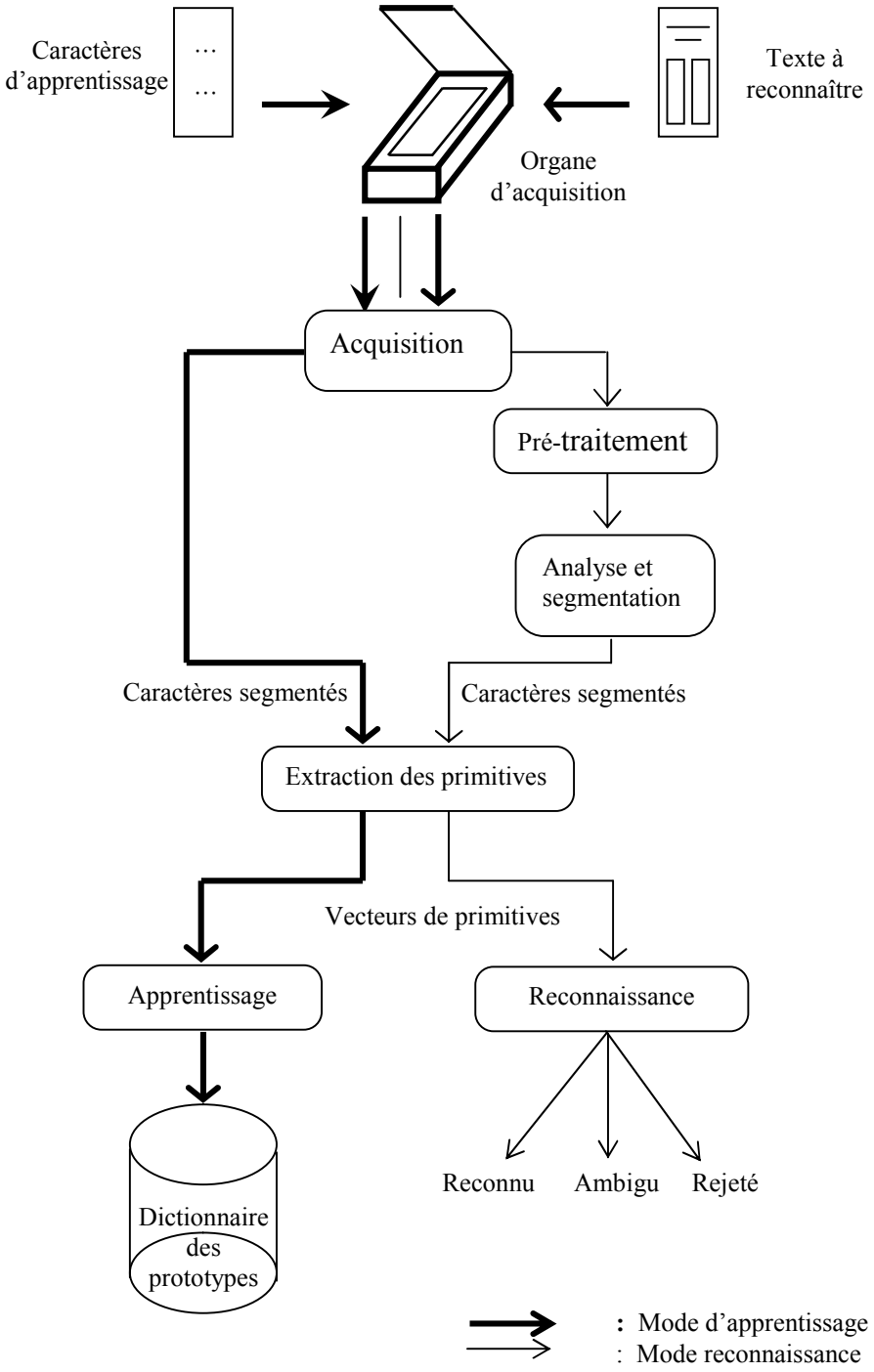


Fig 2. 5 – Système de reconnaissance de caractères (OCR).

2.5.1 Acquisition et pré-traitement

Comme nous nous intéressons essentiellement au traitement hors-ligne ou les systèmes d'acquisition actuels les plus courants sont donc essentiellement des scanners ou des caméras linéaires. Ils permettent en autre la saisie d'images en niveau de gris (jusqu'à 256 niveaux) voire en couleurs.

Le pré-traitement d'images consiste à éliminer le bruit dû aux conditions d'acquisition, et ne garder que l'information significative de la forme représentée. Si l'acquisition a été effectuée en niveau de gris ou en couleurs, la binarisation sera souvent l'un de ces pré-traitements.

Toutefois, il est à remarquer que certains auteurs [73, 74] ne passent pas par cette étape intermédiaire et extraient directement les primitives utiles à la reconnaissance à partir de l'image en niveau de gris.

Enfin on inclut aussi généralement dans les pré-traitements des opérations du type redressement des écritures penchées, détection et éventuellement redressement des lignes de bases.

2.5.2 Segmentation

On utilise essentiellement des images binaires, et on supposera le texte noir sur fond blanc.

A la suite des pré-traitements, il est nécessaire de segmenter l'image pour obtenir des entités élémentaires qui seront analysées dans la phase ultérieure. On peut distinguer deux types d'entités:

(i) Les entités physiques qui correspondent à des groupements de pixels délimités par un séparateur physique de type image [75]. L'entité de base est la composante connexe. Le séparateur est alors la transition blanc / noir ou noir / blanc.
(ii) Les entités logiques qui correspondent à des groupements de pixels délimités par un séparateur logique [76] défini à partir du modèle de l'écriture considéré.
L'entité de base est la lettre, elle ne correspond à une entité physique que dans le cas d'une écriture contrainte en lettres séparées. Mais peut également correspondre à un regroupement de parties de deux composantes connexes (Cas des *i* et *j* ayant des points supplémentaires, cas des lettres accentuées).

2.5.3 Interprétation de l'image du mot : reconnaissance de mots

Il y a trois façons d'aborder cette problématique [77]. Soit le système reconnaît le mot comme une entité entière et indivisible, il s'agit d'une approche *globale* ou *holistique*. Soit il

reconnaît le mot à partir de ses caractères préalablement segmentés, il s'agit d'une approche *analytique*. Soit il n'utilise que certaines propriétés et raffine sa description du mot par rebouclage, nous parlerons alors de *systèmes basés sur la lecture humaine*.

Approche globale

Cette approche reflète le processus humain de lecture qui ne se fait pas caractère par caractère, on s'intéresse à l'allure globale du mot (les lettres individuelles ne sont pas reconnues). Avec cette approche, on recherche des paramètres caractérisant l'allure globale du mot [70, 78]. Par exemple :

- L'allure des contours supérieurs et inférieurs.
- La description du mot sous forme d'un graphe reliant les points caractéristiques d'un squelette : extrémités de tracés, points extrêmes jonctions.
- L'axe d'un mot « le plus court chemin d'une extrémité à l'autre du mot qui reste dans le corps de ligne » et la reconnaissance se fait sur les éléments extérieurs à cet axe.

Comme le mot est une forme plus complexe que le caractère, la description contiendra plus d'informations et sera ainsi moins sensible aux changements de scripteurs. Néanmoins le vocabulaire étant plus large, l'apprentissage nécessite beaucoup d'individus ce qui rend, en pratique, cette approche inutilisable. Pour cette raison, les approches globales sont souvent utilisées pour d'autres tâches. Elles servent, par exemple, à réduire la taille du lexique ([79]) ou à valider une reconnaissance analytique ([80]).

Approche analytique du mot

Par opposition à l'approche précédente, celle-ci tente de retrouver l'entité lettre dans le mot.

La difficulté de cette approche a été explicitée par SAYRE [81] : « pour reconnaître les lettres, il faut segmenter le tracé et pour segmenter le tracé, il faut reconnaître les lettres ». Il convient donc d'adopter une méthode itérative qui raffine la segmentation en fonction des hypothèses de classification. Il y a deux façons de résoudre ce problème.

La première consiste à « disséquer » ([82]) le mot en graphèmes et utiliser la reconnaissance de ces graphèmes pour corriger les erreurs de segmentation (figure 2.6).

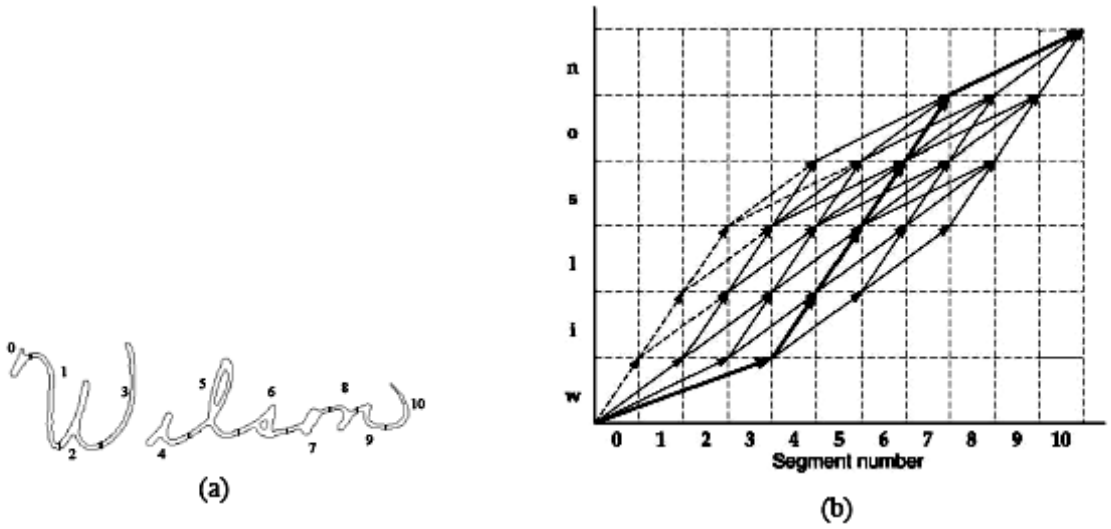


Fig 2. 6 – Reconnaissance analytique de mots basée graphème, (a) dissection, (b) schéma de la reconnaissance du mot wilson par appariement dynamique des graphèmes ([79]).

La seconde consiste à scanner l'image du mot pour y reconnaître des lettres puis à utiliser un dictionnaire afin de valider la reconnaissance (figure 2.7).

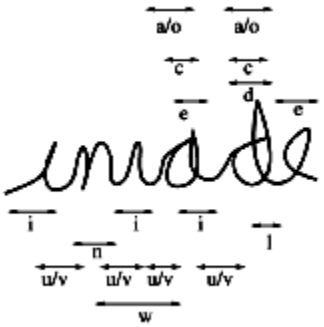


Fig 2. 7 – Reconnaissance analytique de mots basée sur la reconnaissance de lettres ([79]).

Les systèmes basés sur la lecture humaine reposent sur le principe de la supériorité du mot (*word superiority effect*, [83]). Ce principe veut qu'une lettre soit plus facile à reconnaître dans un mot que seule. Un effet secondaire de ce principe, que l'on a tous expérimenté, est la capacité humaine de reconnaître un mot alors même que quelques unes de ses lettres sont inversées. Il apparaît donc que la perception de formes particulières dans un mot suffit à sa lecture. La figure 2.8 donne un exemple de description simple, basée sur les boucles, les hampes et les jambages.

Tous les mots se représentent de manières différentes excepté « five » et « four » qui présentent une ambiguïté nécessitant un raffinement dans la détection de primitives.

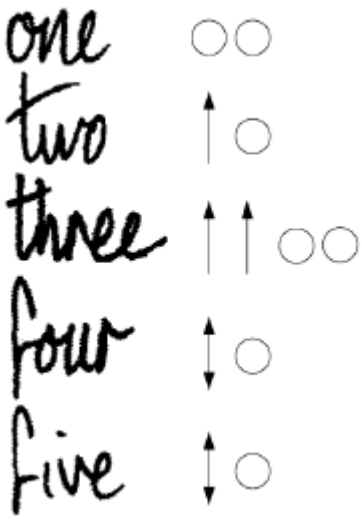


Fig 2. 8 – Exemple de reconnaissance basée sur la lecture humaine ([84]).

2.5.4 Extraction des caractéristiques

Au lieu d'alimenter directement le module de reconnaissance avec l'image du caractère, il est préférable de la paramètrer grâce à l'extraction de certaines de ses caractéristiques.

Cette étape de la reconnaissance consiste à extraire des caractéristiques permettant de décrire de façon non équivoque les formes appartenant à une même classe de caractères tout en les différenciant des autres classes.

Trier et al [85] ont précisé que la sélection et la définition des primitives lors de l'extraction est la partie la plus importante dans le système de reconnaissance, car le choix de celles-ci dépend du type de problème à résoudre.

Il existe différentes méthodes d'extraction de primitives citées dans la littérature. Trier et al [85] décrivent des méthodes d'extraction en fonction de la représentation du caractère.

Cette extraction se fait sur des images en niveaux de gris, en binaire, en contour binaire et selon une représentation vectorielle. Les primitives sont classées en deux catégories [86] : les primitives structurelles (ou primitives locales) basées sur une représentation linéaire du caractère (décomposition du caractère en segments de droites et courbes, contours du caractère, squelette) et les primitives statistiques (ou primitives globales) basées sur la distribution des points, les transformations et les mesures physiques faites sur le caractère (surface, moments, projections). Ces deux types peuvent être combinés formant un vecteur de primitives.

Primitives locales

Ces primitives sont des objets de la forme tel que les extrémités, les croisements de traits, les boucles et les courbes. L'inconvénient de ces primitives est que leur extraction nécessite une squelettisation préalable du caractère, puisque l'épaisseur du trait ne contient pas d'information. Néanmoins ce sont des primitives très robustes vis à vis de la rotation, translation, homothétie [87]. Dans cette catégorie, il existe 4 familles de caractéristiques : intersections avec des droites, arcs concaves et occlusions, extremas et jonctions [86].

Primitives globales

Ces primitives sont dérivées de la distribution des pixels. Heutte et al [86] suggèrent 3 familles de caractéristiques telles que : les moments invariants, les projections, et les profils. Elles sont extraites en considérant la distribution des pixels noirs de l'objet (caractère, mot, chiffres).

Le processus d'identification de la meilleure méthode d'extraction de caractéristiques n'est pas évident. Par exemple, Trier et al [85] rapportent que les moments de Zernike s'appliquent bien sur des images à niveaux de gris et que la projection s'applique souvent sur des caractères segmentés pour résoudre leur problème de reconnaissance de l'écriture manuscrite.

Cependant, il est nécessaire d'effectuer pour chaque problème de reconnaissance une évaluation expérimentale de quelques méthodes d'extraction de primitives les plus prometteuses. Ces expériences permettront de faire un choix judicieux des primitives à extraire car souvent, l'utilisation d'une seule méthode d'extraction de caractéristiques n'est pas suffisante pour obtenir de bonne performance du système de reconnaissance.

La solution évidente est de combiner plusieurs méthodes d'extraction afin de donner une meilleure description de la forme (caractère, chiffre, mot) à classer.

2.5.5 Apprentissage

Cette étape permet de construire un dictionnaire de prototypes. Il s'agit de regrouper plusieurs prototypes dont les primitives se rapprochent en classes.

L'apprentissage peut se faire grâce à un système expert tel que le système de reconnaissance de chiffres manuscrits sans contrainte [78]. Notons aussi que les réseaux de neurones ont attiré, grâce à leur capacité d'apprentissage, l'attention de la communauté de recherche en OCR. A partir de l'étape d'apprentissage, le système doit ajuster ses paramètres afin de donner une réponse lors de la phase de reconnaissance.

2.5.6 Reconnaissance

Le processus de reconnaissance peut toujours se résumer à une décision de classification. Pour cela, il faut choisir une représentation qui permettra une description de l'objet analysé, puis une règle de décision qui s'appuie sur cette description. On peut distinguer trois types de représentation.

Les représentations statistiques basées sur un vecteur de caractéristiques mesurées sur les formes élémentaires résultant de la segmentation. Le processus de classification correspond alors à une partition de l'espace associé à la représentation choisie (une décision basée généralement sur un critère de probabilité d'appartenance à une classe : règle de décision bayesiennes, règle du plus proche voisin…etc) [88].

Les représentations syntaxiques basées sur une grammaire formelle [89], la reconnaissance correspond alors à la comparaison de la phrase analysée, formée par la concaténation des symboles représentatifs des formes élémentaires avec les phrases du langage L(G) engendré par G.

Les représentations structurelles basées sur un graphe mettant en évidence les relations entre les formes élémentaires, la reconnaissance est alors un problème d'isomorphisme de graphes qui est souvent ramené à un problème de comparaison de chaînes de caractères codant ces graphes. La méthode de comparaison la plus répandue est alors un calcul de distance d'édition par programmation dynamique.

Les représentations statistiques ont connu ces dernières années un regain d'intérêt avec d'une part les approches « neuronales » et d'autre part les approches Markoviennes [90,91]. Ces deux types de méthodes ont permis de contourner deux difficultés des approches antérieures.

Les réseaux neuronaux facilitent la construction de la fonction de coût qui permet la décision, élaborent cette fonction à partir d'exemples, et permettent de la choisir dans une classe de fonction plus vaste.

Dans le domaine de reconnaissance de l'écriture manuscrite, les approches neuronales se sont essentiellement concentrées sur la reconnaissance de lettres et ou symboles manuscrits isolés et plus particulièrement sur la reconnaissance des chiffres [92].

Quelque soit la méthode de classification utilisée, ses performances dépendront beaucoup du choix de la représentation associée, et des caractéristiques (l'information utile pour la classification) extraites de l'image qui en découlent.

Plus le travail effectué à ce niveau aura été important, plus le processus ultérieur de classification se trouvera simplifié. A contrario, certains auteurs n'utilisent que des caractéristiques du type masque.

La reconnaissance se termine par une décision qui peut être :

Caractère reconnu : Si le système arrive à associer un et un seul prototype au caractère à reconnaître, il prend une décision unique.

Caractère ambigu : Si le système associe plusieurs prototypes au caractère à reconnaître, il propose ainsi plusieurs choix avec des confiances de même ordre.

Caractère rejeté : Si le système n'arrive à associer aucun prototype au caractère à reconnaître, il ne prend aucune décision de classification.

On définit ainsi plusieurs facteurs de performance de système tels que :

Le taux de reconnaissance qui représente le pourcentage de caractères reconnus parmi les caractères présentés.

Le taux de substitution qui représente le pourcentage de caractères acceptés par le système mais classé de façon incorrecte (à cause du problème d'ambiguïté).

Le taux de rejet qui représente le pourcentage de caractères rejetés (non reconnus) parmi les caractères présentés.

2.5.7 Post-traitement

Une étape de post-traitement peut être rajoutée à un système OCR pour améliorer le taux de reconnaissance en introduisant des informations contextuelles permettant de lever l'ambiguïté dans la reconnaissance de certains caractères, ces informations peuvent se présenter sous forme :

De connaissance pragmatiques sur la largeur moyenne de chacune des lettres ou sur le nombre de lettres constituant un mot (un code postal. Par exemple, est constitué de cinq chiffres).

D'algorithmes de correction orthographique mettant en œuvre des distances d'édition déterministes ou probabilistes.

De connaissances linguistiques de différents niveaux :

(i) lexical : pour valider la reconnaissance effectuer en ne retenant que des mots du dictionnaire et en rejetant les lettres de listes de lettres inconsistantes.

(ii) Syntaxique et sémantique : pour réduire la liste des mots candidats et valider ceux qui ont été retenus à l'étape précédente.

2.6 Sélection des primitives

Les caractéristiques permettent de distinguer une forme appartenant à une classe par rapport aux formes des autres classes. Il est important de bien définir les caractéristiques à extraire d'un objet pour sa reconnaissance. Il existe une très grande variété de caractéristiques mesurables sur des images et trop souvent on pense que chaque caractéristique est importante pour discriminer une forme d'une autre. Dans certains cas, il existe quelques caractéristiques qui n'aideront pas à discriminer entre les classes. Autrement dit, il existe des caractéristiques redondantes ou non pertinentes. Ces caractéristiques seront inutiles dans la classification de l'objet d'où l'utilité d'effectuer la sélection des primitives les plus pertinentes [93, 94]. Cette sélection a parfois un impact considérable dans l'efficacité des résultats de la classification [95].

L'objectif principal de la sélection des primitives est la réduction du nombre de caractéristiques utilisées tout en essayant de maintenir ou d'améliorer la performance de la classification du système de reconnaissance.

Nous rappelons brièvement les principes de méthodes classiques d'analyse de données.

2.6.1 Analyse en Composantes Principales

L'analyse en composantes principales (ACP) - Principal Component Analysis (PCA) - est une ancienne approche, qui effectue une réduction de dimension par projection des points originaux dans un sous-espace vectoriel de dimension plus réduite.

Les nouveaux vecteurs de caractéristiques sont obtenus par la transformation linéaire :

$Z = U^T X$, où X est une matrice à d lignes et à N colonnes qui contient l'ensemble de tous les vecteurs de primitives disponibles, et U une matrice de transformation d x d, réalisant la projection des données selon un nouveau système d'axes orthonormés ($UU^T = I$).

Il est aisé de montrer que minimiser la somme des carrés des distances des points qui représentent les données par rapport aux nouveaux axes est équivalent à maximiser la dispersion des données selon chacun d'entre eux.

La matrice U qui permet d'obtenir la meilleure représentation possible de données est la matrice des vecteurs propres de la matrice d'autocorrélation des variables XX^T [96].

Chaque vecteur propre correspondant à un axe de projection et la valeur propre qui lui est associée est en outre représentative de l'amplitude de la dispersion des données selon cet axe (figure 2.9).

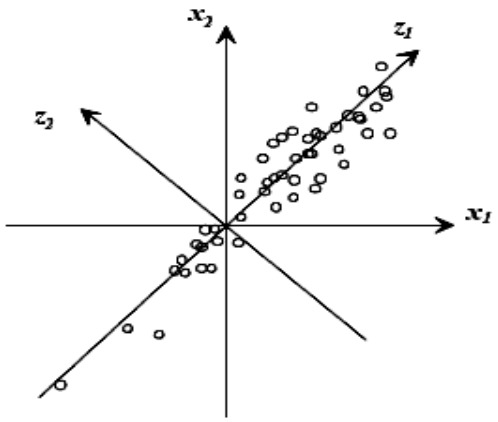


Fig 2. 9 – Analyse en composantes principales : axes de représentation des données avant (x_1, x_2) et après (z_1, z_2).

Dans la base formée par ces axes, les coordonnées ne sont pas corrélées. L'ACP maximise la variance de la projection dans l'espace de caractéristiques, ce qui est équivalent à minimiser l'erreur quadratique moyenne de reconstruction.

L'ACP se calcule en diagonalisant la matrice de corrélations, le plus souvent en utilisant une décomposition en valeurs singulières (SVD). Elle est très utilisée car elle est simple à mettre en œuvre, mais elle est limitée par son caractère linéaire.

Il est préférable d'effectuer l'ACP sur des caractéristiques centrées et réduites. Cette dernière porte le nom d'analyse en composantes principales normée [96].

La matrice de transformation U est à présent obtenue par la recherche des vecteurs propres de Y^TY , où Y est la matrice des caractéristiques centrées, réduites, et dont les éléments sont calculés selon:

$$y_{ij} = \frac{(x_{ij} - \mu_j)}{\sigma_j \sqrt{N}} \qquad (2.1)$$

où x_{ij} représente la valeur de la caractéristique j pour le $i^{ème}$ objet disponible, alors que μ_j et σ_j sont respectivement la moyenne et l'écart-type, estimés à partir de l'ensemble de tous les vecteurs de caractéristiques.

A l'issue de l'analyse en composantes principales, les meilleurs axes de représentation sont ceux selon lesquels la dispersion des données est la plus grande. Une réduction de la dimension de l'espace de représentation est donc réalisée, avec le moins d'erreur possible, en éliminant les axes dont les valeurs propres associées sont les plus faibles.

Toutefois, un axe qui offre une bonne représentation des données n'est pas forcément un axe qui en permet une bonne classification. La figure 2.10 en illustre un exemple extrême pour un problème à deux dimensions: l'axe ***z2*** permet une classification linéaire parfaite, au contraire de l'axe ***z1*** , qui offre pourtant une bien meilleure qualité de représentation des données. Afin de permettre la meilleure classification possible, un autre critère que celui de la valeur propre doit donc être utilisé pour sélectionner les nouveaux axes de représentation des données.

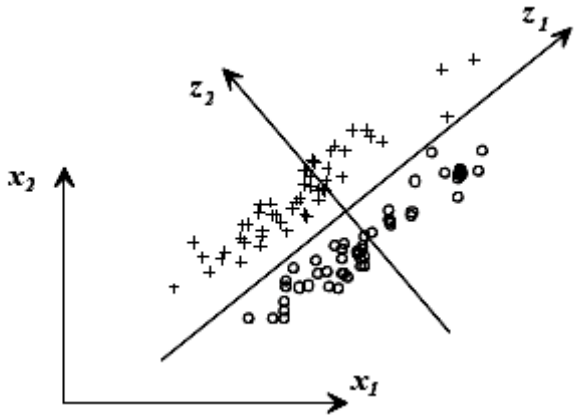


Fig 2. 10 – Illustration du problème de la réduction adéquate de la dimension de représentation: l'axe z1 est celui qui offre la meilleure qualité de représentation des variables, mais c'est l'axe z2 qui en permet la classification.

2.6.2 Analyse Discriminante

Proposée par Ronald A. Fisher en 1936, l'Analyse Factorielle Discriminante - Fisher Discriminant Analysis (FDA) - appelée aussi analyse discriminante linéaire de Fisher, s'applique lorsque les classes des individus sont connues. Elle consiste à calculer tout d'abord les covariances inter-classes et les covariances intra-classes des caractéristiques.

Matrices de covariances intra-classes :

$$S_W = \sum_{i=1}^{C} p(w_i)\Sigma_i \qquad (2.2)$$

Matrices de covariances inter-classes :

$$S_B = \sum_{i=1}^{C} p(w_i)\left[(\mu_i - \mu)(\mu_i - \mu)^T\right] \qquad (2.3)$$

où $p(w_k)$ est la probabilité a priori de la classe w_k.

$\sum_i$ représente la matrice de covariance des vecteurs de caractéristiques de la classe w_i.

μ_i étant le vecteur de caractéristiques moyen de la classe w_i.

μ est le vecteur de caractéristiques moyen global.

La LDA consiste à chercher un espace vectoriel de faible dimension qui maximise une certaine mesure du rapport entre les covariances inter-classes et les covariances intra-classes des nouvelles caractéristiques.

Les nouveaux vecteurs de caractéristiques sont obtenus par la transformation linéaire :

$Z = W^T X$, où W est une matrice de projection d x D ($D \leq d$).

Une simple mesure de ses covariances est donnée par le déterminant des matrices correspondantes, et le critère à maximiser est le suivant :

$$J(W) = \frac{W^T S_B W}{W^T S_W W} \tag{2.4}$$

La solution de ce problème de maximisation consiste à prendre comme colonnes de W les vecteurs propres généralisés associés aux plus grandes valeurs propres [97]:

$$S_B W_j = \lambda_j S_W W_i \tag{2.5}$$

La différence principale entre la LDA et la PCA est expliquée dans la figure 2.11.

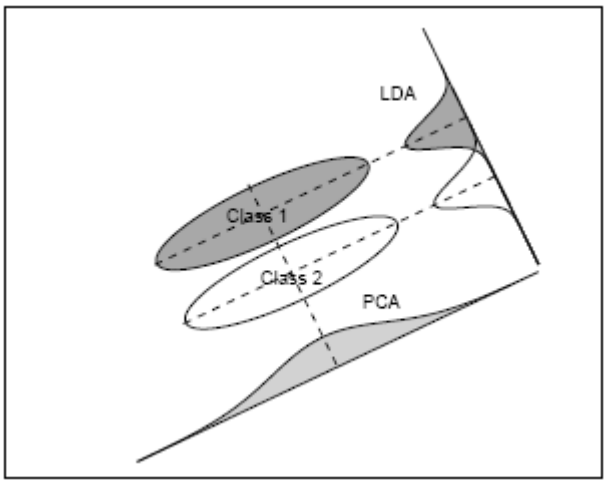


Fig 2. 11 – La différence entre la LDA et la PCA.

La PCA est utilisée parce qu'elle sert à décorréler les caractéristiques. La LDA augmente la discrimination entre les classes à distinguer.

2.6.3 Estimation de la puissance de discrimination : critère de Fisher

Il est évident que les caractéristiques seront d'autant plus discriminantes que leur dispersion inter-classes sera élevée vis-à-vis de leur dispersion intra-classes.

La prise en considération de uniquement les éléments diagonaux de S_B et S_W permet d'obtenir un critère simple pour estimer la puissance moyenne de discrimination d'une caractéristique déterminée [98] :

$$D_k = \frac{\sum_{i=1}^{C} p(w_i)(\mu_{ik} - \mu_k)^2}{\sum_{i=1}^{C} p(w_i)\sigma_{ik}^2} \tag{2.6}$$

où $\mu_{ik} = E[x_k / w_i)$ et $\sigma_{ik}^2 = E[(x_k - \mu_{ik})^2 / w_i]$ représentent respectivement la moyenne et la variance de la caractéristique *k*, calculées sur l'ensemble des éléments de la classe w_i .

Ce critère est connu sous le nom de Critère de Fisher Généralisé.

Une procédure d'analyse en composantes principales normée peut être effectuée avant de procéder au calcule de la puissance moyenne de discrimination. Les caractéristiques non seulement décorrélées, mais aussi centrées et réduites :

$$\begin{cases} \mu_k = 0 \\ \sigma_k = 1 \end{cases} \quad \forall\ k = 1,2,.....,d \tag{2.7}$$

Le critère D_k peut s'exprimer :

$$D_k = \frac{1}{\overline{\sigma_k^2}} - 1 \tag{2.8}$$

avec $\overline{\sigma_k^2} = \sum_{i=1}^{C} p(w_i)\sigma_{ik}^2$, l'ajout ou le retrait d'une constante ne modifiant en rien les valeurs relatives des pouvoirs de discrimination, ces dernier peuvent s'exprimer:

$$D_k = \frac{1}{\overline{\sigma_k^2}} \tag{2.9}$$

La puissance de discrimination d'une caractéristique est alors tout simplement inversement proportionnelle à la moyenne de sa variance selon chaque classe.

Chapitre 3 - Reconnaissance de l'Ecriture Arabe

La langue arabe est la langue officielle dans tous les pays arabes. Elle est utilisée dans la plupart des écrits et, à l'oral, dans les situations officielles ou formelles (discours religieux, politiques, journaux télévisés). Notons également que le farsi (persan), utilisé principalement en Iran et en Afghanistan, partage un grand nombre de caractéristiques communes avec l'écriture arabe.

3.1 Caractéristiques de l'écriture arabe

L'Arabe se distingue des autres écritures à différents points de vue. L'une des différenciations les plus importantes provient de sa semi-cursivité aussi bien dans sa forme imprimée que manuscrite. Elle est cursive, c'est-à-dire que les lettres sont liées généralement entre elles. 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'Arabe imprimé est connu pour sa richesse en fontes et styles, il existe plus de 450 styles et fontes différents. D'une fonte à une autre, les caractéristiques morphologiques du caractère arabe changent considérablement. Dans [99,100] nous trouvons une étude assez détaillée sur les caractéristiques morphologiques de l'Arabe.

L'écriture arabe s'écrit de droite à gauche. En revanche, la plupart des lettres s'attachent entre elles, même en imprimerie, et leur graphie diffère selon qu'elles sont précédées et/ou suivies d'autres lettres ou qu'elles sont isolées (variantes contextuelles). La ligne d'écriture sur laquelle s'alignent les caractères s'appelle ligne de base.

L'alphabet arabe est plus riche que son équivalent latin, Il contient 28 lettres dont la plupart changent de formes selon leur apparition au début, au milieu ou à la fin du mot.

Certaines lettres prennent jusqu'à quatre formes différentes : par exemple la lettre (ع) a quatre formes d'apparitions : isolée «ع », au début «عـ», au milieu «ـعـ » et à la fin «ـع ».

Le tableau 3.1 présente les 28 lettres arabes avec leurs différentes formes d'apparitions dans un mot.

EF	MF	BF	IF	EF	MF	BF	IF
ضـ	ضـ	ضـ	ض	ـا			أ
طـ	طـ	طـ	ط	ـب	ـبـ	بـ	ب
ظـ	ظـ	ظـ	ظ	ـت	ـتـ	تـ	ت
عـ	عـ	عـ	ع	ـث	ـثـ	ثـ	ث
غـ	غـ	غـ	غ	ـج	ـجـ	جـ	ج
فـ	فـ	فـ	ف	ـح	ـحـ	حـ	ح
قـ	قـ	قـ	ق	ـخ	ـخـ	خـ	خ
كـ	كـ	كـ	ك	ـد			د
لـ	لـ	لـ	ل	ـذ			ذ
مـ	مـ	مـ	م	ـر			ر
نـ	نـ	نـ	ن	ـز			ز
هـ	هـ	هـ	ه	ـس	ـسـ	سـ	س
ـو			و	ـش	ـشـ	شـ	ش
ـي	ـيـ	يـ	ي	ـص	ـصـ	صـ	ص

Tableau 3. 1– Les 28 lettres arabes avec leurs différentes formes d'apparitions dans un mot.

Les lettres qui ont juste deux formes d'apparitions ne peuvent pas être liées à la lettre suivante, leur forme de début est simplement leur forme isolée et leur forme de milieu est exactement celle de fin. Mais pour la plupart des lettres, les formes début/milieu et fin/isolé sont identiques à la ligature près. La présence d'une ligature avec la lettre précédente ou avec la lettre suivante ne modifie pas la forme de la lettre de manière significative. En arabe, les ligatures se situent toujours au niveau de la ligne d'écriture (ligne de base), c'est à dire qu'il n'existe pas de lettre à liaison haute comme le 'o' ou le 'v' en alphabet latin. Six lettres de l'alphabet ne s'attachent jamais à leur successeur : (ا, د, ذ, ر, ز, و). Ces lettres introduisent donc une coupure dans le mot, d'où la notion de pseudo mot. Le pseudo mot est donc une chaîne de caractères connectés, le mot arabe peut ainsi être composé d'un ou de plusieurs pseudo mots. Un pseudo-mot est alors une composante connexe de pixels noirs regroupant une ou plusieurs lettres. Un mot peut être composé d'un ou plusieurs pseudo-mots (voir figure 3.1).

La longueur de l'espace inter-mot est généralement supérieure à l'espace intra-mot ; entre caractères non attachés.

Fig 3. 1 – Un mot peut être composé de plusieurs composantes connexes (pseudo-mots).

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.

Lorsqu'une des 22 lettres (28 moins les 6 qui ne se lient pas avec la suivante) apparaît dans sa forme "fin de mot" ou "isolée", cela signifie obligatoirement que l'on arrive à la fin d'un mot.

3.1.1 Les signes secondaires

Dans certains travaux, tous les signes secondaires sont appelés diacritiques, qu'il s'agisse des voyelles, des points ou des autres signes (chadda, madda, hamza, ...).

Dans notre thèse, les "signes diacritiques" désigneront tous les signes secondaires en excluant les voyelles qui représente une classe de secondaires indépendante.

- ***Les voyelles***

Contrairement au latin, les voyelles ne sont pas utilisées systématiquement dans l'écriture arabe; des signes qui correspondent à des voyelles sont employés pour éviter des erreurs de prononciation.

Cela se traduit par le fait que toutes ces lettres, à l'exception du Alif 'ا', sont des consonnes. Le Waw و et le Ya ى sont, elles, des demi-voyelles, dans la mesure où elles représentent à la fois une consonne et une voyelle.

L'absence de voyelles peut toutefois être source de confusions. Comme le rappellent Aloulou et al dans [101], un mot peut avoir plusieurs voyellations possibles. Dans certains cas, une phrase peut donc avoir deux voyellations différentes, ce qui nous donne deux structures syntaxiques possibles.

Les voyelles peuvent parfois être mentionnées sur certaines lettres pour lever l'ambiguïté et faciliter la lecture. Mais en général, les scripteurs les omettent, et c'est au lecteur qu'est réservé le soin d'interpréter correctement le sens de la phrase en fonction du contexte.

On peut distinguer deux types de textes : les textes avec ou sans les signes de voyelles. Quelques textes arabes (Le Coran et les livres d'apprentissage de la lecture et de l'écriture pour les enfants) contiennent des signes de voyelles. Par contre, les livres, les journaux, les publications sont des textes sans ces signes.

- ***Les signes diacritiques***

Il existe des lettres différentes qui ont la même forme, mais qui se distinguent par la position et le nombre de points qui leur appartiennent. Dans l'alphabet arabe, 15 lettres parmi les 28 possèdent un, deux ou trois points. Ces point sont situés soit au-dessus, soit en dessous de la forme à laquelle ils sont associés, mais jamais les deux à la fois. Par exemple, les trois différentes lettres (ب, ت, ث) ont la même forme de base mais la position et le nombre de points sont différents.

Néanmoins, en manuscrit arabe, deux points peuvent être écrits comme deux points connectés (segment droit). Un groupe de trois points peut s'écrit comme deux poins connectés et un point isolé, tringle ou une courbe ouverte. La figure 3.2 illustre la variabilité des styles d'écriture des points en écriture manuscrite arabe.

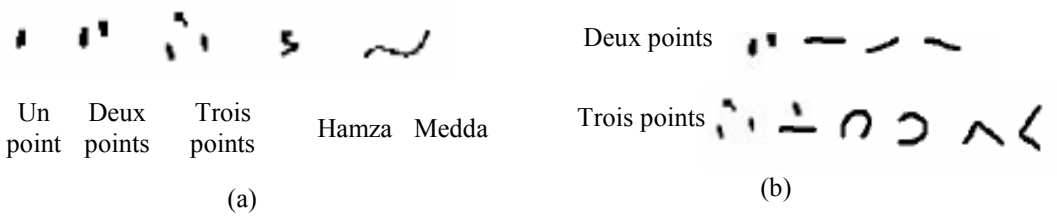


Fig 3. 2 – Signes diacritiques: (a) différentes formes des diacritiques, (b) variabilité des styles des points diacritiques.

Les autres signes diacritiques sont la hamza, la chadda et la madda. La chadda est une accentuation de la lettre (c'est l'équivalent d'une consonne doublée en français). Hamza et madda suivent des contraintes morphosyntaxiques plus complexes.

3.1.2 Ligatures et chevauchements verticaux

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 de base.

Les caractères d'une même composante connexe peuvent être ligaturé horizontalement ou verticalement. Deux jusqu'à quatre caractères immédiatement voisins peuvent être combinés verticalement pour donner naissance à des dessins assez particuliers (appelés ligatures verticales) dont la morphologie est complètement différente de celle des caractères qui les

constituent. Ceci rend la segmentation a priori en caractères quasi-impossible. La figure 3.3 montre une ligature verticale de trois caractères Laam (ـلـ), Miim (ـمـ)et Jiim(ـجـ).

Finalement, les chevauchements verticaux peuvent se produire par l'intersection des descendants qui se prolongent horizontalement sous la ligne de base et les composantes connexes suivantes (pseudo-mots, voir figure 3.3).

Les chevauchements et ligatures dépendent de la fonte utilisée et du style d'écriture du scripteur.

Fig 3. 3 – Chevauchements et ligatures.

Nous retenons également la forte dépendance du dessin d'un caractère de son contexte, la complexité et la multiplicité des graphies des lettres, la variabilité des liaisons inter-caractères aussi bien horizontales que verticales.

3.2 Différentes approches et systèmes existants

La majorité des approches de reconnaissance proposées a été testée sur l'écriture latine puis appliquée telle quelle pour la reconnaissance de l'écriture arabe. Les deux principales méthodes qui ont été utilisées sont l'approche globale et l'approche analytique. La première modélise le mot dans sa globalité. Elle présente l'avantage de garder le caractère dans son contexte de voisinage mais son inconvénient est qu'elle est limitée à la reconnaissance de vocabulaire très réduit et statique [102, 103]. L'approche analytique quand à elle se base sur la modélisation de l'alphabet de la langue et la segmentation du mot en entités représentant un caractère ou un pseudo-caractère. La reconnaissance du mot consiste à identifier ces entités et à proposer des hypothèses de mots. Cette approche est très liée aux résultats de la segmentation [104] et elle présente l'avantage de pouvoir manipuler un vocabulaire ouvert.

Les surveys suivants donnent un état de l'art sur les travaux déjà faits pour la reconnaissance de l'écriture Arabe.

Adnan Amin [105] analyse différentes approches pour la reconnaissance automatique de l'écriture arabe. Il présente les approches On-Line/OffLine, ainsi que les approches Globale/Analytique. Il pose quelques unes des problématiques fondamentales de la reconnaissance de l'écriture arabe (alphabet, signes diacritiques, styles d'écriture, codage de

l'écriture, pseudo-mots, ...). Finalement, l'auteur conclut que le sujet reste ouvert et de nombreuses voies d'amélioration sont possibles.

Liana M. Lorigo et Venu Govindaraju [106] présentent un survey intéressant sur la reconnaissance de l'écriture arabe en passant en revue la plupart des problèmes spécifiques à la reconnaissance de l'écriture arabe.

En conclusion, les auteurs expliquent que des systèmes de reconnaissance de l'écriture arabe donnent des performances satisfaisantes sur des applications contraintes (petite taille de lexique, forme des mots relativement contrainte...). L'avenir de la discipline se situe dans la capacité des systèmes à traiter de l'écriture libre, comme des courriers manuscrits. De telles applications nécessitent des modèles de langage développés, qui sont pour l'instant des voies largement inexplorées en langue arabe. L'intégration de contraintes morphologiques (analyse de la formation des mots à partir d'affixes, suffixes, racines) permettrait également de reconnaître des mots hors-vocabulaire.

A. Belaïd et Ch. Choisy [107] s'appuient sur le modèle de McClelland et Rumelhart pour structurer leur analyse des systèmes de reconnaissance de l'écriture arabe manuscrite en insistant sur le fait que ce modèle est également applicable à la reconnaissance de l'écriture manuscrite arabe, à condition de lui rajouter un niveau intermédiaire : le niveau pseudo-mot.

Les auteurs soulignent le fait que les primitives de bas niveau sont indépendantes du langage alors que les primitives de plus haut niveau dépendent de la langue considérée, et nécessitent le développement de procédures spécifiques à la langue. Ils mettent aussi en évidence la difficulté de la segmentation en mots [108], qui est selon eux l'une des raisons qui expliquent l'absence de système commercial de reconnaissance de l'écriture arabe manuscrite.

Les auteurs insistent également sur le fait que la segmentation en lettres est un problème mal posé. Or la segmentation d'un mot manuscrit arabe est un problème plus complexe que la segmentation d'un mot latin [109]. Les auteurs déplorent donc la quantité de travaux qui s'attachent à optimiser la reconnaissance de lettres extraites à partir d'une segmentation manuelle, compte tenu qu'il semble illusoire qu'une segmentation automatique puisse fournir une segmentation en lettres idéale en entrée d'un reconnaisseur de caractères isolés.

Finalement, les auteurs montrent un vif intérêt pour les systèmes hybrides qui, selon eux, semblent très prometteurs : ils combinent efficacement différents niveaux perceptifs, permettant ainsi de discriminer des mots sans avoir accès à une description complète. L'ajout d'informations locales à un système global permet d'étendre le vocabulaire en limitant les

confusions. Les approches hybrides ne nécessitent pas une segmentation complète, et sont moins sujettes aux perturbations induites par les problèmes de perte d'information.

Mohamed Cheriet [110] fait une synthèse des différentes approches, avec l'intention d'ouvrir une réflexion sur de futures applications industrielles. il part du constat suivant : les systèmes de reconnaissance de l'écriture manuscrite arabe, même s'ils obtiennent des performances encourageantes, n'ont pour l'instant fait leurs preuves que sur des données académiques, dans le cadre d'applications à l'environnement contraint. Il n'y a pas de système commercial de reconnaissance de l'écriture arabe manuscrite.

Il pose un certain nombre de questions ouvertes en insistant également sur l'importance de l'analyse morphologique. Il s'interroge sur le meilleur endroit pour intégrer l'analyseur morphologique : au sein de la chaîne, ou à la fin en tant que post-traitement ?

Il s'interroge également sur les moyens de constituer un lexique pour guider un système de reconnaissance à vocabulaire ouvert, et conclut par l'intérêt d'utiliser le Coran en tant que corpus, pour en dériver un lexique de manière automatique ou semiautomatique.

Nous présenterons par la suite un état de l'art de la reconnaissance de l'écriture arabe selon les différentes fonctionnalités à savoir : les prétraitements, la segmentation, la reconnaissance, et les post-traitements.

3.3 Prétraitements

La plupart des pre-traitement sont les mêmes appliqués à l'écriture latine, d'où nous nous limitons à présenter les différentes techniques de détection de ligne de base et de correction de l'inclinaison des textes Arabes. L'analyse et la reconnaissance de documents a pour but de convertir un document sous format papier vers un format électronique compréhensible et réutilisable. Le document papier, une fois converti sous forme électronique, permet une recherche par le contenu, un transfert très rapide, un archivage et une gestion beaucoup plus aisée. L'un des problèmes le plus fréquent dans un système d'analyse et de reconnaissance de documents est la détermination de l'inclinaison du document. En général, l'inclinaison des images est provoquée essentiellement soit par un mauvais positionnement des pages lors de la saisie optique, soit par une mise en page fantaisiste et irrégulière de l'auteur.

3.3.1 Correction de l'inclinaison et ligne de base

Le choix de caractéristiques est très décisif pour l'étape de reconnaissance de l'écriture. Certaines caractéristiques dépendent de la qualité de la ligne de base, d'où l'importance de cette dernière. Dans la littérature, plusieurs méthodes sont proposées pour l'extraction de la

ligne de base. Certains chercheurs extraient les lignes de base après avoir corriger l'angle de l'inclinaison du mot. D'autres, les détectent sans faire la correction de l'inclinaison [111].
Dans cette section, nous présentons quelque méthodes d'extraction de la ligne de base de textes arabes manuscrits.
La transformée de Hough a été testée sur des documents arabes imprimés. Mais, le problème majeur était la lenteur des calculs. Ainsi et afin de palier à ce problème, la réduction de points à traiter est necessaire, et ce sans altérer la précision dans la détection de l'angle d'inclinaison recherché pour la totalité des points de l'image. La transformée de Hough est une technique de détection des lignes et des courbes dans une image. Elle est utilisée aussi pour détecter l'angle d'inclinaison avec un intervalle de détection compris entre 0° et 180°. Cette méthode est exacte, robuste et appropriée pour des documents multi-colonnes, mais elle nécessite un espace mémoire important et un temps de traitement prohibitif.
Dans [112], Sehad et al, la transformée de Hough est utilisée pour détecter l'angle d'inclinaison avec un intervalle de détection compris entre 0° et 180°. Cette méthode est exacte, robuste et appropriée pour des documents multi-colonnes, mais elle nécessite un espace mémoire important et un temps de traitement prohibitif.
L'idée de base est de chercher à limiter la quantité d'informations sur laquelle on effectue la transformation de Hough. Dans un premier temps, les auteurs déterminent les boîtes englobantes des liaisons détectées. Ensuite, ils appliquent une transformée de Hough sur leurs centres de gravité. Cette méthode donne des résultats intéressants : 96% pour un seuil de tolérance de 1° pour sur des documents arabes imprimés.
Dans [113], Peter Burrow propose une méthode basée sur l'ACP pour déterminer l'axe principal de la distribution des pixels de l'image. Le vecteur principal de l'ACP correspond à la direction de la ligne de base.
Cette technique donne un angle, mais pas la position de la ligne der base. Pour retrouver la position de la ligne de base, il faut effectuer une rotation de l'angle correspondant et déterminer la position du pic de l'histogramme horizontal. Néanmoins, cette méthode est sensible aux mots courts qui possèdent beaucoup d'ascendants et de descendants. La présence de nombreux signes diacritiques peut également perturber l'évaluation de la ligne de base.
Dans [114], Pechwitz et Märgner présentent une méthode basée sur le squelette pour extraire la ligne de base. Des primitives sont extraites à partir des squelettes pour une image donnée pour déterminer une première estimation de la ligne de base. Cette estimation permet de définir une bande dont la hauteur correspond à 1/3 de la hauteur du mot, dans laquelle on va

chercher des points supports plus précis. Des bons candidats sont les minimums des boucles qui sont dans cette bande, et les points supérieurs des longs tracés courbes qui correspondent aux descendants. Une ligne de base est obtenue par régression linéaire de tous ces candidats.
L'algorithme proposé extrait des lignes de base dont l'erreur est inférieure à 15 pixels dans 95% des cas dans un test sur la base IFN/ENIT.
Farooq et al. [115] proposent une méthode basée sur la détection des minima locaux du contour supérieur des mots. Afin de trouver la position de la ligne de base, une régression linéaire est appliquée sur les minima locaux.
Dans [116], Masmoudi et Amiri localisent la bande de base à l'aide d'un seuil fixé sur l'histogramme de projection horizontale.
– Limite haute : 18% de la valeur max de l'histogramme horizontal, au dessus du pic de l'histogramme.
– Limite basse : 30% de la valeur max de l'histogramme horizontal, en dessous du pic de l'histogramme.
Cette méthode est sensible à la présence de signes diacritiques et aux successions de descendants qui peuvent affecter l'histogramme et donc perturber l'extraction de la bande de base.

3.4 Segmentation

L'objectif d'une technique de segmentation repose sur sa stratégie de décision qui définit la meilleure option de coupure et permet d'isoler le mieux possible un caractère afin qu'il soit reconnu comme tel par le module de reconnaissance.
La segmentation d'un texte Arabe se fait à l'aide de trois étapes à savoir : segmentation en lignes, segmentation en mot et pseudo-mots, et segmentation en caractères/graphèmes.

3.4.1 Segmentation en lignes

La segmentation d'un texte manuscrit en lignes d'écriture est une étape nécessaire dans le développement d'un système de reconnaissance automatique de l'écriture. Cette opération est rendue délicate, dans le cas de l'écriture manuscrite, par la présence des espacements irréguliers entre lignes et des fluctuations de la ligne directrice de l'écriture par rapport à l'horizontale. Pour l'écriture arabe, la présence massive des points diacritiques complique en plus cette tâche. Les techniques utilisées pour la segmentation en lignes de l'écriture arabe sont basées sur : une projection horizontale de limage du texte [117,118] et suivi du contour partiel [119, 120].

3.4.2 Segmentation en mots et en pseudo-mots

Un mot peut être formé de plusieurs parties connexes car même si l'écriture arabe est cursive il existe des lettres qui ne s'attachent jamais à la lettre suivante [121]. Nous appelons ces lettres, des caractères de fin de tracé et les analysons séparément [122], voir figure 3.4.

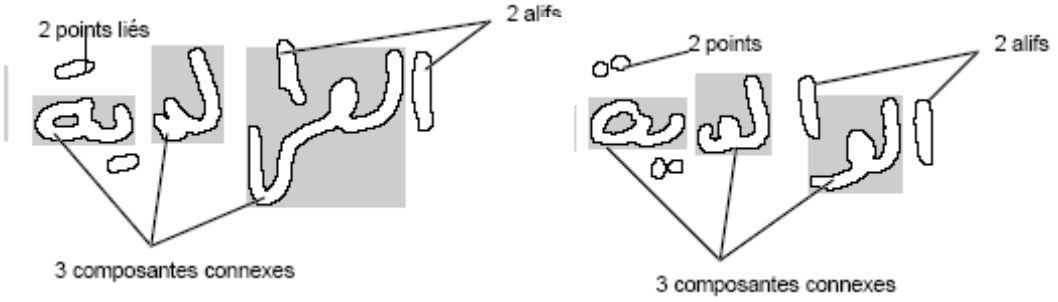


Fig 3. 4 – Deux mots appartenant à la même classe [122].

S. Srihari et al [123] présentent une technique de segmentation en utilisant un réseau de neurones pour déterminer les points de coupures d'une segmentation en mots. Les composantes connexes qui correspondent aux pseudo-mots sont extraites, et les espaces entre deux pseudo-mots successifs sont considérés comme des candidats de séparation inter-mots. Un vecteur de 9 primitives de deux pseudo-mots successifs est ainsi extrait pour qualifier chaque espace. Un réseau de neurones est entraîné pour classer ces candidats en deux classes : séparateur de mots ou non. Les auteurs remarquent que la présence de alif donne une information importante pour guider une segmentation en mots. Sur l'ensemble des 10 scripteurs de la base, le système segmente correctement 60% des mots. Ces performances semblent relativement faibles.

Miled et al [124, 125] décrivent un certain nombre de problèmes liés à la segmentation en pseudo mots :

– sous segmentations en pseudo mots (voir figure 3. 5) :

– succession de caractères avec jambes qui peuvent se toucher. Cette situation fait apparaître un minimum local du contour supérieur en dessous de la bande de base.

– surcharge de pixels noirs dans la zone médiane : lorsque l'un des caractères qui ne devraient pas être attaché à son successeur touche le caractère suivant. Aucune réponse n'est apportée à ce problème.

– sur segmentations en pseudo mots (voir figure 3.6) :

– problèmes de numérisation, une composante connexe est découpée en deux.

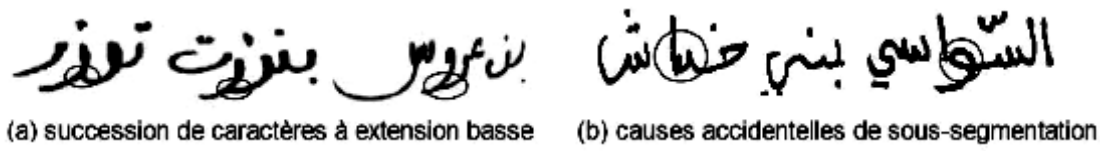


Fig 3. 5 – Exemples de sous-segmentations en pseudo-mots (figure extraite de [125]).

Fig 3. 6 – Exemples de sur-segmentations en pseudo-mots (figure extraite de [125]).

– levée de plume : la lettre qui suit un levé de plume n'est pas liée à la lettre précédente et fait apparaître dans le tracé une coupure indésirable. Ce phénomène crée donc une composante connexe supplémentaire.

3.4.3 Segmentation en caractères et en graphèmes

Sari et al [109] proposent un système de segmentation en lettres du texte arabe manuscrit. Les points de segmentation potentiels se situent au niveau des minimums locaux de contours extérieurs bas. Ces points de segmentation doivent respecter certaines règles pour les retenir comme points de segmentation décisifs. La segmentation est finalement validée en utilisant un reconnaisseur de caractères.

Menasri [126,127] propose un système de reconnaissance de l'écriture arabe évalué sur la base IFN/ENIT. La segmentation en graphèmes est la même que celle utilisée dans la reconnaissance de l'écriture cursive latine [128]. Un exemple de segmentation en graphèmes est illustré, figure 3.7.

Fig 3. 7 – Exemple de segmentation en graphèmes (figure extraite de [127]).

L'auteur propose un alphabet de corps de lettres de 34 symboles (voir figure 3.8).

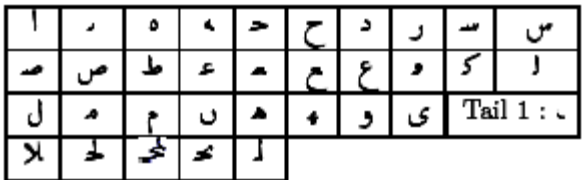


Fig 3. 8 – Liste des 34 symboles qui constituent l'alphabet.

Dans [129], Motawa et al exploitent des techniques de morphologie mathématique pour segmenter des mots manuscrits en graphèmes. Leur algorithme est basé sur le principe des régularités/singularités.

Dans [130], C. Olivier et al proposent une technique basée sur la recherche des minimums locaux des contours supérieurs des mots. Les points de segmentation sont les minimums qui satisfont les règles suivantes:

– Si un point candidat est au-dessus d'une boucle, il est éliminé.

– L'épaisseur du tracé à l'endroit du point candidat doit être inférieure à un seuil

– Si plusieurs candidats sont voisins, on choisit le plus proche de la ligne de base.

3.5 Reconnaissance de mots

Il y a deux façons d'aborder cette problématique. Soit le système reconnaît le mot comme une entité entière et indivisible, il s'agit d'une approche globale ou holistique. Soit il reconnaît le mot à partir de ses caractères préalablement segmentés, il s'agit d'une approche analytique.

Dans cette section, nous nous limitons aux approches de type markovien et nous présentons un état sur l'emploi des HMMs de différents types en reconnaissance de l'écriture arabe.

Nous montrons la contribution de ces modèles dans la résolution des problèmes liés au l'écriture arabe tels que la présence de ligatures et de signes diacritiques et les recouvrements partiels des caractères tout en évoquant les limitations enregistrées.

Des applications relativement limitées basées sur les HMMs, ont été développées pour la reconnaissance de l'arabe, aussi bien imprimé que manuscrit (en-ligne et hors-ligne). Par exemple, Amin et Mari ont utilisé les HMMs pour la correction lexicale de textes arabes multifonte [131]. Les travaux sont présentés selon le mode de reconnaissance utilisé, en-ligne ou hors-ligne en précisant pour chacun l'approche de reconnaissance globale ou analytique.

3.5.1 Reconnaissance en ligne l'écriture manuscrite

Relativement peu de travaux ont été menés en reconnaissance automatique de l'écriture arabe manuscrite en-ligne.

Mahjoub *et al.* développent des modèles de durée pour la reconnaissance en-ligne de caractères isolés [132, 133, 134]. Les auteurs se sont investis dans la modélisation de la durée d'état dans une chaîne de Markov dans ses deux contextes stationnaire et non stationnaire. Un modèle de type ergodique à durée d'états explicite (DHMM) a été appliqué à la reconnaissance en-ligne des caractères arabes. Chaque caractère est représenté par une séquence de distances radiales normalisées (figure 3.9). Une matrice de distribution de la durée dans chaque état est définie. Bien qu'elle soit caractérisée par une matrice de transitions stationnaires, la modélisation de la durée d'état par DHMM, a permis d'améliorer la discrimination de 0 à 12 % entre les modèles des différents caractères par comparaison aux HMMs classiques.

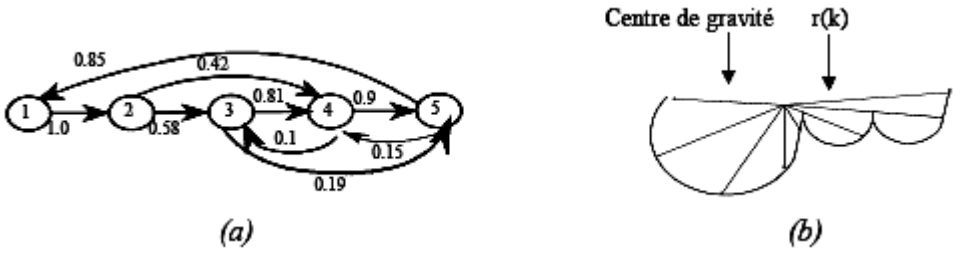


Fig 3. 9 – (a) HMM associé au caractère "س" et (b) Séquence radiale, d'après [134].

Dans le but de mieux exprimer la nature dynamique du script arabe et mieux gérer les transitions entre les états, l'exploration des HMMs non stationnaires (NSHMM) a été proposée. Le NSHMM est caractérisé par un ensemble de paramètres de probabilité de transitions dynamiques. Les résultats obtenus par le NSHMM sont meilleurs que ceux du DHMM, cependant le modèle est pénalisé par le nombre volumineux d'opérations à effectuer particulièrement au niveau de l'apprentissage. Les mêmes modèles ont été testés sur un ensemble de 2850 pseudo-mots manuscrits dans un contexte multiscripteurs. Les scores s'échelonnent entre 89% et 92% en mode monoscripteur et entre 87% et 92% en mode multiscripteurs.

3.5.2 Reconnaissance hors-ligne de l'écriture manuscrite

La segmentation en caractères (ou en parties de caractères) ou encore en pseudo-mots, constitue le problème le plus ardu lié à la reconnaissance de l'écriture arabe. Les difficultés

rencontrés à ce niveau sont de même type que celles affrontées lors de la reconnaissance du latin manuscrit, mais souvent plus complexes à cause de la diversité des formes du caractère arabe, de la courte liaison qui existe entre deux caractères successifs, de l'allongement des ligatures horizontales et de la présence de ligatures verticales. Tous ces problèmes se trouvent accentués dans le cas du manuscrit à cause de la variabilité du script.

- ***Mots à l'aide de HMM 1-D Graphèmes***

Des HMMs 1D ont été proposés par Fehri *et al.* pour la reconnaissance manuscrite de noms de villes tunisiennes [135, 136]. La segmentation en caractères (ou en segments) est opérée sur le squelette de la chaîne considérée.

Chaque caractère est modélisé par un HMM 1D à trois états représentant les différentes situations de segmentation. Le modèle d'un mot, représenté par la figure 3.10, est une concaténation des modèles des caractères qui le composent. Ce modèle est à même de représenter des mots de longueur différente. Les transitions entre les modèles de caractères sont obtenues statistiquement sur un dictionnaire de mots arabes. La reconnaissance est basée sur une méthode hybride connexionniste/markovienne dans laquelle le modèle neuronal assure une classification préliminaire des segments issus de la phase de segmentation. La méthode a été testée sur un ensemble de 50 noms de villes tunisiennes en mode monoscripteur, les premiers résultats ont été jugés encourageants par les auteurs.

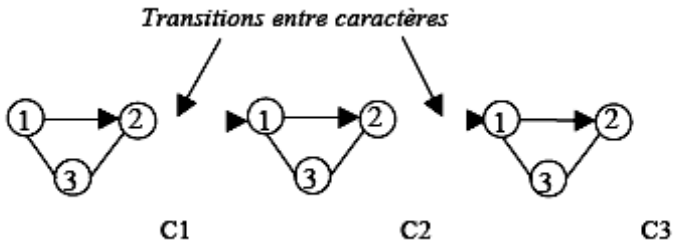


Fig 3. 10 – Modèle de mot d'après [135].

Dans les travaux de Miled *et al.,* deux approches markoviennes ont été proposées pour la reconnaissance hors-ligne omni-scripteurs de l'écriture arabe, dans un vocabulaire relativement étendu (232 classes de mots différents correspondants à des noms de villes tunisiennes) [137, 138, 139].

Dans l'approche globale, des modèles classiques de type droite-gauche ont été élaborés pour la modélisation d'un mot donné. Etant donné un mot, des indices visuels extraits des diacritiques et du contour du tracé du mot sont transformés en une séquence d'observations qui sont gérées par des HMMs 1D. Les modèles ont été entraînés sur une base de 4720 mots

et les tests ont été effectués sur une autre base de 5900 mots. Des résultats modestes de 60% (top1) ont été enregistrés.

Devant ces scores, relativement faibles, une approche analytique a été envisagée. Les HMMs sont prévus dans ce cas afin de superviser les erreurs de segmentation. Le caractère est segmenté au maximum en trois graphèmes et il existe au moins un point de segmentation après chaque paire de caractères (figure 3.11). Ainsi, le modèle d'un caractère est un modèle droite-gauche à trois états. La figure 3.12-a montre comment le HMM simule les différentes possibilités de segmentation d'un caractère en graphèmes. Le modèle d'un pseudo-mot n'est autre qu'une concaténation des modèles des caractères qui le constituent. Le modèle final du mot est alors une fusion des deux modèles précédents (figure 3.12-b). Pour un mot donné, le nombre d'état est égal à la somme du nombre de graphèmes issus de la segmentation et du nombre de transitions entre pseudo-mots. Un état fictif désigné par le symbole "#", est prévu pour représenter l'observation liée à l'espace entre pseudo-mots d'un même mot. Il permet également de modéliser les variations en terme de séquences de pseudo-mots dans le mot.

Des scores de 81% en 1er choix et 90% en 5ème choix ont été obtenus sur un vocabulaire de 5900 mots et pour une base d'apprentissage de 4720 mots.

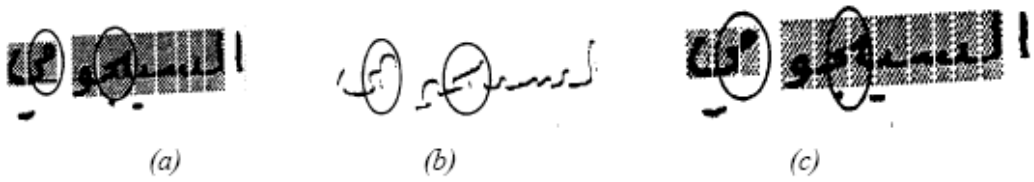
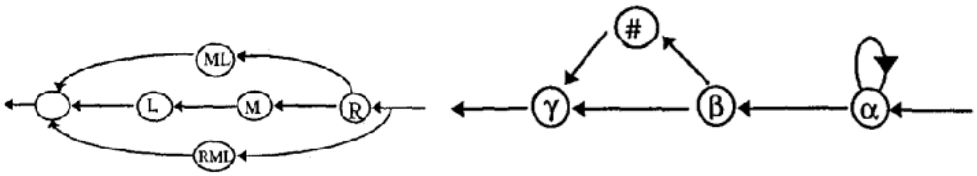


Fig 3. 11 – Segmentation en graphèmes : (a) mauvaise segmentation, (b) détection du chevauchement, (c) correction, d'après [139].



G : gauche, M : milieu, D : droite,
DM : droite-milieu,
DMG : droite-milieu-gauche

(a)

α, β, et γ: représentent les caractères
et # : l'espace entre pseudo-mots.

(b)

Fig 3. 12 – Modèle de caractère (a) et de mot (b), d'après [137].

Par ailleurs, les auteurs ont intégré les deux approches précédentes dans un système à trois niveaux de perception relatifs respectivement aux caractères, aux pseudo-mots et aux mots

[139]. Le troisième niveau propose une modélisation markoviennne de mot pseudo-analytique, basée sur la concaténation des modèles des pseudo-mots associés (figure 3.13). Par rapport au modèle proposé dans [138] où les observations sont associées à l'entité caractère, les observations dans [139] sont associées à l'entité logique pseudo-mot. Il a été constaté que l'information contenue dans le pseudo-mot a amélioré de 5 % les performances du classifieur par rapport au classifieur global classique.

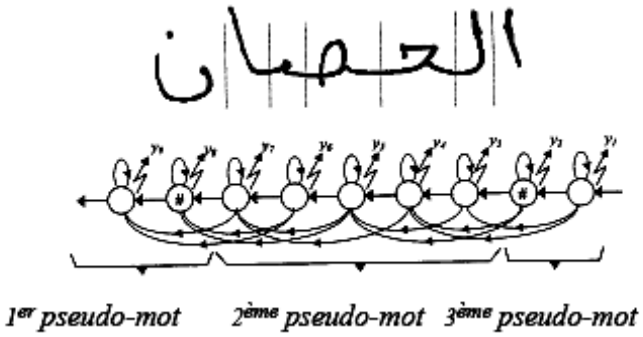


Fig 3. 13 – Modèle final associé au mot "الحصان", d'après [139].

Dans [140], Khorsheed propose une approche à base de Modèles de Markov Cachés.
Il utilise la méthode d'extraction du squelette proposée par T. Y. Zhang et C. Y. Suen [141]. Les boucles sont extraites à partir du squelette et les autres tracés sont approximés par des petits segments droits et traduits en symboles. Deux primitives sont extraites sur chacun de ces symboles : sa longueur et son angle.
Les lettres sont modélisées par des HMMs et les modèles de mots sont obtenus par concaténation de modèles de lettres qui les constituent. L'apprentissage est réalisé à l'aide de la méthode de Baum-Welch.

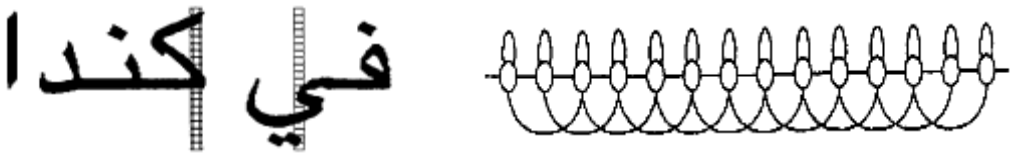
- ***HMMs par la technique de la Fenêtre glissante***

Plusieurs travaux utilisent une segmentation implicite par fenêtre glissante et s'appuient sur un reconnaisseur à base de HMM.
Les systèmes de reconnaissance qui obtiennent les meilleurs taux de reconnaissance sur le manuscrit arabe sont basés sur cette architecture. El Hajj et al [142, 143] proposent un système de reconnaissance à base de HMM à fenêtre glissante. Les primitives sont extraites dans une bande verticale de 8 pixels de large. Cette bande verticale est découpée en cellules homogènes. Le détail des primitives utilisées est donné dans [143].
Dans [144, 145], A. Benouareth et al. mettent en concurrence deux stratégies de segmentation en bandes verticales (voir figure 3.14). La première est uniforme (toutes les bandes ont la

même largeur) : les auteurs fixent empiriquement la largeur d'une bande à 10 pixels. La deuxième stratégie de segmentation est non uniforme : le système découpe l'image en bandes verticales dont la largeur varie. Ce découpage s'appuie sur une analyse de l'histogramme de projection vertical. Mais les auteurs ne donnent pas précisément la procédure qu'ils utilisent pour segmenter l'image en bandes à partir de l'histogramme. Benaroueth et al. comparent les résultats obtenus en modélisant explicitement la durée d'état dans les HMMs avec plusieurs lois de distribution. Par ces expériences, ils montrent d'une part que la segmentation non-uniforme donne de meilleurs résultats que la segmentation en bandes uniformes. Et d'autre part, ils montrent également que l'utilisation de DHMM (HMM à durée d'état explicite) se révèle plus performante que l'utilisation de HMM semi-continus standards.

Fig 3. 14 – Segmentation en bandes uniformes et non-uniformes (figure extraite de [144]).

Schwartz et al. [146] ont présenté un système de reconnaissance de textes imprimés arabes. Une ligne de texte (supposée être horizontale) est parcourue de la droite vers la gauche, selon le principe de la fenêtre glissante, est fractionnée en bandes verticales de largeur égale environ à 1/15 de la hauteur de la ligne (figure 3.15-a). Chaque bande est alors divisée en 20 cellules égales; sur chacune d'entre elles, 80 caractéristiques sont extraites et transformées en une séquence d'observations qui sont gérées par des HMMs 1D. Le caractère est alors modélisé par un HMM de type gauche-droite à 14 états (figure 3.15-b). Le modèle d'un mot est obtenu par concaténation des modèles de caractères qui le constituent.



(a) (b)

Fig 3. 15 – (a) Extraction des primitives d'une ligne de texte, (b) HMM à 14 états associé aux caractères, d'après [146].

- ***Mots à l'aide de HMM planaires***

Les HMM planaires sont une extension des HMM pour résoudre des problèmes à deux dimensions. En pratique, les probabilités d'émission du HMM principal sont remplacées par la vraisemblance d'un deuxième HMM. En général, le HMM principal décrit l'image verticalement ligne par ligne, tandis que les HMM secondaires décrivent une ligne site par site.

Appliqués à l'arabe, les HMM planaires ont été utilisés en reconnaissance de l'écriture imprimée, et en reconnaissance de l'écriture manuscrite.

Dans les deux cas, l'image d'un mot est décomposée verticalement en plusieurs zones selon la topologie de la forme considérée (figure 3.16). Les zones correspondent aux bandes de variations de l'écriture arabe, mettant ainsi en évidence des caractéristiques morphologiques donnant le contexte de type géographique des différents caractères qui constituent le pseudo-mot, à savoir : hampes, points diacritiques supérieurs, boucles et/ou lieu de ligature verticale, points diacritiques inférieurs et/ou lieu de ligature verticale et jambage. Chaque zone est modélisée par un modèle secondaire horizontal (HMM 1D) de type gauche-droite, dont les paramètres sont étroitement liés à sa topologie. 1 HMM-1D vertical de 'super-états' gère les transitions verticales.

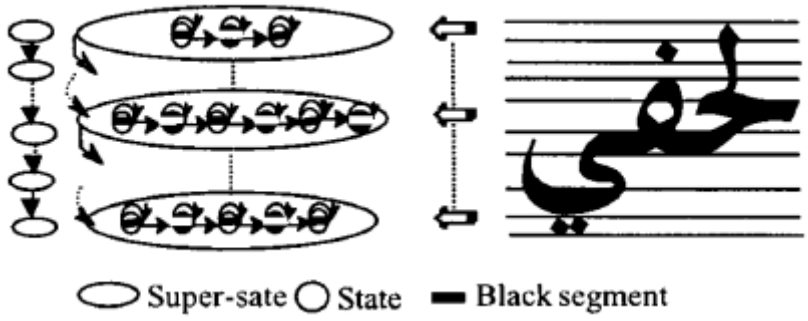


Fig 3. 16 – PHMM : Découpage en bandes. Un HMM-1D par bande horizontale, et un HMM vertical de 'super-états' (figure extraite de [147]).

Toutefois, les résultats de la compétition ICDAR 2005 sur la reconnaissance de l'écriture arabe manuscrite [148] montrent que sur des vocabulaires plus importants, les systèmes à base de PHMM appliqués à l'écriture arabe ne parviennent pour l'instant pas à atteindre des performances satisfaisantes.

Chapitre 4 - Reconnaissance des montants littéraux arabes

Dans le domaine de la reconnaissance omniscripteur des mots manuscrits cursifs (hors ligne), on distingue deux principales approches:

La première approche concerne les méthodes qui consistent, à partir d'une hypothèse sur le mot obtenue par une reconnaissance globale ou contextuelle, à vérifier la segmentation et la reconnaissance des lettres (méthodes descendantes) ;

La seconde approche concerne les méthodes analytiques qui se basent sur des hypothèses de lettres pour aboutir à la reconnaissance du mot (méthodes ascendantes).

La principale critique que l'on peut formuler à l'égard de la première approche est que la génération d'hypothèses sur le mot n'exploite pas suffisamment la notion de lettre (l'extraction de primitives est globale dans le mot), c'est une attitude opposée à celle que l'on peut adopter dans le cas de notre système où la seule reconnaissance des lettres/graphèmes suffit pour aboutir à la reconnaissance complète du mot.

Nous présentons une nouvelle méthode de reconnaissance de mots manuscrits fondée sur un modèle hybride de type neuro-markovien qui intègre des réseaux neuronaux (RN) et des modèles de Markov cachés (MMC) dans une architecture complémentaire.

4.1 Architecture et description du système de reconnaissance utilisé

Le schéma de notre système de reconnaissance est décrit en détail sur la figure 4.1. Ce système a fait l'objet d'une publication internationale [149].

A travers les différentes étapes effectuées dans notre système, on peut constater que la structure générale d'un système de reconnaissance hors-ligne de mots manuscrits a été conservée à savoir : Prétraitement, segmentation, extraction de caractéristiques et classification/reconnaissance.

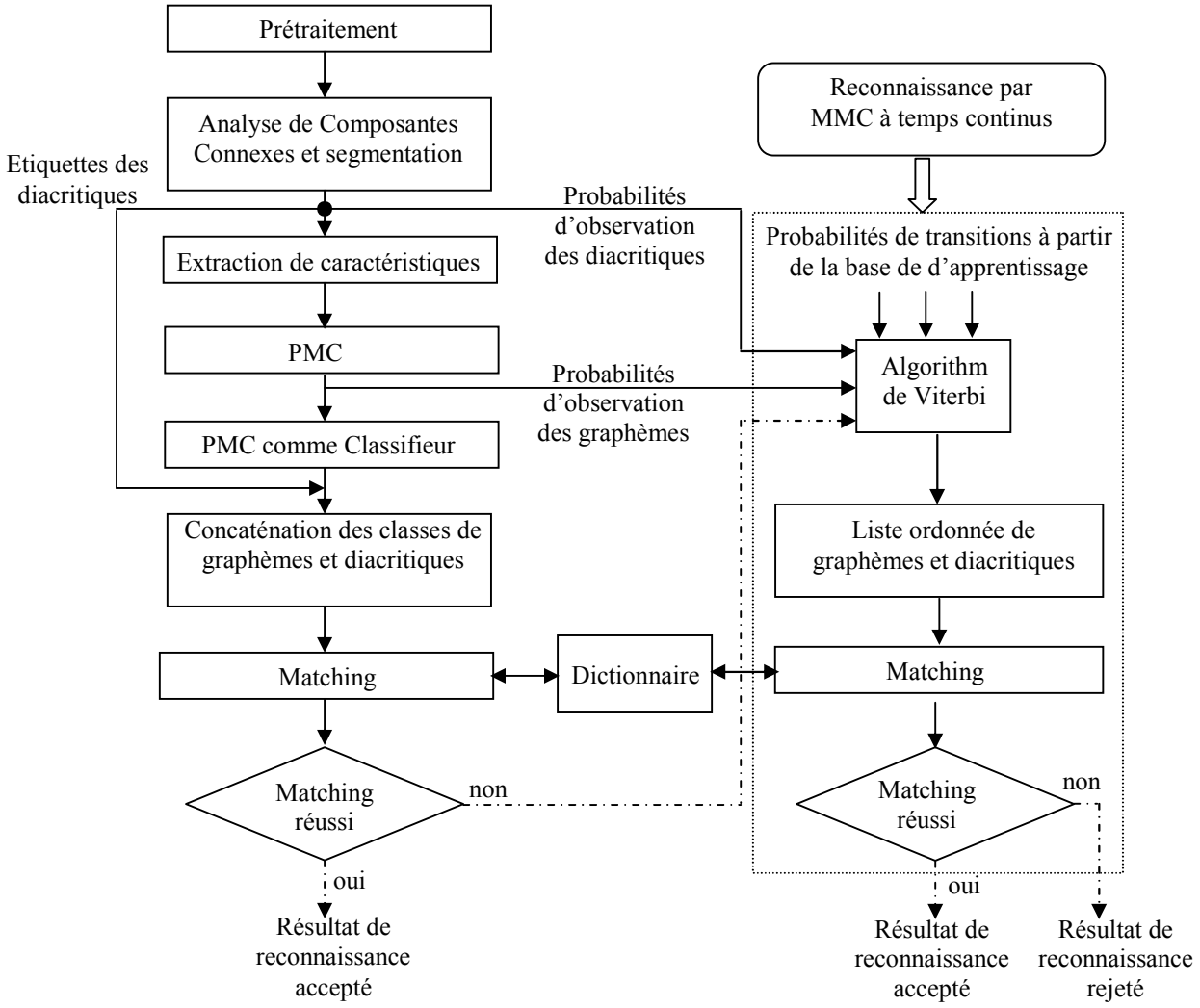


Fig 4. 1 – Schéma block du système de reconnaissance proposé: les flèches en tiret et continues représentent respectivement les flèches de contrôle et de données.

4.2 Segmentation des mots

Deux axes se dégagent des techniques rencontrées dans la littérature abordant le problème de la segmentation : le premier concerne les techniques de segmentation de type implicite où les options de coupures dépendent directement de la reconnaissance. La seconde voie explore, quant à elle, des techniques où la décision de la segmentation précède la reconnaissance. Dans ce cas, la segmentation est dite explicite du fait de son action antérieure à la reconnaissance. Notre application est développée autour de ce deuxième principe. La segmentation est fondée sur des propriétés contextuelles propres à l'écriture Arabe.

Nous décomposons la segmentation de l'image en deux étapes : d'abord la segmentation d'une ligne en composantes connexes où chaque composante connexe est classée en deux

catégories : signes diacritiques et pseudo-mots contenant les caractères isolés; et ensuite la segmentation des pseudo-mots en graphèmes, beaucoup plus délicate que les deux autres.

4.2.1 Segmentation en composantes connexes

Les caractères Arabes présentent en général un problème de chevauchement. Deux caractères se chevauchent lorsque l'on ne peut pas encadrer un caractère dans une fenêtre correspondant à sa dimension sans croiser le caractère voisin. Deux pseudo-mots adjacents peuvent alors se chevaucher, rendant la séparation par projection verticale délicate. Pour l'écriture Arabe, la présence massive des points diacritiques complique en plus cette tâche. Pour résoudre ce problème de chevauchement nous utilisons la méthode de détection et étiquetage des composantes connexes.

Dans notre application, les composantes connexes sont extraites puis classifiées suivant la taille et la position en deux classes : les pseudo-mots incluant les caractères isolés et les signes diacritiques incluant les points. La figure 4.2 montre les différents types de composants connexes formant le mots quatre-vingts dix '' **تسعون**''

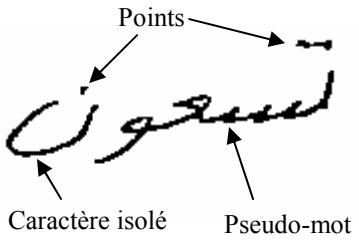


Fig 4. 2 – Différents types de composants connexes formants le mots quatre-vingts dix '' **تسعون**''.

- ***Détection des signes diacritiques***

La détection des signes diacritiques a deux utilités. D'une part, un trop grand nombre de signes diacritiques risque de perturber la segmentation. Et d'autre part, les signes diacritiques ainsi détectés vont par la suite réduire l'alphabet du lexique.

La détection des signes diacritiques se fait en deux étapes.

La première étape consiste à effectuer un filtrage des composantes connexes en s'appuyant sur des critères assez simples : taille des boîtes englobantes et superposition verticale des composantes connexes.

L'objectif est de filtrer les signes diacritiques tout en gardant les corps de lettres et les pseudo-mots. Les seuils de filtrage sont fixés empiriquement.

4.2.2 Segmentation en graphèmes

En arabe, les descendants peuvent avoir des portions horizontales suffisamment longues qui vont perturber la segmentation en graphèmes, la figure 4.2 illustre ce phénomène. La détection des CCs met en évidence les pseudo-mots et les lettres séparés. Ces coupures naturelles sont considérées comme des points de segmentation. Pour un groupe de lettres liées (pseudo-mot), la séparation se fait de façon explicite : la nouvelle méthode de segmentation est basée sur le calcule de l'histogramme de transitions (du blanc vers le noir) selon la direction verticale de chaque pseudo-mot. Pour éviter que l'histogramme de transitions ne soit perturbé par la présence de signes diacritiques, on effectue au préalable un filtrage de ces composantes car la présence des points diacritiques peut donner lieu à de faux points de segmentation. Le mot, préalablement dépourvu de ses points diacritiques, est divisé en graphèmes qui peuvent correspondre à des parties de caractères, des caractères ou encore des traits de liaison (ligatures horizontales) inter-caractères.

La segmentation du mot en graphèmes est principalement réalisée à l'aide d'une fonction d'évaluation basée sur le nombre de transitions blanc/noir de chaque colonne. On effectue un balayage de haut en bas et de droite à gauche, car c'est le sens de l'écriture Arabe, et on localise chaque transition blanc/noir au pixel noir en retenant son niveau de gris 0 et en remplaçant les autres pixels noirs par des 1s (figure 4.3-b). Le second critère de segmentation que nous avons utilisé est lié à la détection des variations brusques du contour supérieur du tracé qui possède une seule transition. La détection de cette variation locale est obtenue par le calcul de la dérivée du contour supérieur, c'est-à-dire, dans l'espace discrétisé, de la différence des ordonnées de deux points d'abscisses consécutives.

Pour résumer, les étapes de segmentation de notre algorithme sont comme suit :

Etape 1

1- le nombre de transitions passe de où à 1.

2- Le contour supérieur des parties du mot ayant une seule transition varie rapidement du haut en bas ou l'inverse.

Par exemple, les lettres (**سـ**, **شـ**) comporte une seule transition (blanc/noir) mais ont des variations brusques au niveau du contour supérieur de chaque tracé.

Cette procédure fournit les points de segmentation primaires (PSPs) qui sont représentés par des coupures verticales où les pseudo-mots sont segmentés en différentes parties, voir figure 4.3-c.

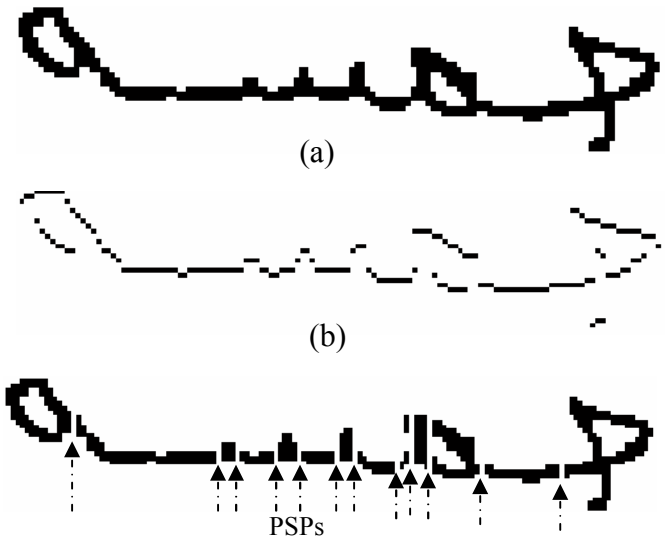


Fig 4. 3 – Les étapes de base pour extraire les (PSPs) : (a) le tracé principal du mot cinq "**خمسة** ", (b) résultat de localisation verticale de transitions blanc/noir, (c) les points de segmentation primaires (PSPs).

Etape 2

Utiliser quelques règles pour vérifier si les points de segmentation primaires sont des points de segmentations réelles (PSRs) ou non.

Règle 1 : Il arrive parfois que l'algorithme précédent divise les graphèmes de manière trop fine. Il est utile de recoller de trop petits segments aux lettres voisines afin d'éviter qu'ils ne soient considérés comme des points de coupures.

Les parties de petites largeurs sont dues au bruit (faux pixels) et doivent être non comptés/considérés. Chaque partie redondante introduit deux points de segmentation primaires qui doivent être enlevés (figure 4.4). On remarque que p1 apparaît au début/fin du caractère et, p2 et p3 apparaissent à l'intérieur du caractère. Par conséquent, p2 et p3 doivent être enlevés, et on ne garde que p1.

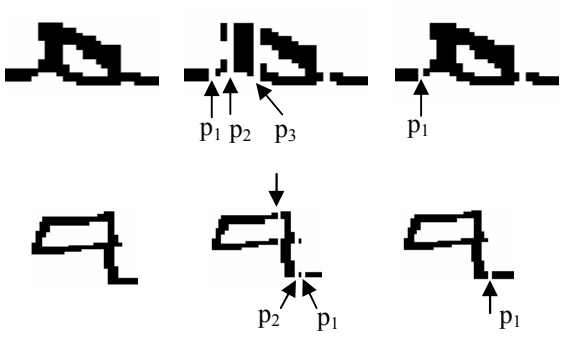


Fig 4. 4 – p_2 et p_3 sont à enlever et seulement p1 est retenu.

Règle 2 : enlever les derniers PSPs qui segmentent le dernier caractère, comme "ر", "و", et "د". Dans le cas où la dernière composante est une longe barre verticale ou une boucle, le point de segmentation est retenu. La figure 4.5 schématise en détaille cette deuxième règle.

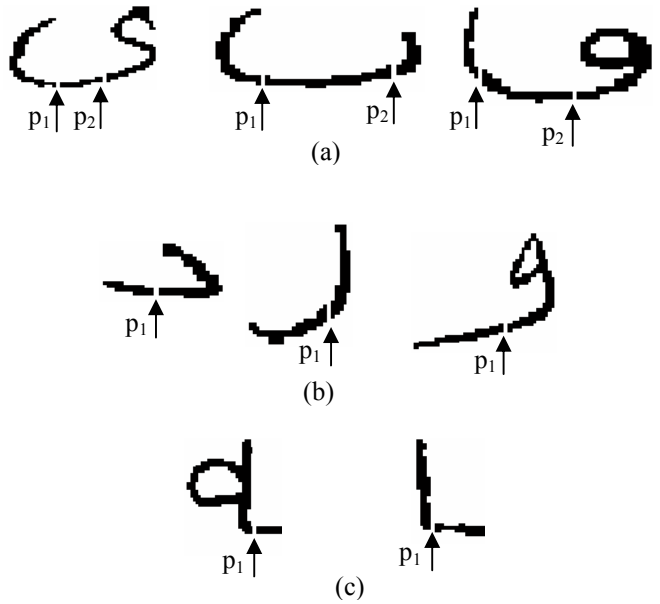


Fig 4. 5 – Filtrage des PSPs redondants: (a et b) les PSPs sont à enlever pour éviter la sur-segmentation des caractères, (c) les PSPs sont maintenus.

Ces règles sont alors appliquées à chaque PSP pour valider les points de segmentations réelles (RSPs) qui sont par la suite utilisés pour la segmentation des images de pseudo-mot en graphèmes.

Les allongements (ligatures) horizontaux ont des longueurs variables ce qui implique une variabilité intra-mots. Pour réduire cette variabilité et assurer une bonne description du mot en graphèmes, nous supprimons ces allongements redondants et ne gardons que les corps des graphèmes. La figure 4.6 montre la segmentation du mot " خمسة" de la figure 4.3-a en graphèmes ; les PSRs finaux sont dessinés par des segments blancs verticaux.

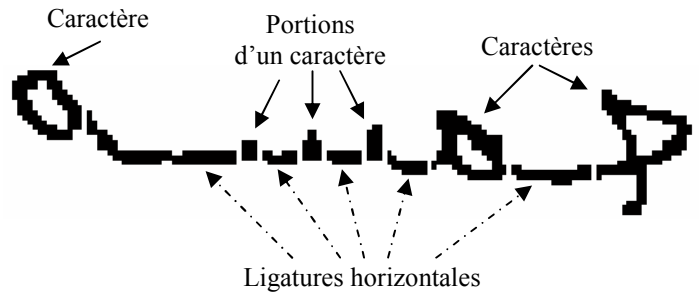


Fig 4. 6 – Segmentation du mot "خمسة" de la (Fig.4.3-a) en graphèmes : les PSRs finaux sont dessinés par des segments blancs verticaux.

Cette segmentation conduit inévitablement à la sur-segmentation des lettres présentant des variations rapides, tels que les segments verticaux dans le cas ‘سـ’ ‘sin’ ou ‘شـ’ ‘chin’.
Alors, le caractère ‘ سـ ’ est sur-segmenté en une succession de trois petits segments verticaux qui seront reconnus comme tels. Comme illustré dans la figure 4.6, chaque portion (graphème) du caractère سـ a la même classe que le corps principal du caractère “ بـ”.
Finalement, les mots sont segmentés en graphèmes qui seront reconnus individuellement.

4.3 Description en graphèmes

S’il est souhaitable de se rapprocher le plus possible d’une description du mot cursive en lettres, nous savons qu’un tel modèle suppose une segmentation parfaite du mot en lettres. Cette propriété n’était jamais atteinte en pratique, nous avons choisi d’utiliser une description en graphèmes obtenus à l’issue de la segmentation. Nous espérons cependant rester plus proche de la réalité cursive du mot et en conséquence garantir une robustesse de la description aux variabilités des styles d’écritures. Nous devons être capable de définir un alphabet de graphèmes permettant de garantir une description suffisante pour permettre la reconnaissance des mots manuscrits.
Les graphèmes sont des éléments qui conduisent, par leur enchaînement, à des hypothèses de mots et qui doivent être ordonnés selon un ordre respectant l’ordre des lettres dans le mot.
L’analyse de l’alphabet arabe révèle certaines particularités intéressantes, qui peuvent être exploitées pour améliorer les performances d’un système de reconnaissance. Menasri et al [126, 127] ont proposé un alphabet qui exploite la similarité des corps des lettres qui ne se différencient que par leurs position dans le mot et les signes diacritiques qui leurs associent. En revanche, La forme d’une lettre en fin de mot est donc la forme en début/milieu de mot, à laquelle on rajoute une ” ligature ”.

4.3.1 Notre alphabet

Nous proposons un alphabet qui exploite un certain nombre de spécificités de l’écriture arabe, en particulier la redondance des formes que peut prendre un même graphème en fonction de sa position dans le mot ; et la redondance des formes qui ne se différencient que par les signes diacritiques [126].
Pour prendre en compte la redondance des formes de graphèmes qui proviennent de la phase de segmentation, nous proposons un alphabet de corps de graphèmes, mieux adapté à l’algorithme de segmentation développé. Comme illustré dans la figure 4.3, chaque portion (graphème) du caractère سـ a la même classe que le corps principal du caractère “ بـ”. En

conséquence, le nombre des graphèmes à reconnaître est plus petit que celui des caractères composant l'alphabet des montants littérales Arabes. Par conséquent, nous avons introduit un nouvel alphabet à base de graphèmes pour la reconnaissance de montants littéraux Arabes en profitant de quelques propriétés inhérentes à l'écriture Arabe.

Cet alphabet comporte 18 classes de graphèmes, pour représenter les différents mots de l'alphabet des montants Arabes. Un pseudo-mot arabe sera traduit en un mot dans notre alphabet (une séquence de graphèmes de notre alphabet).

Dans ce cas là, étant donné une séquence de formes, il est impossible de déterminer à quelle séquence de graphèmes elle se rapporte, sans analyser les signes diacritiques. Les signes diacritiques sont alors ajoutés à notre alphabet pour décrire les mots de façons assez robuste.

4.4 Extraction de caractéristiques : description des graphèmes

Cette section présente les caractéristiques utilisées en détail. Celles-ci ont pour objectif de traduire une image d'un graphème dont les dimensions peuvent être variables en un vecteur de dimension fixe et indépendant de la taille du graphème. Ainsi, une séquence de graphèmes sera convertie en une séquence d'observations ou vecteurs de taille fixe. Les descripteurs des graphèmes sont des caractéristiques extraites à partir des contours, des profils et des transitions horizontales et verticales.

4.4.1 Description d'un graphème à partir de ses contours.

Le contour contient une information structurelle suffisamment complète pour permettre de décrire un objet dans une image. Les contours d'un caractère apportent autant d'informations que le caractère lui-même, il semble donc naturel d'en extraire des caractéristiques. Les contours sont définis comme l'ensemble des pixels du caractère ayant au moins un pixel en commun avec le fond en 4 ou 8 connexité.

Après l'extraction du contour, le contour conserve un ensemble de petits segments sans importance qui affectent le suivi du contour. Le nettoyage de ces barbules sépare le contour intérieur du contour extérieur et rend le contour en quelque sorte plus "lisible" et facile à suivre. S'il y a un pixel qui a plus de deux pixels dans ses 8-voisins, il existe alors des pixels redondants qui doivent être détectés et supprimés. Un exemple de la procédure de nettoyage de pixels redondants est illustré sur la figure 4.7 où le pixel 2 est redondant, et doit être supprimé. Une fois les pixels redondants sont supprimés, le contour devient d'un pixel de largeur et par conséquent, on trouve qu'il n'existe que quatre segments élémentaires qui sont considérés comme des segments de base qui forment le contour.

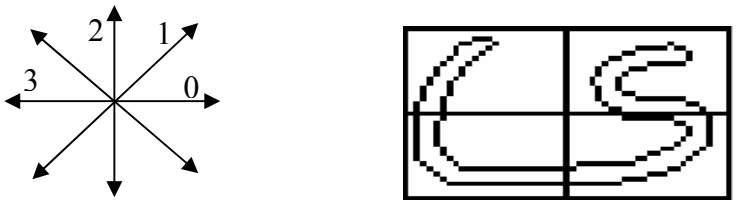
4		
	3	2
		1

Fig 4. 7 – Principe de nettoyage de pixels redondants : le pixel 2 est redondant, et doit être supprimé.

Dans ce travail, on a choisi de calculer la pente entre deux pixels successifs, qui va donner l'angle entre la ligne qui joint les deux pixels et l'axe des abscisses *x*. l'ensemble de tous les directions possibles sont alors (0°, 45°, 90°, 135°), qui sont eux-mêmes identiques aux directions (180°, 225°, 270°, 315°). Le suivi du contour est alors basé sur l'approximation angulaire la plus simple : le code de Freeman [150]. Elle consiste à coder chaque vecteur entre deux points successifs par un chiffre allant de 0 à 3, représentant les 4 directions de Freeman. Ce codage est une représentation exacte de chaque pixel du contour relativement à ses voisins. La Fig 4.8-a montre les quatre codes de directions où l'un de ces quatre codes est assigné à chaque vecteur entre deux points successifs. Le codage global du contour d'un graphème est donc la succession de tous les codes correspondants aux directions prises pour suivre le contour.

Une description par contours obtenue par un codage à partir des directions de Freeman est en général difficile à analyser à cause de la trop grande variabilité de la longueur de la chaîne codée et du point de départ (repère). C'est pour quoi plusieurs chercheurs ont utilisé l'histogramme des orientations locales des images des contours [151]. Un histogramme des orientations locales d'un contour est une fonction qui donne la fréquence d'occurrence de chaque orientation (code) et qui peut être utilisé comme vecteur de caractéristiques pour la raison suivante: le recensement des orientations locales à l'aide d'un histogramme permet d'obtenir une représentation invariante à la translation et par rapport à l'échelle.

Pour caractériser un graphème à partir de son contour, le plus petit rectangle englobant l'image du graphème est devisé en 2×2 (4) régions adjacentes de même taille (figure 4.8-b) et en calculant pour chaque région l'histogramme des orientations des contours à l'intérieur de cette région. Si le nombre de lignes/colonnes est impair, une ligne/colonne doit être ajoutée pour rendre l'image divisible en 4 régions.



(a) (b)

Fig 4. 8 – Extraction des chaînes de codes: (a) codage de Freeman à 4 directions, (b) contour du graphème "ى" devisé en 4 régions.

Le suivi du contour commence à partir d'un pixel d'extrémité du contour en cherchant le pixel voisin dans le 8 directions jusqu' à la rencontre du dernier pixel du contour. On réitère la même procédure pour tous les éléments du contour restants dans cette zone.

Un histogramme de directions de 4 classes est construit où chaque classe représente la fréquence d'occurrence du code correspondant. La concaténation des 4 histogrammes forme alors les descripteurs du contour de chaque graphème.

L'histogramme résultant est normalisé en module. Le vecteur de caractéristiques du contour est alors composé de 4×4 (16) composantes normalisé entre 0 et 1.

$$f_i = \frac{h_i}{\|h_i\|},\ i = 1,2\ldots\ldots 16 \qquad (4.1)$$

Où *hi* est la i^{ème} composante de l'histogramme du graphème, f_i est la $i^{ème}$ composante du vecteur de caractéristiques *f*, et $\|\ \|$ est la norme Euclidienne. D'où, $\|f\| = 1$. Par conséquent, ce vecteur de caractéristiques permet une meilleure résistance aux changements d'échelle.

En plus, la densité de pixels des contours calculés dans les différentes zones (zoning) de l'image du graphème est considérée. Le nombre de pixels dans chaque zone est calculé et normalisé (en le divisant par la surface totale du rectangle englobant le graphème) puisque les graphèmes ne sont pas tous de même taille.

4.4.2 Description d'un graphème à partir des transitions

Le deuxième ensemble des caractéristiques est basé sur des informations extraites à partir du nombre de transitions d'un pixel blanc vers un pixel noir dans les deux directions horizontale et verticale.

On a trois catégories de caractéristiques : la première correspond aux parties de graphèmes qui ont une seule transition, la deuxième correspond aux parties qui ont deux transitions et la troisième correspond à ceux qui ont plus de deux transitions. Les caractéristiques sont définies comme suit :

$fh_k = \frac{lh_k}{w}$, $fv_k = \frac{lv_k}{h}$ pour k=1,2 et $fh_3 = \frac{\sum_{k\geq 3} lh_k}{w}$, $fv_3 = \frac{\sum_{k\geq 3} lv_k}{h}$.

Où lh_k et lv_k sont les largeurs des parties du graphème qui ont k transitions dans les deux directions horizontale et verticale, respectivement. w et h représentent respectivement la largeur et la hauteur du graphème.

4.4.3 Description d'un graphème à partir des profils

Le troisième ensemble des caractéristiques est extrait à partir des profils externes des graphèmes qui sont très utilisés pour les caractères manuscrits.

Les quatre profils (haut, bas, droite, gauche) [152] sont obtenus par l'intermédiaire de sondes appliquées sur le graphème. Pour le profil gauche d'un graphème par exemple, on lance des sondes depuis le bord gauche de l'image qui s'arrêtent lorsqu'elles rencontrent le premier pixel noir. Les abscisses des sondes constituent le profil gauche du graphème. On obtient alors les profils haut, bas, gauche et droit du graphème en mesurant la distance séparant chaque pixel du graphème au bord du cadre.

Les profils sont caractérisés par la surface que constitué chaque profil. Les profils doivent être normalisés en les divisant par la surface de l'image du graphème. Ainsi, les petites ruptures de tracés n'influenceront pas les valeurs moyennes du profil.

A ce niveau, chaque graphème est décrit par 3 ensembles différents des caractéristiques :

La juxtaposition de ces caractéristiques donne un vecteur de dimension 32.

4.5 Modélisation statistique du système de reconnaissance

Le but de la modélisation statistique est de pouvoir intégrer au mieux toute l'information contextuelle de manière, en particulier, à déterminer la reconnaissance correcte des graphèmes et des diacritiques composant le mot à reconnaître.

4.5.1 Modélisation par un MMC

Les MMCs utilisés pour la reconnaissance des montants littéraux des chèques s'essaient à modéliser les variations possibles de l'écriture manuscrite. Après avoir segmenté le mot à reconnaître en une séquence d'observations $o_1,.....o_n$ des graphèmes et diacritiques associés, on essaie d'estimer la probabilité $p(o_1 \ldots\ldots o_n/M)$ d'apparition de cette séquence selon le processus stochastique de Markov.

La reconnaissance peut être effectuée de deux façons différentes soit dans le cas d'un modèle par classe, par recherche du modèle discriminant (Model Discriminant), soit dans le cas d'un

seul modèle pour toutes les classes, par recherche du chemin optimal qui fournira la classe [32].

Le modèle de Markov que nous utiliserons dans notre application est un seul modèle de Markov caché (MMC) pour toutes les classes (Path Discriminant) qui repose sur le principe bayesien de vraisemblance. Dans ce cas, la reconnaissance consiste à déterminer le chemin correspondant à la séquence d'observation, c'est-à-dire à trouver dans le modèle, la meilleure suite d'états, appelée suite d'états de Viterbi, qui maximise la quantité *P(E/O, Λ)*.

L'utilisation de modèles de reconnaissance suppose que l'image a été préalablement segmentée en graphèmes. De cette séquence de graphèmes, on tire une séquence de vecteurs de dimension fixe, c'est la séquence d'observations qui décrit la séquence des graphèmes.

Les scores de reconnaissance des mots sont alors des valeurs de vraisemblance.

$$V(M / o_1,.....o_n) = p(o_1,.....o_n / M)\,p(M) \qquad (4.2)$$

ou $p(o_1,.....o_n / M)$ représente la probabilité que le modèle du mot *M* engendre la séquence de segments d'image (observations) $o_1,.....o_n$. *p(M)* représente la probabilité du modèle mot M.

Dans le paradigme MMC, ce terme se décompose en une somme sur tous les chemins (séquences d'états) possibles des produits de la probabilité du chemin par la probabilité selon laquelle le chemin produit la séquence d'observation.

$$p(o_1,....o_n/M) = \sum_{e_1...e_n} p(o_1,....o_n/e_1,....e_n,M)p(e_1,....e_n/M) \qquad (4.3)$$

Dans le cas de l'écriture manuscrite pour chaque modèle une succession (un chemin) prédomine largement [153] et l'on écrit alors.

$$p(o_1,....o_n/M) = p(o_1,....o_n/e_1,....e_n,M)p(e_1,....e_n/M) \qquad (4.4)$$

Dans un MMC, chaque élément de séquence est supposé ne dépendre que de l'état correspondant.

$$p(o_1,.....o_n / e_1,.....e_n, M) = \prod_{j=1}^{n} p(o_j / e_j, M) \qquad (4.5)$$

Où $p(o_j / e_j, M)$ est la probabilité d'émettre l'observation o_j depuis l'etat e_j.

En considérant que les mots suivent une chaîne de Markov du premier ordre, on a :

$$p(e_1,.....e_n / M) = \prod_{j=2}^{n} p(e_j / e_{j-1}, M) \qquad (4.6)$$

où les quantités $p(e_j / e_{j-1})$ peuvent être estimées par l'algorithme de Baum-Welch .

On a montré sous ces deux hypothèses que la probabilité d'apparition de la séquence d'observation selon un chemin pour le modèle mot est donnée par :

$$p(o_1,.....o_n,e_1,.....e_n / M)=\prod_{j=1}^{n} p(o_j / e_j,M)\prod_{j=2}^{n} p(e_j / e_{j-1},M) \qquad (4.7)$$

Les modèles de Markov à temps discret ont deux inconvénients le premier est que l'apprentissage n'est pas discriminant en ce sens que les paramètres de chaque modèle mot ne sont ajustés qu'avec les images de mots associés à sa classe. D'autre part, la faiblesse du système discret est que les observations sont souvent des vecteurs continus, l'utilisation de modèles à distributions discrètes implique donc une phase préalable de quantification des vecteurs, avec les dégradations qui en résultent. Il est dés lors intéressant d'inclure des densités d'observations dans les modèles de Markov en utilisant les réseaux de neurones, qui intègrent des fonctions discriminantes à la suite d'un apprentissage par l'exemple.

La solution adoptée pour estimer les probabilités d'apparition d'observations est d'utiliser un réseau de neurones multicouches (PMC) : Multilayer perceptron.

4.5.2 Le perceptron multicouches et estimation des paramètres du réseau neuro-markovien:

Topologiquement, le perceptron multicouches est constitue de trois couches comme détaillé dans la section 1.5.4.

La première couche est la couche d'entrée constituée du vecteur des caractéristiques obtenu à partir de l'image d'entrée, la deuxième couche comporte les unités cachées. Enfin la couche de sortie est dimensionnée au nombre de classes à discriminer.

Un réseau de neurones est un système à apprentissage qui à partir d'une base d'exemples contenant des formes d'entrées et les sorties associées ajuste ses paramètres internes, dans notre cas les poids synaptiques (w_{ij})et (z_{ij}). A l'issue d'un entraînement optimal, les sorties du perceptron multicouches constituent une bonne estimation des probabilités à posteriori que possède chaque classe d'être celle de l'objet présenté à l'entrée du réseau de neurones.

Les sorties du réseau approximent asymptotiquement les probabilités bayesienne d'appartenance aux classes de la forme présentée [36, 154]. Au terme de l'apprentissage on peut donc écrire pour la forme x et la classe k :

$$p(c_k / x) \approx S(k,x) \qquad (4.8)$$

Plus simplement l'apprentissage est discriminant en ce sens que lors de l'apprentissage, on force le réseau à mettre à 1 la sortie correspondant à l'exemple présenté à l'entrée et à 0 les autres sorties.

- ***Estimation des paramètres par le réseau PMC***

Nous utilisons un perceptron multicouches pour estimer les probabilités d'émission $p(o_j / e_j, M)$.

Chaque état de la chaîne de Markov sera donc considéré comme une classe du réseau et l'observation sera le vecteur de caractéristiques extrait à partir de l'imagette d'un graphème.

L'alignement Markovien utilise par contre la probabilité $p(o/e)$ d'observation de l'imagette *o* depuis l'état *e*, les deux termes sont liés par la formule de bayes :

$$p(o/e) = \frac{p(e/o) \times p(o)}{p(e)} \tag{4.9}$$

et pour une séquence $o_1,.....o_n$ et un chemin $e_1,.....e_n$:

$$p(o_1,.....o_n,e_1,.....e_n / M) = \prod_{j=1}^{n} p(e_j / o_j, M) \times \prod_{j=2}^{n} p(e_j / e_{j-1}, M) \times \frac{\prod_{j=1}^{n} p(o_j)}{\prod_{j=1}^{n} p(e_j)} \tag{4.10}$$

Comme le produit des probabilités des segments d'image $p(o_j)$ ne dépend pas de l'hypothèse de mot M, nous pouvons écrire :

$$p(o_1,....o_n,e_1,....e_n / M) \propto \frac{\prod_{j=1}^{n} p(e_j / o_j, M) \times \prod_{j=2}^{n} p(e_j / e_{j-1}, M)}{\prod_{j=1}^{n} p(e_j)} \tag{4.11}$$

Dans les formules ci-dessus, les termes de forme $p(e_j / e_{j-1})$ sont des probabilités qui peuvent être estimés par comptage à partir d'un dictionnaire, les termes de forme $p(e_j)$ sont les probabilités à priori des états et les termes de forme $p(e_j / o_j)$ sont bien estimés par les sorties du réseau de neurones.

Nous pouvons omettre le mot M dans cette formule puisque les états correspondent aux lettres/graphèmes du mot indépendamment du mot lui même.

4.6 Application

Le modèle de reconnaissance combine un réseau de neurones de type PMC et un modèle de Markov caché et fonctionne selon le schéma figure 4.1 afin de reconnaître un mot.

Le modèle de Markov utilisé est organisé en colonnes d'états, chaque colonne comprend *N* états où *N* est le nombre de toutes les classes possibles des symboles (graphèmes, diacritiques et espace intra-mot) composant les mots du lexique.

Dans notre application, le réseau de neurones est en amont du MMC. Plusieurs études ont démontré q'un perceptron multicouches entraîné dans des conditions adéquates est asymptotiquement équivalent à un estimateur de probabilité a posteriori d'appartenance à une classe [14, 15, 16].

A chaque colonne du MMC est associé le réseau dans lequel chaque neurone de sortie correspond à un état du MMC. Chaque observation est alors associée aux états (classes) de l'alphabet (figure 4.9).

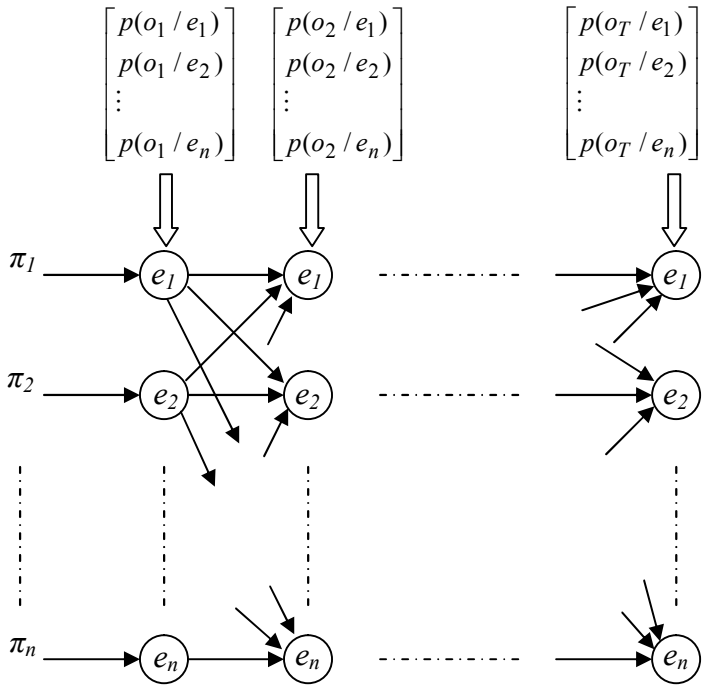


Fig 4. 9 – Schéma du système hybride.

Un montant littéral est une association de graphèmes, de signes diacritiques appartenant à un alphabet, réunis dans des mots d'un vocabulaire donné.

Le système de reconnaissance des mots présentés ici suppose que la segmentation en graphèmes est correcte. D'où la réussite de la reconnaissance des mots dépend fortement de la bonne segmentation des graphèmes dans le mot.

Durant le processus de reconnaissance, les signes diacritiques doivent être associés aux lettres adéquates. Néanmoins, la position des diacritiques est variable, et par conséquent, le processus de reconnaissance devient plus compliqué. Le problème qu'on a rencontré est comment introduire (intégrer) les signes diacritiques dans notre modèle. La modélisation qu'on a faite propose ainsi une solution au problème des diacritiques, ce qui n'a pas été pris en compte jusqu'à présent par les modèles de Markov.

Une fois la segmentation effectuée, il ne reste plus qu'à ordonner les graphèmes et les diacritiques afin de former une séquence d'observations. Ce problème n'est pas simple et doit inclure des étapes de regroupement afin de traiter des problèmes tels que des diacritiques qui doivent être associés aux lettres/graphèmes appropriés. Dans un premier temps, les points diacritiques sont intégrés par l'algorithme d'ordonnancement, ils sont affectés aux graphèmes les plus proches. L'observation de chaque diacritique vient alors juste après le graphème associé, cela signifie que si le model est à l'état i (graphème de la classe i) à l'instant t, alors le modèle sera à l'état j (diacritique classe j) à l'instant t+1. D'où, pour un mot donné, le nombre d'états est égal à la somme du nombre de graphèmes issus de la segmentation et du nombre de points diacritiques associés. Un état fictif désigné par le symbole "#", est ajouté pour représenter l'observation liée à l'espace entre pseudo-mots d'un même mot. Nous obtenons ainsi un alphabet composé de 24 éléments.

La figure 4.10 montre la séquence d'observation du mot **" مائة ".** Seulement les parties encerclées symbolisent la séquence d'observation, les parties restantes représentent des liaisons inutiles (ligatures horizontales) qui doivent être enlevées.

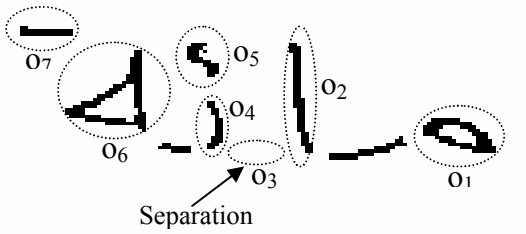


Fig 4. 10 – Séquence d'observations du mot " مائة ".

Dans notre application, nous avons adopté une modélisation (Path Discriminant) qui dispose d'un seul modèle pour toutes les classes des mots, par recherche du chemin optimal. Après décodage, la série d'observation est associée à un chemin optimal, c'est-à-dire celui qui maximise la probabilité d'observation selon la formule (4.11). Ce chemin fournit alors un étiquetage pour chaque observation en indiquant l'état de chaîne de Markov cachée auquel elle est associée. L'apprentissage consiste à estimer les probabilités de transitions et

d'observations pour l'ensemble des graphèmes constituant le lexique. Les différentes probabilités de transitions entre états sont estimées par comptage sur toute la base d'apprentissage où les transitions "ن→ و" des exemples du mot "أربعون " ou " ستون" alimentent donc la même transition "ن→ و". Les probabilités d'observation des graphèmes sont estimées par les sorties du réseau de neurones, ce réseau ayant autant de sorties que d'états possibles. Au total 18 graphèmes sont utilisés dans les 48 mots du vocabulaire des montants des chèques. Par conséquent, c'est un réseau à 18 classes qu'il faut construire dans ce cas. En plus, l'existence de chaque signe diacritique est considérée comme un état du MMC avec une probabilité d'observation égal à 1. De la même façon, la présence d'un espace intra-graphèmes est aussi associée à un état avec une probabilité d'observation égal à 1. D'où, le MMC se compose de 24 états.

On considère une séquence d'observations obtenue à partir d'une image segmentée en graphèmes (figure 4.10), le système de reconnaissance doit retrouver ces graphèmes, les reconnaître d'abord individuellement, puis les valider par reconnaissance lexicale des mots qui les contiennent. Les graphèmes résultants sont ordonnés de gauche à droite selon leur apparition dans le mot. D'où, le mot est décrit par une liste ordonnée de graphèmes et leurs diacritiques associés.

La méthode de reconnaissance analytique basée sur la segmentation en graphèmes comporte alors une première phase de classification des graphèmes en utilisant le PMC comme classifieur, puis une phase de vérification de la concordance des graphèmes avec les mots du lexique. La vérification est un simple matching qui consiste à trouver le mot dans le dictionnaire qui correspond exactement au mot à reconnaître. Suivant cette approche, les graphèmes assument le rôle des lettres dans le dictionnaire.

L'exemple de la reconnaissance du mot "مائة " avec les graphèmes montre en détail le fonctionnement de cette procédure (figure 4.10).

Un mot est considéré comme reconnu si tous les graphèmes et les diacritiques qui le composent sont correctement classifiés. Si ce matching n'arrive à reconnaître aucun mot de la liste des mots candidats, on a alors recours au système hybride. Le rôle du PMC est alors d'assigner une probabilité bayesienne a posteriori aux différentes classes de graphèmes utilisés dans notre alphabet.

Pour reconnaître un mot par le système hybride, on calcule la vraisemblance des mots en faisant la somme des probabilités sur tous les chemins possibles à travers le modèle MMC.

Ensuite, on utilise l'algorithme de Viterbi afin de trouver le chemin optimal qui représente la séquence d'états du mot reconnu. Ce chemin fournit alors une suite d'étiquettes de graphèmes et diacritiques qui représentent le mot reconnu.

Pour une image de mot à reconnaître, on va retenir une et une seule succession d'états (classes) fournie par le MMC qui maximise la probabilité d'observation.

4.7 Résultats expérimentaux

Pour l'évaluation de notre système, nous avons utilisé une base de données semblable à celle présentées dans [158]. La base de données consiste de 48 classes différentes de mots formant un lexique de 48 mots, voir tableau 4.1. Les résultats des expériences sont validés sur une base des données de 7200 images de mots, chaque mot est écrit trois fois par 50 scripteurs différents

ملياران	ألفا	اربعمائة	ستون	تسعة	احدى
ملايير	الفان	خمسمائة	سبعون	عشر	اثنان
سنتيم	مليون	ستمائة	ثمانون	عشرة	ثلاثة
و	ملايين	سبعمائة	تسعون	اثنا	اربعة
دينار	مليونا	ثمانمائة	مائة	عشرون	خمسة
دنانير	مليونان	تسعمائة	مائتا	ثلاثون	ستة
سنتيمات	مليار	ألف	مائتان	اربعون	سبعة
جزائري	مليارا	الاف	ثلاثمائة	خمسون	ثمانية

Tableau 4. 1 – La base des données consiste de 48 classes différentes de mots.

La figure 4.11 montre quelques échantillons extraits de la base de données.

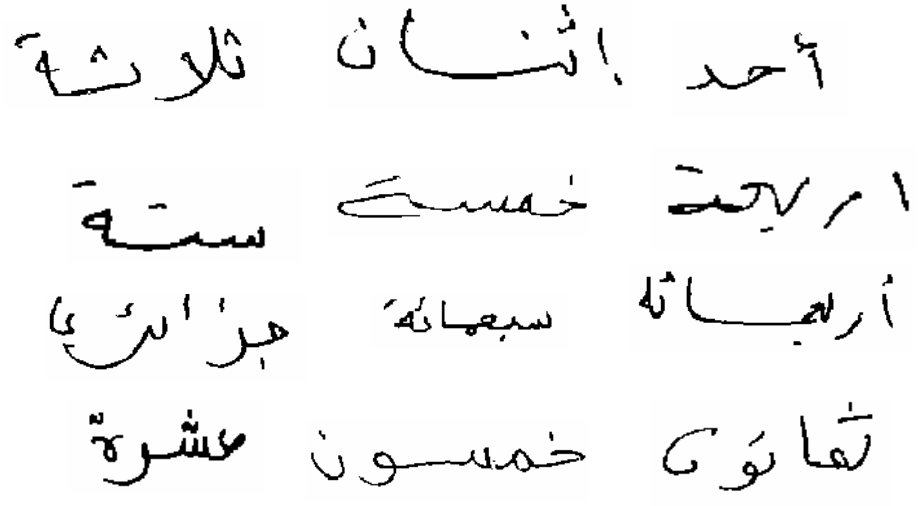


Fig 4. 11 – Quelques échantillons extraits de la base de données.

Pour l'algorithme de segmentation, les résultats expérimentaux montrent que l'algorithme atteint environ 94% de segmentation correcte. Cependant, des cas exceptionnels des règles conduisent à sur-segmentation ou sous-segmentation, nous devons donc tenir compte davantage de critères pour obtenir des résultats de segmentation plus précise.

La figure 4.12-a illustre une certaine confusion due aux sur-segmentations causées par l'existence de parties redondantes comme les branches parasites et de petites boucles où les règles utilisées n'arrivent pas à segmenter correctement les pseudo-mots. On observe que uniquement les PSPs p1 et p2 doivent être maintenus et tous les autres doivent être enlevés.

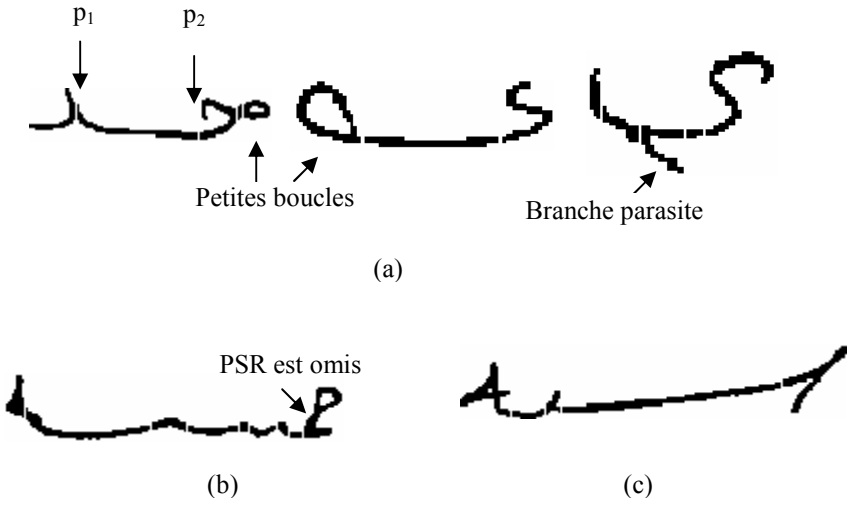


Fig 4. 12 – Types de segmentations incorrectes: (a) Sur-segmentation: p1 et p2 sont des RPSs et tous les autres segments doivent être enlevés, (b) Exemple de sous-segmentation (ligature verticale), (c) les segments verticaux de "سـ" sont omis

Certains caractères peuvent être ligaturés verticalement où deux jusqu'à quatre caractères immédiatement voisins peuvent superposés verticalement pour donner naissance à des dessins assez particuliers dont la morphologie est complètement différente de celle des caractères qui les constituent. Ceci rend la segmentation en graphèmes impossible à l'aide de notre algorithme. La figure 4.12-b montre une ligature verticale de deux caractères (حـ) et (مـ) où le point de segmentation entre les deux graphèmes n'est pas détecté, ce qui conduit à une sous-segmentation des mots. En plus, des segments verticaux de certains caractères, comme le caractère "سـ", ne sont pas dessinés ce qui rend leur segmentation très délicate (figure 4.12-c).

Pour la reconnaissance des mots, la base de graphèmes a été crée automatiquement par segmentation à partir de la base de données utilisée. Pour les jeux d'apprentissage et test, nous n'avons sélectionné que les mots qui ont été segmentés correctement par le module de segmentation lors d'une expérience utilisant un lexique de 7200 images de mots. La base de graphèmes a été également utilisée pour déterminer l'architecture du classificateur neuronal.

Le classificateur neuronal est basé sur un PMC à trois couches. Afin d'assurer que la sortie du classificateur estimerait des probabilités bayesiennes a posteriori, nous avons utilisé une fonction de coût basée sur le carré de l'erreur et une fonction d'activation sigmoïdale lors d'un apprentissage par rétro-propagation.

La base de données est divisée en trois parties, une partie pour l'entraînement du réseau et les deux autres parties pour le test. Ensuite, nous avons étudié l'influence du nombre des neurones cachés M sur les performances du classifieur. L'architecture du PMC (32–25–18) a été déterminée expérimentalement en fonction des résultats sur les bases d'apprentissage et de test de la base des graphèmes. La meilleure performance du classificateur PMC est de 94,60% avec un nombre de neurones cachés égal à 25.

Les confusions les plus fréquentes générées par le PMC sont liés à des configurations spécifiques des pairs de graphèmes, par exemple"**ـمـ**"-"**ـفـ**", "**ه**"-"**ـه**", "**ـة**"-"**ـف**", "**ـمـ**"-"**ـه** ", "**ـعـ**"-"**و** ", "**د**"-"**ـحـ** ", "**ن**"-"**لا**"-"**ـمـ**"-"**ـحـ** ", "**ن**"-"**ر**", et "**ـمـ**"-"**ـحـ**". Après avoir analysé les erreurs, nous avons observé que beaucoup de confusions ont eu lieu entre des paires de graphèmes visuellement similaires. Dans de tels cas, de nombreux graphèmes, ont été regroupés l'un près de l'autre dans l'espace de caractéristiques ne laissant aucune chance de les reconnaître séparément l'un de le l'autre.

Parmi les confusions du système neuronal levées par le système neuro-markovien est le cas du mot "**عشرون** " où le PMC n'arrive pas à reconnaître le premier caractère "**عـ**" et reconnaît parfaitement les autres graphèmes. Toutefois, le système neuro-markovien propose le mot " **عشرون**" dans la première position avec une probabilité d'observation de *1.654 exp(-15)* .

Une autre ambiguïté entre la lettre"**ـمـ**" et "**ه**" est produite par le PMC et levée par le système neuro-markovien. Le réseau de neurones propose la lettre "**ه**" avec une probabilité a posteriori de 0,9903 en première position au lieu de "**ـمـ**" du mot " **خمسة**" avec une probabilité a posteriori de 0,1006. Toutefois, le système neuro-markovien reconnaît bien le mot " **خمسة**" en première position avec une probabilité d'observation de 8.2742 exp(-8)

Pour démontrer l'efficacité du système hybride MMC-PMC, une comparaison avec le classifieur PMC seul a été faite. Le PMC est un état de l'art des techniques de classification et

il a été appliqué avec succès à de tels problèmes réels comme la reconnaissance de caractères et chiffres isolés. Une façon d'observer la contribution de l'information contextuelle est de mesurer les taux de reconnaissance au niveau graphèmes (par le réseau de neurones) et au niveau mots (par le système neuro-markovien), puis nous faisons la comparaison.
Le tableau 4.2 présente les résultats de la reconnaissance des graphèmes en utilisant le PMC et la reconnaissance des mots en utilisant le réseau hybride MMC-PMC.

Classifieur	Taux de reconnaissance	Taux d'erreur
MLP	94.60%	05.40%
HMM-MLP	97.20%	02.80%

Tableau 4. 2 – Taux de reconnaissance et d'erreur pour les deux classifieurs.

Le taux de reconnaissance aux niveau graphèmes est de 94.60 %. Cela montre que la majorité des graphèmes des mots sont reconnues du réseau MLP. La prise en compte du contexte par système hybride apporte une augmentation du taux de reconnaissance d'environ 3 %.
Dans cette étude, nous nous sommes limités au classifieur PMC en raison de leurs excellentes propriétés de classification. En outre, la littérature a montré de meilleurs résultats pour la reconnaissance de chiffres à l'aide du PMC [155, 156, 157].
La comparaison avec les méthodes publiées est très délicate en raison de l'utilisation de différentes bases de données, différents nombres d'échantillons d'apprentissage et de test, et aussi des caractéristiques et des algorithmes de reconnaissances différents. Par exemple, Farah et al. [158] a présenté une approche en utilisant une base de données contenant 4800 mots similaires à la notre pour la reconnaissance des montants littéraux arabes et ils revendiquent un taux de reconnaissance de 96%. Menasri et al. [126] et Benouareth et al. [159] ont évalué leurs approches sur la base de données l'IFN / ENIT. Les résultats obtenus sont de 87,4% et 90,20%, respectivement. Par rapport à ces résultats, nous avons atteint un taux de reconnaissance plus élevé évalué sur une base de données de 7200 mots. En outre, l'avantage de cette approche est la réduction du lexique en utilisant un algorithme sophistiqué de segmentation tout en introduisant une nouvelle modélisation des mots. Cette approche, à notre connaissance, n'a pas été appliquée, telle qu'elle est présentée dans ce travail pour la segmentation et la reconnaissance des mots Arabes dans un lexique restreint. Comme nous le savons, pour la reconnaissance des mots à base de MMC, il existe deux approches principales: la première repose sur une segmentation implicite par la technique de

la fenêtre glissante [160, 161]. Les systèmes de reconnaissance qui obtiennent les meilleurs taux de reconnaissance sur le manuscrit arabe sont basés sur cette architecture. G. R. Ball et al. [162] indiquent qu'une segmentation par fenêtre glissante donne de meilleures performances que leur segmentation en graphèmes. En revanche, ils montrent également qu'une segmentation idéale donnerait de meilleurs résultats que la segmentation par fenêtre glissante. Ce résultat suggère qu'une segmentation en graphèmes de meilleure qualité pourrait offrir de meilleures performances que l'approche par fenêtre glissante.

La seconde, à laquelle appartient notre approche, utilise une technique de segmentation explicite plus sophistiquées [137, 126] pour segmenter les mots en unités plus significatifs ou graphèmes, qui ne sont pas forcement des lettres entières. Comme décrit dans [126], nous estimons que la technique de segmentation explicite en graphèmes est bien adaptée à l'écriture Arabe. Une des raisons est que certaines lettres comme ر ou و ont des descendants qui peuvent se prolonger horizontalement sous la ligne de base, ce qui introduit un chevauchement vertical entre la lettre qui comprend le descendant et la lettre suivante, ce qui se passe souvent dans l'écriture Arabe. La construction d'une séquence de graphèmes par segmentation explicite résout ce problème intrinsèque, alors que les fenêtres glissantes seront obligées de traiter une partie d'image qui contient des parties de deux lettres différentes en même temps. En plus, les signes diacritiques ne sont souvent pas à la position exacte par rapport à la partie principale de la lettre. Par conséquent, l'approche de la fenêtre glissante sera finalement scindé les lettres de leurs signes diacritiques, et réduit ainsi la précision de reconnaissance de caractères.

Par comparaison avec le travail Menasri et al [126, 127], leur approche introduit un nouvel alphabet à base de corps de lettres afin de réduire la taille du lexique, qui est semblable à la nôtre. Cependant, notre alphabet est plus réduit comme la lettre ســ a été segmentée en trois portions (graphèmes). Chaque portion a la même classe que la forme principale de la lettre "ـبـ" (figure 4.3). En outre, leurs modèles des mots sont construits par la concaténation des modèles de lettres qui les composent où chaque lettre est modélisée par un MMC élémentaire. Cependant, dans notre approche du chemin discriminant, un seul MMC est utilisé pour toutes les classes des mots, ce qui réduit considérablement le nombre de paramètres du MMC. Ainsi, il peut être réalisée plus efficacement par matching avec toutes les classes des mots en même temps que par matching, un par un dans les autres méthodes.

4.8 Conclusion

Dans ce chapitre, nous avons présenté un nouveau système basé sur une segmentation explicite pour la reconnaissance des mots arabes manuscrits. Nous avons utilisé un seul MMC

pour modéliser explicitement les mots segmentés en graphèmes, conduisant à une meilleure discrimination entre eux. Les probabilités a posteriori sont calculées par le perceptron multi-couches, les probabilités de transition sont estimées par comptage, et les probabilités a priori sont obtenues en calculant le nombre d'occurrences de chaque état (classe) dans la base d'apprentissage. Au début, le PMC est utilisé comme un classifieur pour reconnaître les graphèmes du mot à reconnaître. Si le PMC n'arrive pas à reconnaître les graphèmes composant un mot candidat, les sorties du PMC sont utilisées comme des probabilités a posteriori des graphèmes et le système neuro-markovien est ensuite appliqué. Enfin, l'algorithme de Viterbi est utilisé afin de trouver le chemin optimal représentant le mot reconnu.

Pour résumer, les principaux avantages de cette application sont :

- Car la classification dans un petit alphabet est à la fois plus efficace et plus précise que dans un grand alphabet, nous avons introduit un nouvel alphabet à base de graphèmes pour la reconnaissance des mots manuscrits arabe où le nombre de graphèmes à reconnaître est plus petit que le nombre de caractères de l'alphabet des montants littéraux arabes.
- Le réseau PMC a autant de sorties que de graphèmes composant l'alphabet. Typiquement, dans cette application, il y a 18 graphèmes. En outre, le PMC a une couche d'entrée de petite taille ; 32 paramètres d'entrée. Par conséquent, notre système est plus simple en terme de calcul si l'on compare le nombre de ses paramètres avec le nombre des paramètres des autres systèmes dans la littérature. L'intérêt principal de cette technique est qu'un nombre important des mots sont efficacement reconnus (94,60%) en utilisant le PMC comme classifieur sans l'intervention du MMC, cette approche est ainsi effectuée à un faible coût de calcul et nécessite moins de mémoire.
- Le principal argument pour l'utilisation du PMC pour obtenir les probabilités de sortie, c'est que celui-ci est entraîné d'une façon discriminative et qu'aucune hypothèse n'est faite sur leurs distributions contrairement aux mélanges de gaussiennes.
- Enfin, la contribution la plus intéressante de ce travail est l'algorithme de segmentation. L'avantage de cet algorithme est qu'il est plus facile de trouver l'ensemble de points de segmentation potentiels car il analyse la forme structurelle des caractères tels qu'ils ont été scannés sans aucune transformation (projection, amincissement) et il est indépendant de la ligne de base, ce qui le rend plus robuste en particulier pour l'écriture manuscrite par rapport aux algorithmes existants.

Le principal inconvénient de cet algorithme est la sous-segmentation, mais nous pouvons y remédier en utilisant des règles plus appropriées. De plus, certains signes diacritiques sont

parfois omis par les scripteurs. L'omission de certains signes diacritiques et les erreurs de segmentation peuvent également être remédiées en utilisant une correction lexicale par une distance d'édition appliquée en phase de post-traitement entre la séquence de symboles reconnue, et la séquence de symboles correspondant à un mot donné.

Dans l'avenir, nous aimerions étendre cet algorithme pour la segmentation des textes arabes en caractères où le mot sera segmenté en graphèmes puis les graphèmes seront recombinés pour former des caractères.

Chapitre 5 - Reconnaissance de chiffres farsis manuscrits

Dans ce chapitre, nous proposons un système de reconnaissance de chiffres farsis/persans qui pourrait être utilisé à la lecture automatique des montants numériques de chèques. Dans ce travail, nous présentons un nouveau jeu de primitives pour la caractérisation des chiffres manuscrits. Pour montrer l'efficacité de cette approche dans le domaine de la reconnaissance de chiffres manuscrits, trois types de classifieurs (PMC, SVM et neuro-flou) sont évalués et comparés. L'utilisation conjointe de SVM et du nouveau jeu de primitives a permis d'obtenir un meilleur résultat. Le système de reconnaissance est évalué sur une grande base de données de chiffres persans manuscrits, nommé "Hoda" [163].

5.1 Base de données : HODA farsi digits

Cette base de données a été rendue publique en février 2007 par H. Khosravi et E. Kabir [163]. Il s'agit d'une base de chiffres farsi manuscrits isolés. Elle est composée de 102352 chiffres extraits à partir d'environ 12000 formulaires, remplis par des étudiants et des personnels universitaires. Cette base de données se compose de dix classes de chiffres de 0 à 9 (۰ ۱ ۲ ۳ ۴ ۵ ۶ ۷ ۸ ۹).

La base est constituée de 6.000 échantillons par classe (60000 échantillons) pour l'apprentissage et de 2.000 échantillons par classe (20000 échantillons) pour le test, plus un troisième ensemble qui contient des données restantes (22352 exemples).

Les images sont stockées dans une base de données au format CDB. Deux codes en C++ et en Matlab [163] permettent de lire les chiffres isolés.

La figure 5.1 montre un exemple de caractères farsi manuscrits

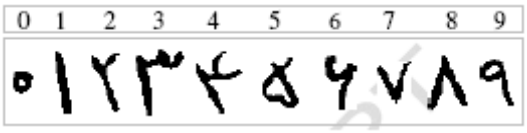
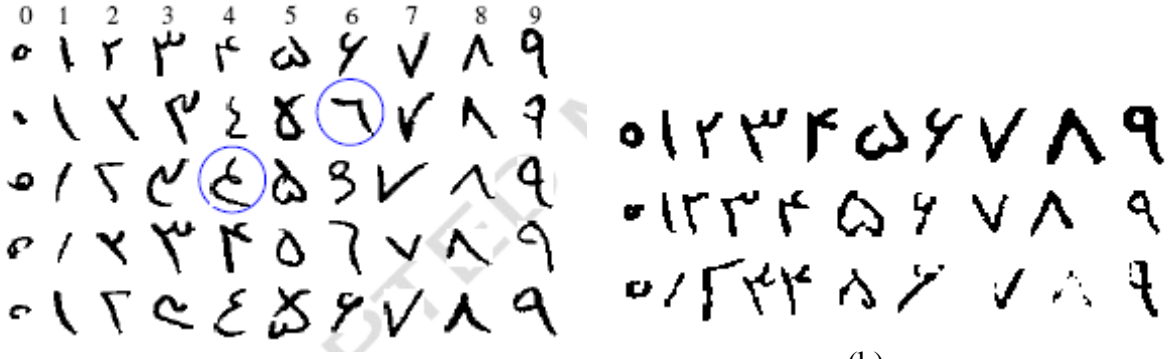


Fig 5. 1 – Exemples de chiffres farsi manuscrits.

La figure 5.2 illustre différents styles et qualités des échantillons de la base de données.



(a) (b)

Fig 5. 2 – (a) différents styles des chiffres farsi manuscrits, (b) échantillons des chiffres de différentes qualités.

5.2 Prétraitement

Les prétraitements qu'on a faits sont la binarisation et l'extraction de contours.

La binarisation est faite à l'aide d'un seuil adapté à chaque image du chiffre. Pour chaque chiffre, un simple seuillage global suffit pour la binarisation. Chaque pixel de l'image est comparé à ce seuil : ceux de niveaux de gris inférieurs sont mis à 0 «noir », ceux supérieurs mis à 1 «blanc». Ensuite, l'opérateur de Canny est utilisé pour l'extraction des contours des chiffres.

5.3 Extraction de caractéristiques

Cette fois-ci, deux types de caractéristiques sont utilisés : l'histogramme des orientations locales des images des contours des chiffres et une autre variantes des caractéristiques mesurées à partir des transitions blancs/noirs de l'image du chiffre.

L'histogramme des orientations locales des images des contours permet d'obtenir une représentation invariante à la translation et par rapport à l'échelle.

La boîte englobante de chaque contour de chiffre est divisée en neuf zones égales et au sein de chaque zone, la contribution des 4 codes de Freeman est comptée dans les histogrammes correspondants.

L'histogramme résultant est normalisé en module. Le vecteur de caractéristiques du contour est alors composé de 3×3×4 (36) composantes normalisé entre 0 et 1.

$$f_i = \frac{h_i}{\|h_i\|}, \quad i = 1,2 \ldots\ldots 36 \qquad (5.1)$$

Le deuxième ensemble des caractéristiques est basé sur des informations extraites à partir du nombre de transitions d'un pixel blanc vers un pixel noir dans les deux directions horizontale et verticale.

Les caractéristiques définies par les transitions blancs/noirs de l'image du chiffre sont caractérisées par la surface qui se trouve entre le bord du rectangle englobant et le contour que fait chaque type de transition. On a trois types de caractéristiques : la première correspond aux parties de chiffres qui ont une seule transition, la deuxième correspond aux parties qui ont deux transitions et la troisième correspond à ceux qui ont plus de deux transitions. Dans la direction horizontale, le rectangle englobant le chiffre est devisé en deux bandes horizontales de même surface et pour chaque bande, les caractéristiques de transitions horizontales sont extraites. Dans la direction verticale, les caractéristiques sont calculées sur le chiffre entier sans aucune division. Une autre caractéristique est définie comme le rapport entre la surface du rectangle englobant le chiffre et la plus grande surface des chiffres de la base d'apprentissage. Au total, 46 caractéristiques sont extraites à partir des chiffres.

5.4 Reconnaissance

Nous avons implémenté trois classifieurs : les réseaux de neurones de type Perceptron multicouches (PMC), les machines à vecteurs de support (SVM-Support Vectors Machines) et un système hybride neuro-flou. Les trois architectures implémentées ont été entraînées sur une base d'apprentissage (60000 échantillons) et testées sur une base de test (20000 échantillons) en utilisant les mêmes vecteurs de primitives.

5.4.1 Classification par un PMC

Les réseaux de neurones de type PMC sont parmi les techniques les plus couramment utilisées. Ils ont prouvé une grande efficacité notamment en classification [155, 156, 157]. Le réseau est constitué d'une couche d'entrée à 46 neurones (correspondants aux 46 primitives retenues), d'une couche cachée ayant un nombre de neurones variant, ce nombre est déterminé pendant la phase d'apprentissage et d'une couche de sortie à 10 neurones correspondant au 10 classes des chiffres farsis. Nous essayons différentes topologies de réseaux de neurones pour déterminer la meilleure configuration. Un perceptron Multicouches qui contient 25 neurones sur la couche cachée permet d'obtenir les meilleures performances (98.48 %).

La matrice de confusion est donnée dans le tableau 5.1 où r et e représentent respectivement le taux de reconnaissance et le taux d'erreurs pour chaque classe.

	0	1	2	3	4	5	6	7	8	9	r (%)	e (%)
0	1985	1	0	1	1	9	0	2	0	1	99.2500	0.7500
1	1	1988	4	0	2	0	0	0	0	5	99.4000	0.6000
2	0	0	1952	24	11	1	3	1	0	8	97.6000	2.4000
3	0	0	59	1917	20	2	0	1	1	0	95.8500	4.1500
4	0	0	12	17	1964	2	2	0	0	3	98.2000	1.8000
5	5	0	0	0	6	1986	1	0	2	0	99.3000	0.7000
6	0	4	5	0	7	3	1956	0	0	25	97.8000	2.2000
7	0	2	7	1	0	1	1	1988	0	0	99.4000	0.6000
8	0	3	0	0	0	1	0	0	1990	6	99.5000	0.5000
9	1	14	2	0	3	2	6	1	2	1969	98.4500	1.5500

Tableau 5. 1 – Matrice de confusion pour la reconnaissance par le PMC. Les colonnes correspondent aux valeurs reconnues, les lignes aux valeurs vraies.

L'analyse de la matrice de confusion permet de déterminer deux types de confusions :

– certains chiffres '6' sont reconnus comme des '9'

– des confusions entre les chiffres '2', '3' et '4' où la confusion est plus problématique. Ces chiffres sont parfois visuellement très proches où la décision est difficile, même pour un opérateur humain.

5.4.2 Classification par les SVMs

Un SVM est un classifieur binaire, il est donc nécessaire de combiner plusieurs SVMs pour résoudre un problème multi-classes. La stratégie la plus classique est le « un contre tous » qui consiste à construire un SVM par classe. Chaque classifieur est alors entraîné à distinguer les exemples de sa classe de ceux de toutes les autres classes. Les structures des classifieurs ont été empiriquement fixé comme suit. Le classifieur SVM utilise un noyau RBF avec le paramètre de variance σ égal à 1 ($\sigma = 1$) et le hyperparamètre C égal à 100 ($C = 100$). Le vecteur de caractéristiques utilisé est égal à $10.f$, où f est le vecteur de caractéristiques décrit ci-dessus.

Le tableau 5.2 montre le détaille des taux de reconnaissance et d'erreur pour chacun des chiffres où la précision de la reconnaissance de 98,48 % est obtenue sur l'ensemble du test (20000 échantillons).

Chiffre (Farsi)	Nombre de chiffres mal classés	Taux de reconnaissance (%)
0	14	99.30
1	7	99.65
2	42	97.90
3	109	94.55
4	32	98.40
5	15	99.25
6	30	98.50
7	20	99.00
8	4	99.80
9	32	98.40

Tableau 5. 2 – Chiffres mal classés et taux de reconnaissance pour chaque chiffre.

Il convient de noter que sur 305 échantillons méconnu, 183 (60 %) échantillons appartiennent à l'ensemble des classes {2, 3, 4} et 122 (40 %) échantillons appartiennent aux classes des chiffres restants. Par conséquent, un classifieur qui n'est pas en mesure d'atteindre une bonne précision dans toutes les classes, il peut obtenir une bonne précision sur un nombre réduit de classes. Pour améliorer la précision de la reconnaissance, nous utilisons un système similaire à celui introduit dans [164].

La procédure de reconnaissance est faite selon le schéma de la figure 5.3. Nous utilisons d'abord des SVMs un contre sept où {0, 1} et {2, 3, 4} représentent deux classes distinctes. Ensuite, des SVMs sont utilisées pour classifier les chiffres {0, 1, 2, 3, 4}.
La figure 5.3 illustre le système de reconnaissance où les poids des flèches représentent le nombre des chiffres non reconnus de chaque classe.

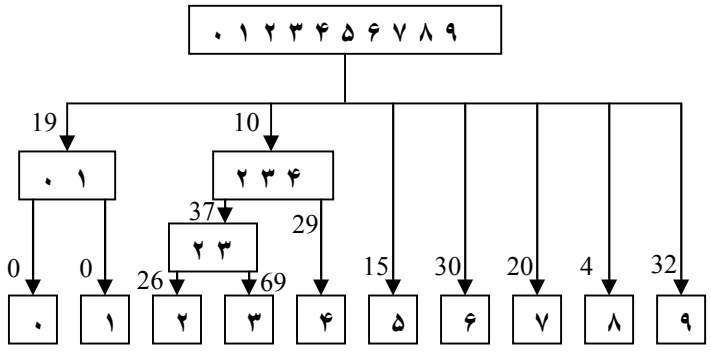


Fig 5. 3 – Système de reconnaissance: les poids des flèches représentent le nombre de chiffres non reconnus de chaque classe.

Sur un total de 20000 chiffres dans l'ensemble de test, il y a 293 chiffres qui ne sont pas correctement reconnu. Par conséquent, on obtient un taux de reconnaissance de 98,53%, qui montre une légère amélioration par rapport à la première phase. Notez que nous avons obtenu une précision de 100% sur les échantillons d'apprentissage (60.000 échantillons). Nous avons également une précision de 100% lorsque l'apprentissage se fait sur l'ensemble des données (80.000 échantillons) et le même ensemble est utilisé pour la phase de test.

5.4.3 Système neuro-flou

Le système d'inférence floue (FIS) choisi utilise des règles de Sugeno d'ordre 0. L'objectif de cette partie est de remplacer l'expert humain pour automatiser la tâche de génération de règles, en utilisant une approche hybride neuro-floue. Chacune des techniques a des propriétés particulières. En effet, les réseaux neuronaux sont des excellents classifieurs pour la reconnaissance de formes. Cependant, ils sont incapables d'expliquer comment ils atteignent leurs décisions. Aussi pour les systèmes de la logique floue qui peuvent raisonnez avec l'information imprécise et expliquer leurs décisions mais ne peuvent cependant pas acquérir automatiquement les règles qu'ils l'utilisent pour prendre ces décisions. Avec l'accroissement de la complexité du modèle, nous rencontrons aussi une difficulté pour développer des règles floues et des fonctions d'appartenance. Cela a mené au développement d'une autre approche telle que l'approche hybride Neuro-floue. Une description de l'architecture du classifieur ainsi que ses règles d'apprentissage sont aussi décrites dans l'annexe A.

L'apprentissage du réseau neuro-flou consiste à régler quatre paramètres principaux: le nombre de neurones dans l'unique couche cachée ou le nombre des gaussiennes, la position des centres et la largeur des gaussiennes, et les poids de connexions entre les neurones cachés et les neurones de sortie. L'apprentissage est effectué en deux phases : dans la première phase, le nombre et les centres des clusters (sous-classes) sont choisis par une technique de partition floue de type fuzzy C-means). Pour initialiser les poids du réseau neuro-flou, les centres des gaussiennes sont les centres des clusters et l'écart type de chaque cluster est considéré comme la largeur de la gaussienne correspondante. D'où, on a réalisé une méthode simple et directe pour traduire un ensemble de centres de clusters calculés dans la phase non supervisée en un système d'inférence flou [121].

Les paramètres des gaussiennes avec des valeurs aléatoires des paramètres singletons sont considérés comme les paramètres d'initialisation de la seconde phase d'apprentissage pour

obtenir la base de règles floues finale. L'apprentissage non supervisé et supervisé sont faits consécutivement jusqu'à un meilleur taux de classification est achevé.

Pour évaluer les performances des règles floues extraites par le système neuro-flou, il est nécessaire d'exécuter notre algorithme plusieurs fois avec un nombre différent de clusters pour découvrir la forme adéquate des règles qui correspond aux meilleures performances de classification.

Le meilleur système, avec 20 neurones sur la couche cachée, achève un taux de reconnaissance de 98.34 % sur la base de test. Le tableau 5.3 montre la matrice de confusion où *r* et *e* représentent respectivement le taux de reconnaissance et le taux d'erreur pour chaque classe.

	0	1	2	3	4	5	6	7	8	9	*r* (%)	*e* (%)
0	1988	1	0	0	2	7	0	0	0	2	99.4000	0.6000
1	0	1996	1	0	1	0	1	0	0	1	99.8000	0.2000
2	0	5	1939	20	24	0	8	1	0	3	96.9500	3.0500
3	2	0	61	1906	30	0	0	1	0	0	95.3000	4.7000
4	0	0	17	17	1962	1	2	0	0	1	98.1000	1.9000
5	4	1	0	0	11	1978	2	0	3	1	98.9000	1.1000
6	0	3	9	1	8	4	1956	0	0	19	97.8000	2.2000
7	0	5	3	1	2	0	4	1985	0	0	99.2500	0.7500
8	0	0	0	0	0	1	0	0	1998	1	99.9000	0.1000
9	1	13	2	0	0	2	21	0	1	1960	98.0000	2.0000

Tableau 5. 3 – Matrice de confusion pour la reconnaissance par le réseau hybride neuro-flou. Les colonnes correspondent aux valeurs reconnues, les lignes aux valeurs vraies.

Après un apprentissage optimal du réseau neuro-flou, une base de règles composée de 20 règles floues est dérivée. Les fonctions d'appartenance sont aussi apprises d'une manière automatique à partir des exemples de l'ensemble d'apprentissage.

Les paramètres obtenus après apprentissage seront interprétés sous formes de règles Si-Alors.

La figure 5.4 montre l'activation des règles lorsque un échantillon du chiffre ۳ est présenté à l'entrée du réseau où l'activation est normalisée entre 0 et 1. Chacune des 20 règles est représentée par une barre verticale.

On remarque que seule la règle 12 est fortement activée alors que les autres sont pratiquement nulles. D'où une seule règle (la règle 12) est suffisante pour reconnaître le chiffre ۳.

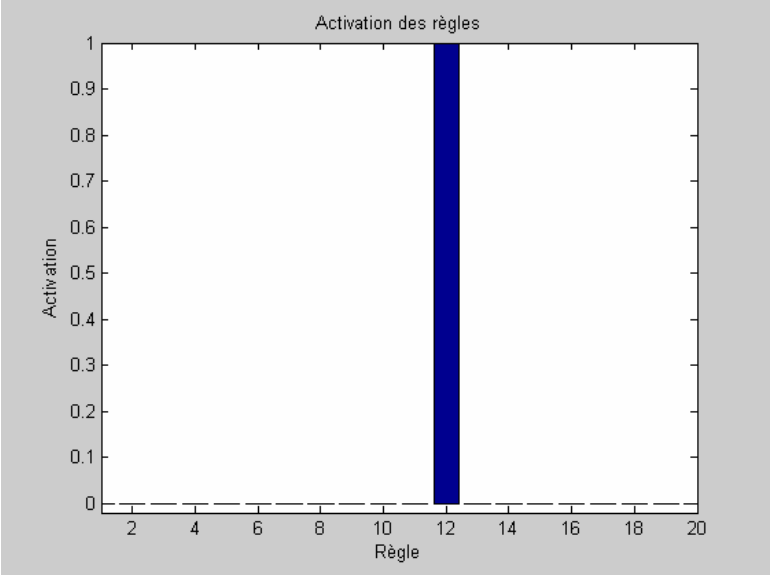


Fig 5. 4 – Activation normalisée entre 0 et 1 des 20 règles obtenues par apprentissage.

La figure 5.5 donne les 10 sorties (y_1, y_2,.... y_{10}), correspondant aux différentes classes des chiffres farsis lorsque un échantillon du chiffre ۳ est présenté à l'entrée du réseau. Les sorties sont normalisées entre 0 et 1 et représentées par des barres verticales.

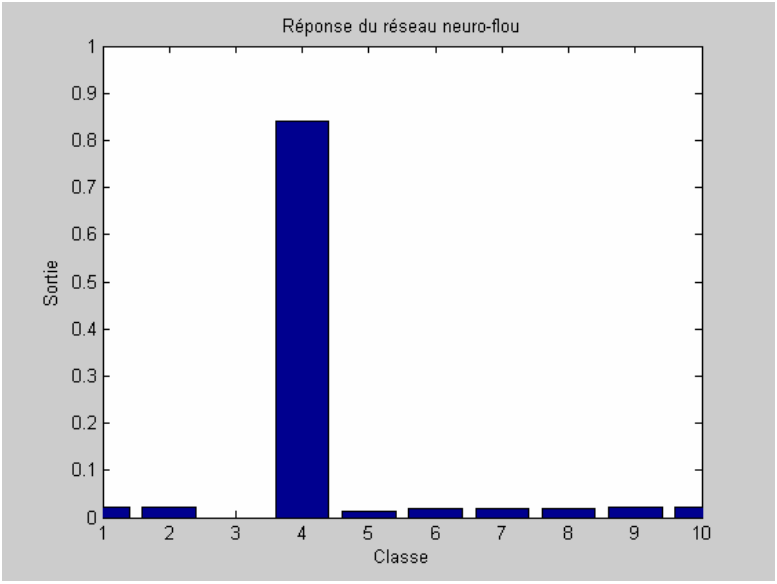


Fig 5. 5 – Les 10 sorties correspondant aux 10 classes des chiffres farsis.

A partir des figure 5.4 et figue 5.5, il est clair que la règle la plus active est la règle 12, ce qui donne une forte activation à la sortie 4 (classe du chiffre ۳) et faible activation aux autres.
Si on prend les dix premières primitives (0.48, 0.41, 0.23, 0.65, 0, 0, 0.39, 0.24, 0, 0.43) du chiffre ۳ comme un exemple, les degrés d'appartenance correspondants sont marqués dans la figure 5.6 par des lignes verticales traversant les dix premières fonctions d'appartenance de la

règle 12. Rappelons que les dix premières primitives sont extraites à partir des transitions blancs/noirs et la surface du rectangle englobant de chaque chiffre.

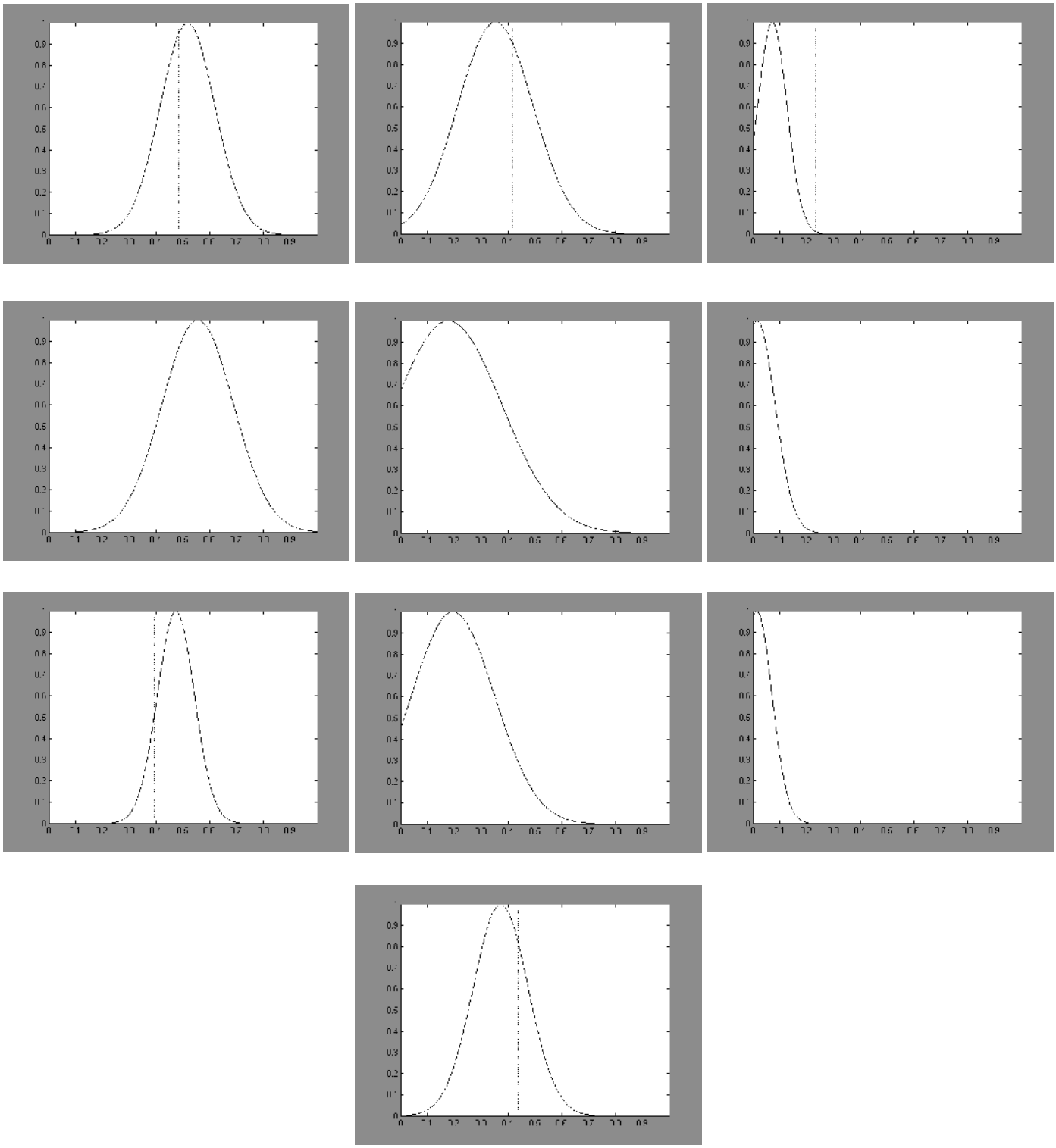


Fig 5. 6 – les dix premières fonctions d'appartenance de la règle 12 : les degrés d'appartenance sont marqués par des lignes verticales.

A l'opposé d'autres approches floues où les règles ont été définies par un expert, les règles ici sont déterminées d'une façon automatique sur l'ensemble d'apprentissage où Chaque règle est interprétable par les analystes (experts).

La tâche de cette étude est de déterminer une configuration appropriée des fonctions d'appartenance et de générer un ensemble de règles floues à partir des données d'apprentissage telles que les sorties estimées du système neuro-flous rapprochent celles désirées. En outre, la méthodologie neuro-flou nous a permis de mieux comprendre les mécanismes de classification, contrairement à ce qui arrive lorsque les réseaux de neurones sont appliqués seuls.

5.5 Comparaison des résultas

Nous présentons (tableau 5.4) les performances comparées obtenues avec les meilleures configurations trouvées à la fois pour des architectures de type MLP, SVM et Neuro-flou. Où, le paramètre Nh désigne le nombre de neurones dans la couche cachée pour les réseaux MLP et Neuro-flou, et les paramètres σ et C désignent respectivement la variance et le hyperparamètre C pour le classifieur SVM.

Réseau	Paramètres	Taux de reconnaissance	
		Base d'apprentissage	Base de test
MLP	Nh=25	99.35 %	98.48 %
SVM	σ = 1 et C=100	100 %	98,53 %
Neuro-flou	Nh=20	99.03 %	98.34%

Tableau 5. 4 – Performances des trois classifieurs sur la base 'Hoda'.

Les meilleurs résultats obtenus sont de 98.48 % pour le réseau PMC, 98,53 % pour le cas du SVM et de 98.34% pour le réseau Neuro-flou.

Les résultas obtenus par le réseau neuro-flou sont légèrement moins performants que ceux obtenus par les modèles de type boîtes noires (SVM et MLP), mais ils sont plus interprétables et plus significatifs pour l'expert, du fait qu'ils se basent sur des règles floues de type

Mamdani. L'analyse des résultats montre que le classifieur SVM est relativement plus performant au niveau performances globales.

Pour valider notre approche nous avons comparé nos résultats avec ceux existants dans la littérature. Le tableau 5. 5 montre une comparaison avec les plus excellents travaux existants, qui sont au meilleur de notre connaissance, les seuls travaux de la littérature qui traitent les chiffres persans de la base de données 'Hoda'.

Nous remarquons que les résultas obtenus par notre approche sont légèrement moins performants de ceux obtenus par trois méthodes sur la base de test, comme le montre le tableau 5.5. Mais ils ont utilisé un nombre important (196) de primitives alors que les taux de reconnaissance dans notre application sont réalisés avec seulement 46 primitives.

Algorithmes	Nombre de primitives	Taille de la base de données		Taux de reconnaissance (%)	
		Apprent	Test	Apprent	Test
[165]	196	60000	20000	99.99	98.71
[165]	196	80000	-----	99.37	-----
[164]	196	60000	20000	99.99	99.02
[166]	106	60000	10000	----	98.89
[167]	106	40000	20000	----	97.12
Algorithme	46	60000	20000	100	98,53
proposé (SVM)	46	80000	----	100	-----

Tableau 5. 5 – Comparaison avec les plus excellents travaux existants.

L'amélioration la plus notable est que le taux de reconnaissance de 100% est atteint dans la phase d'apprentissage pour l'ensemble de tous les exemples (80000 échantillons), qui est supérieur à ceux présentés dans la littérature. Nous estimons que nos résultats sont très compétitifs et très prometteurs car nous avons utilisé un nombre très réduit de primitives (46). Comme nous l'avons atteint un taux de 100 % sur l'ensemble d'apprentissage, alors nous pourrions parvenir à un taux de reconnaissance plus élevé, parce que la propriété de généralisation des SVMs ne dépend pas de toutes les données d'apprentissage, mais seulement des vecteurs de support.

Enfin, nous avons montré que la précision des résultats obtenus est due à la capacité de discrimination des caractéristiques et la capacité de régression des classifieurs SVMs.

5.6 Sélection des caractéristiques

Afin de limiter les coûts calculatoires engendrés par le système de reconnaissance, il est possible de procéder à une réduction de dimensionnalité de l'espace de caractéristiques. Il suffit pour cela de ne conserver qu'un nombre réduit de caractéristiques à chaque essai du classifieur.

Le tableau 5.6 illustre les puissances de discrimination, obtenus sur l'ensemble de caractéristiques sans procédure d'analyse en composantes principales. La première colonne représente les caractéristiques extraites à partir des transitions et les autres représentent celles extraites à partir du contour.

Puissance de discrimination	Puissance de discrimination	Puissance de discrimination	Puissance de discrimination	Puissance de discrimination
0.1553	0.1241	0.1679	0.2145	0.1318
0.2092	0.1926	0.3490	0.2129	0.1849
0.2671	0.1769	0.2135	0.3142	0.1755
0.3647	0.1887	0.1451	0.1790	0.2491
0.8217	0.2023	0.2191	0.2594	0.2347
0.2760	0.1504	0.3249	0.1415	0.2302
0.3711	0.1732	0.1553	0.2287	
0.3879	0.1704	0.1562	0.2178	
0.1391	0.1667	0.1251	0.3810	
0.2024	0.1824	0.1668	0.1901	

Tableau 5. 6 – Puissances de discrimination des 46 caractéristiques utilisées.

Il apparaît clairement ici qu'il n y a pas des caractéristiques de puissances de discrimination très faibles devant les autres, ce qui confirme la pertinence de la totalité des caractéristiques.

L'objectif principal de la sélection des primitives est la réduction du nombre de caractéristiques utilisées tout en gardant la performance de la classification du système de reconnaissance.

La valeur du seuil de puissance de discrimination à utiliser pour sélectionner les primitives doit être déterminée à l'aide d'essais pratiques. D'abord, les primitives ont été classées par ordre décroissant de leurs puissances de discrimination. Nous entreprenons ensuite un apprentissage sur les primitives dont les plus grandes puissances de discrimination (jugées les plus pertinentes).

Le tableau 5.7 reprend les meilleurs résultats obtenus, pour différents nombres de caractéristiques sélectionnées en utilisant le classifieur SVM avec les mêmes paramètres utilisés plus haut ($\sigma = 1$ et C=100).

Seuil de puissance de discrimination	Nombres de caractéristiques sélectionnées	Taux de reconnaissance
0.1500	40	98.46 %
0.1600	36	98.44 %

Tableau 5. 7 – Résultats de sélection de primitives.

Le plus haut taux de reconnaissance atteint ici est de 98.46 % pour 40 caractéristiques et de 98.44 % pour 36 caractéristiques. A titre d'illustration, si nous entreprenons un apprentissage sur les 36 primitives jugées les plus pertinentes, il suffit de retenir les 36 primitives dont les plus grandes puissances de discrimination. C'est à dire que nous pouvons réduire de 10 la dimensionnalité des primitives tout en conservant l'essentiel de l'information.
Le meilleur compromis performance/complexité a été obtenu avec un classifieur SVM ($\sigma = 1$ et C=100) entraîné sur 36 primitives qui est plus performant qu'un réseau neuro-flou entraîné sur toutes les 46 primitives. Par ailleurs, il est important de préciser que l'architecture d'un classifieur, grâce à la réduction de sa taille, demande une moindre capacité de stockage pour ses paramètres.

5.7 Conclusion

Dans ce chapitre, une nouvelle technique efficace d'extraction de caractéristiques basées sur le codage de Freeman avec de nouvelles caractéristiques extraites à partir des transitions blancs/noirs dans l'image des chiffres dans les deux directions horizontale et verticale.

A partir des résultats expérimentaux, il est évident que notre système atteint une bonne performance. Nous avons constaté que la plupart des échantillons qui ont été mal classés appartiennent aux classes (۲, ۳ et ۴), qui sont similaires dans leurs formes. La reconnaissance de ces chiffres similaires est parfois difficile, même pour l'être humain.
Les principaux avantages des caractéristiques extraites sont les suivantes:

- Il ne nécessite pas de normalisation des images des chiffres où la plupart des ouvrages publiés ont besoin de normalisation de chiffres, qui dégradent la qualité de l'image.

À l'avenir, nous prévoyons d'utiliser certaines fonctionnalités comme l'analyse structurelle des concavités qui peut supprimer certains des confusions entre les classes similaires. En ce qui concerne les classificateurs SVM, le choix du hyperparamètre C est une tâche délicate. Les paramètres (σ et C) utilisés dans nos expériences ne sont pas les meilleurs choix et ne sont pas appliquées de façon optimale, parce que nous avons essayé à quelques expériences de les choisir. Ainsi, nous pouvons améliorer les performances du système en testant plusieurs valeurs de ces paramètres en estimant le taux de généralisation.

Conclusion Générale

La reconnaissance de l'écriture manuscrite arabe reste relativement mal servie malgré l'importance de cette dernière dans les différents domaines tels que la lecture automatique des chèques, des formulaires administratifs et des adresses postales. La majorité des solutions proposées a été testée sur l'écriture latine puis appliquée telle quelle pour la reconnaissance de l'écriture arabe imprimée. Ces méthodes supposent généralement que les caractères peuvent être isolés par une étape de segmentation. Cette étape de segmentation est possible dans le cas d'un texte latin imprimé, mais très difficile dans le cas de l'écriture cursive ou semi-cursive, le cas de l'écriture arabe.

L'objectif de cette thèse est de proposer un système de reconnaissance de l'écriture manuscrite hors-ligne.

Dans la première partie de ce travail de thèse, nous nous sommes principalement intéressés à la reconnaissance des montants littéraux arabes manuscrits. L'originalité de notre approche basée sur la segmentation en graphèmes, par rapport aux méthodes classiques, réside dans les points suivants :

- Une nouvelle méthode de segmentation qui est adaptée à la particularité de l'écriture Arabe. L'avantage de cet algorithme est qu'il est plus facile de trouver l'ensemble de points de segmentation potentiels car il analyse la forme structurelle des caractères tels qu'ils ont été scannés sans aucune transformation (projection, amincissement) et il est indépendant de la ligne de base, ce qui le rend plus robuste en particulier pour l'écriture manuscrite par rapport aux algorithmes existants.
- Un nouvel alphabet réduit qui exploite un certain nombre de spécificités de l'écriture arabe est ainsi introduit. Nous avons introduit un nouvel alphabet à base de graphèmes pour la reconnaissance des mots manuscrits arabe où le nombre de graphèmes à reconnaître est plus petit que le nombre de caractères de l'alphabet des montants littéraux arabes.
- Un nouveau modèle hybride de type neuro-Markovien qui permet d'incorporer plus de contexte en attribuant les signes diacritiques à leurs plus proches graphèmes. Ainsi, un seul modèle qui comprend les états (classes) de graphèmes et les états (classes) des signes diacritiques est construit pour représenter la totalité de l'alphabet.

Pour reconnaître un mot segmenté en graphèmes, au début, le PMC est utilisé comme un classifieur pour reconnaître les graphèmes du mot à reconnaître. Si le PMC n'arrive pas à

reconnaître les graphèmes composant un mot candidat, les sorties du PMC sont utilisées comme des probabilités a posteriori des graphèmes et le système neuro-markovien est ensuite appliqué. Enfin, l'algorithme de Viterbi est utilisé afin de trouver le chemin optimal représentant le mot reconnu.

Notre système est plus simple en terme de calcul si l'on compare le nombre de ses paramètres avec le nombre des paramètres des autres systèmes dans la littérature. L'intérêt principal de cette technique est qu'un nombre important des mots sont efficacement reconnus (94,60%) en utilisant le PMC comme classifieur sans l'intervention du MMC, cette approche est ainsi effectuée à un faible coût de calcul et nécessite moins de mémoire.

Le principal argument pour l'utilisation du PMC pour obtenir les probabilités de sortie, c'est que celui-ci est entraîné d'une façon discriminative et qu'aucune hypothèse n'est faite sur leurs distributions contrairement aux mélanges de gaussiennes.

Le principal inconvénient de cet algorithme est la sous-segmentation, mais nous pouvons y remédier en utilisant des règles plus appropriées. De plus, certains signes diacritiques sont parfois omis par les scripteurs. L'omission de certains signes diacritiques et les erreurs de segmentation peuvent également être remédiées en utilisant une correction lexicale par une distance d'édition appliquée en phase de post-traitement entre la séquence de symboles reconnue, et la séquence de symboles correspondant à un mot donné.

Dans l'avenir, nous aimerions étendre cet algorithme pour la segmentation des textes arabes en caractères où le mot sera segmenté en graphèmes puis les graphèmes seront recombinés pour former des caractères.

La deuxième partie décrit un système de reconnaissance de chiffres farsis qui pourrait être utilisé à la lecture automatique des montants numériques de chèques. L'apport principal de cette partie est la synthèse des nouvelles techniques d'extraction de caractéristiques permettant une amélioration des performances des systèmes de reconnaissance. Pour montrer l'efficacité de cette technique, trois types de classifieurs (PMC, SVM et neuro-flou) sont évalués et comparés où le SVM avec ce nouveau jeu de caractéristiques a permis d'obtenir les meilleures performances.

Les résultas obtenus par le réseau neuro-flou sont légèrement moins performants que ceux obtenus par les modèles de type boîtes noires (SVM et MLP), mais ils sont plus interprétables et plus significatifs pour l'expert, du fait qu'ils se basent sur des règles floues de type Mamdani. A l'opposé d'autres approches floues où les règles ont été définies par un expert, les règles ici sont déterminées d'une façon automatique sur l'ensemble d'apprentissage où Chaque règle est interprétable par les analystes (experts).

A. La Logique floue

La logique floue est un concept introduit par Lotfi Zadeh en 1965 et qui utilise la notion de variables linguistiques utilisées en langage humain. Par opposition à la logique classique qui n'admet que deux valeurs : vrai et faux, la logique floue encode les valeurs de vérité dans l'intervalle des nombres réels [0, 1] et utilise une fonction d'appartenance qui met en relation les propositions par leur degré d'appartenance à un ensemble de valeurs.

A.1 Sous-ensemble flou

Un sous-ensemble flou A dans un univers du discours U est caractérisé par sa fonction d'appartenance $\mu_A(x)$ qui associe à chaque élément x de U une valeur dans l'intervalle des nombres réels [0, 1].

La grandeur $\mu_A(x)$ définit le degré d'appartenance de l'élément x à l'ensemble A.

$$\mu_A : U \rightarrow [0,1]$$
$$x \rightarrow \mu_A(x)$$

Si $\mu_A(x)$ =0,30 alors x appartient à l'ensemble flou A avec un degré d'appartenance de 30%

Les fonctions d'appartenance peuvent avoir diverses formes selon leur définition : triangulaire, trapézoïdale, Gaussienne, Sigmoïdes…

La figure A.1 représente quelques fonctions d'appartenance qui prennent différentes formes en fonction de la nature de la grandeur à modéliser.

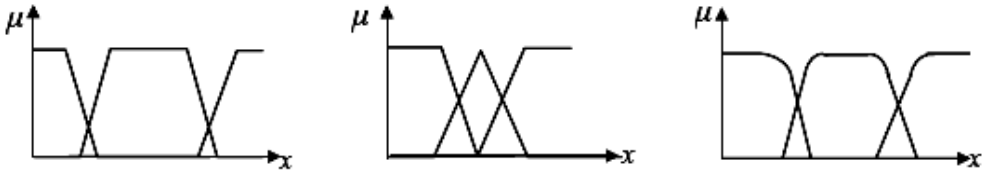


Fig A. 1 – Différentes formes de fonctions d'appartenance.

La Logique floue est basée sur des variables floues dites *variables linguistiques* à valeurs linguistiques dans l'univers du discours U. Chaque valeur linguistique constitue alors un ensemble flou de l'univers du discours. La figure A.2 illustre une partition floue de la variable linguistique « température ».

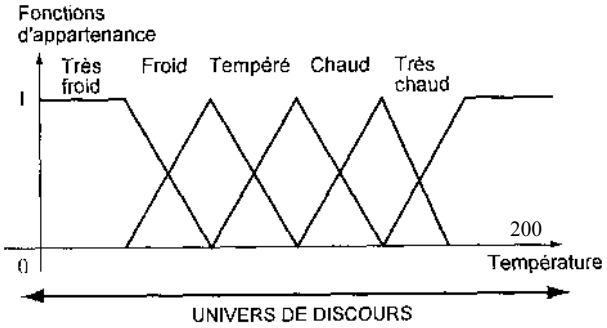


Fig A. 2 – Partition floue de la variable linguistique « température ».

où :

Univers du discours : Gamme de température de 0°C à 200°C.

Variable linguistique : La température.

Valeurs linguistiques : « Très froid» « Froid » « Tempéré » « Chaud » « Très Chaud ».

Les relations entre les opérateurs logiques et les opérateurs ensemblistes sont redéfinies pour la théorie des ensembles flous. Parmi les opérateurs les plus connus, on peut citer:

Les normes triangulaires (ou t-normes). Il s'agit d'opérateurs de conjonctions («et» ou Λ), les plus connus sont le minimum et le produit.

Les co-normes triangulaires (ou t-conormes). Ce sont des opérateurs de disjonctions («ou» ou $\vee$), la plus connue des t-conormes est le maximum.

Les connectifs mixtes. Il s'agit de combinaisons linéaires ou non linéaires de t-normes et de t-conormes.

La négation utilise la plupart du temps l'opérateur de complémentation.

En particulier, l'union, l'intersection et le complément sont définis par :

Union : $\mu_{A\cup B}(x) = \mu_A(x) \vee \mu_B(x) = \max[\mu_A(x), \mu_B(x)]$

Intersection : $\mu_{A\cap B}(x) = \mu_A(x) \wedge \mu_B(x) = \min[\mu_A(x), \mu_B(x)]$

Complément : $\neg\mu_A(x) = 1 - \mu_A(x)$

A.2 Propositions et règles floues

Les propositions floues sont caractérisées par un degré de vérité compris entre 0 et 1. Par exemple «X est A » constitue une proposition floue.

Une règle floue est définie comme une proposition floue correspondant à la mise en relation de deux propositions floues par une implication : « **Si** X est A **alors** Y est B ». L'implication utilisée pour relier le prémisse et la conclusion d'une règle est une relation floue qui peut être définie de différentes manières suivant le contexte.

A.3 Fonctionnement d'un système flou

Un système flou est composé de trois parties : fuzzification, inférences et défuzzification.

- **Fuzzification**

L'étape de fuzzification à pour but de transformer une donnée numérique en variable linguistique en calculant les degrés d'appartenance de la donnée aux sous-ensembles flous caractéristiques d'une variable linguistique. Il s'agit alors de la conversion des données en entrée en degré d'appartenance par l'intermédiaire de fonctions d'appartenance.

- **Inférences floues**

Dans cette partie, on doit décrire toutes les règles qui contrôlent le système. Une règle doit être sous la forme : Si condition, alors conclusion.

Une fois que l'on a établi une liste de règles d'inférences, il suffit qu'appliquer chaque règle aux variables linguistiques calculés dans l'étape de fuzzification.

Les résultats de ces règles pourront directement aller à l'étape finale de défuzzification.

- **défuzzification**

La dernière étape pour avoir un système flou opérationnel s'appelle la défuzzification. Lors de la seconde étape, on a généré un tas de commandes sous la forme de variables linguistiques (une commande par règle). Le but de la défuzzification est de fusionner ces commandes et de transformer les paramètres résultants en donnée numérique.

Il existe plusieurs méthodes pour défuzzifier. Parmi les plus utilisés, on peut citer la méthode du centre de gravité. Elle consiste à prendre l'abscisse correspondant au centre de gravité de la fonction d'appartenance. Formellement, on l'exprime comme :

$$sortie = \frac{\int_U x\mu(x)dx}{\int_U \mu(x)dx}$$ où U est l'univers du discours.

A.4 Exemple d'application

L'exemple est destiné à illustrer les démarches de contrôle du déplacement d'un robot le long d'un mur en utilisant des règles floues. L'objectif est de garder le robot à 10 cm du mur lors de son déplacement voir figure A.3.

Les règles de commande distance (d) → angle d'orientation (α) sont :

Règle 1 – Si la distance est petite, tourner à gauche (angle négatif).

Règle 2 – Si la distance est autour de 10 cm, garder la direction actuelle.

Règle 3 – Si la distance est grande, tourner à droite (angle positif).

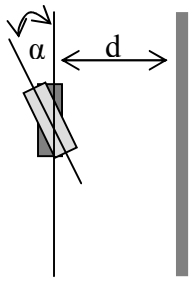


Fig A. 3 – Déplacement d'un robot le long d'un mur.

- **Fuzzification**

Cette étape consiste à partitionner l'ensemble de définition des attributs numériques quantifiant les variables linguistiques d'entrée et de sortie. La figure A.4 donne la partition des univers du discours de la distance *d* associée à la variable linguistique « distance » et de l'angle α associée à la variable linguistique « angle ». Les deux partitions utilisent des fonctions triangulaires.

La figure A.4-a montre la fuzzification de la distance $d = 6$ cm qui se traduit en logique floue par :

d appartient à l'ensemble flou petit avec un degré d'appartenance de 80%.

d appartient à l'ensemble flou moyen avec un degré d'appartenance de 20 %.

d appartient à l'ensemble flou grand avec un degré d'appartenance de 0 %.

- **Implication**

L'implication floue utilisée est de type Mamdani. L'ensemble flou de conclusion est construit en réalisant le minimum entre le degré d'activation et la fonction d'appartenance, sorte d'écrêtage de la fonction d'appartenance de conclusion (figure A.4-c).

- **Agrégation : composition de règles**

L'ensemble flou global de sortie est construit par agrégation des ensembles flous obtenus par chacune des règles. On considère que les règles sont liées par un 'OU' logique, et on calcule donc le maximum entre les fonctions d'appartenance résultantes pour chaque règle, voir figure A.4-d.

- **Défuzzification**

A ce stade, on a la fonction d'appartenance d'un ensemble flou qui caractérise le résultat.

Il faut défuzzifier, c'est à dire : Associer à cette ensemble flou un nombre interprétable par l'utilisateur. La valeur de l'angle d'orientation calculée à l'aide de la défuzzification par centre de gravité vaut donc ***-13.7*** comme illustrée sur la figure A.4-d, ce qui est une valeur plus réaliste.

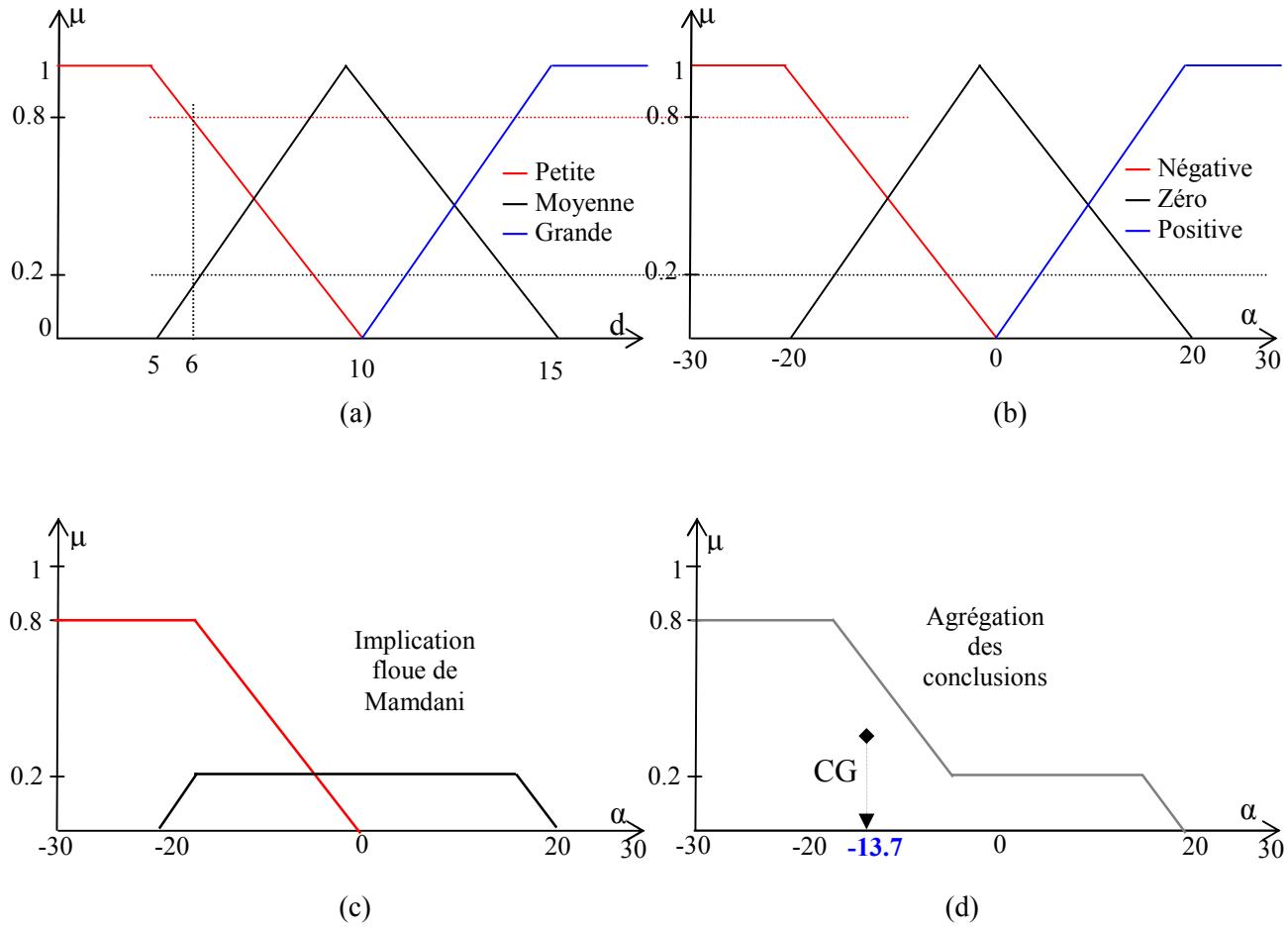


Fig A. 4 - Exemple de moteur d'inférence flou. (a) partition des univers du discours de la distance d. (b) partition des univers du discours de l'angle α. (c) résultat de l'implication floue de Mamdani définie par l'opérateur MIN. (d) Résultat d'Agrégation des conclusions définie par l'opérateur MAX et défuzzification.

B. Réseau hybride neuro-flou

Le système neuro-floue est un modèle qui réalise la logique floue avec les outils du réseau de neurones. Il combine les réseaux de neurones avec les systèmes d'inférence floue. Ces systèmes de classification permettent ainsi d'utiliser aussi bien les capacités d'apprentissage et de généralisation des réseaux de neurones, que les capacités de la logique floue de prendre en compte un environnement incertain et évolutif donc une bonne représentation de connaissance. L'une des premières méthodes neuro-floues est le codage du système d'inférence floue sous la forme d'un réseau de neurones multicouches dans lequel les poids correspondent aux paramètres du système. L'architecture du réseau dépend du type de règles et des méthodes d'inférence, d'agrégation et de défuzzification choisies.
Un système d'inférence flou est un système flou comprenant plusieurs règles :

R_k : Si $(x_1 \text{ es } A_1^k, x_2 \text{ est } A_2^k, \ldots\ldots, x_n \text{ est } A_n^k)$ Alors $(y_1 \text{ est } v_{k1}, y_2 \text{ est } v_{k2}, \ldots\ldots, y_m \text{ est } v_{km})$.

Où R_k est la $k^{ème}$ règle ($1 \leq k \leq H$) d'une base de H règles floues.
$\{x_i\}_{i=1:n}$ sont les variables d'entrée (antécédent), $\{y_j\}_{j=1:m}$ sont les variables de sortie (conclusion/conséquent), A_i^k sont les ensembles flous définis sur les variables d'entrée, et v_{kj} sont les singletons flous définis sur les variables de sortie.

B.1 Architecture du réseau neuro-flou

L'architecture du réseau de neurones proposé réalise le mécanisme d'inférence du modèle flou de Takagi-Sugeno d'ordre zéro, basé sur une collection de H règles floues. En utilisant des fonctions d'appartenance gaussiennes, on peut faire le rapprochement entre le classifieur de Sugeno et les réseaux de fonctions à base radiale (RBF). Le réseau RBF utilisé est constitue de trois couches (figure B.1).

1. La couche d'entrée : les neurones de cette couche reçoivent les valeurs d'entrées (x_1, $x_2, \ldots, x_n$) et agissent comme des ensembles flous A_i^k. Chaque neurone calcule le degré d'appartenance de x_i à l'ensemble flou A_i^k ; la fonction d'appartenance utilisée est la fonction gaussienne :

$$f_{ik}^{(1)}(x_i) = e^{-(x_i - w_{ik})^2 / \sigma_{ik}^2} \qquad \text{(B.1)}$$

où $\{w_{ik}\}$ et $\{\sigma_{ik}\}$ sont respectivement les centres et les écarts type des gaussiennes.

Les neurones dans cette couche sont arrangés en H groupes ; chaque groupe représente la partie ***si*** (***if***) d'une règle floue.

2. La couche cachée : le nombre de neurones dans cette couche est égal au nombre de règles floues où chaque neurone représente une règle floue ; le k^{ème} neurone caractérise l'opérateur flou ET, qui est réalisé ici par un simple produit :

$$f_k^{(2)}(x) = \prod_{i=1}^{n} f_{ik}^{(1)}(x_i) \qquad \text{(B.2)}$$

3. la couche de sortie : les neurones dans cette couche sont les variables de sortie du système. Chaque neurone *j* agit comme defuzzifier, évalue les conclusions de toutes les règles et donne la réponse du système à une entrée donnée.

$$y_j = f_j^{(3)}(x) = \frac{\sum_{k=1}^{H} f_k^{(2)}(x).v_{kj}}{\sum_{k=1}^{H} f_k^{(2)}(x)} \qquad \text{(B.3)}$$

Où v_{kj} sont les poids reliant la couche cachée à celle de la sortie.

Cette équation montre comment les différents règles participent à la reconnaissance de toutes les classes : plus une règle est activée, plus il participe au score global de la classe.

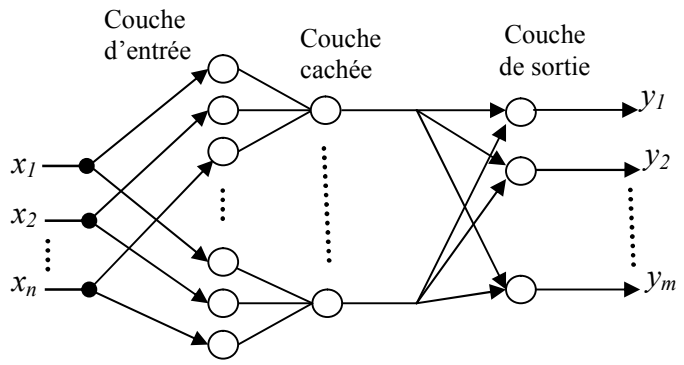


Fig B. 1 – Architecture d'un réseau neuo-flou.

B. 2 Apprentissage du réseau neuro-flou

L'apprentissage du réseau neuo-flou consiste à régler quatre paramètres principaux: le nombre de neurones dans l'unique couche cachée ou le nombre des gaussiennes, la position des

centres de ces gaussiennes, la largeur de ces gaussiennes et les poids de connexions entre les neurones cachés et le(s) neurone(s) de sortie.

Le réseau RBF consiste à minimiser l'erreur quadratique totale calculée entre les sorties obtenues du réseau et celles désirées.

La règle d'ajustement des poids est définie comme suit :

1- Calculer les signaux d"erreurs pour les neurones des trois couches $j \in L_3$, $k \in L_2$ et $ik \in L_1$:

$$\delta_j^{(3)} = -\frac{\partial E}{\partial f_j^{(3)}} = y_j - f_j^{(3)} \tag{B.4}$$

$$\delta_k^{(2)} = -\frac{\partial E}{\partial f_k^{(2)}} = \frac{\sum_{j=1}^{m} \delta_j^{(3)} (v_{kj} - f_j^{(3)})}{\sum_{t=1}^{H} f_t^{(2)}} \tag{B.5}$$

$$\delta_{ik}^{(1)} = -\frac{\partial E}{\partial f_{ik}^{(1)}} = \delta_k^{(2)} . \frac{\partial f_k^{(2)}}{\partial f_{ik}^{(1)}} \tag{B.6}$$

2- Ajuster les poids $\{v_{kj}\}, \{w_{ik}\}$ et $\{\sigma_{ik}\}$ respectivement suivant les quantités d'ajustement suivantes:

$$\Delta v_{kj} = \eta \delta_j^{(3)} . \frac{f_k^{(2)}}{\sum_{t=1}^{K} f_t^{(2)}} \tag{B.7}$$

$$\Delta w_{ik} = \eta \delta_{ik}^{(1)} \left[\frac{2(x_i - w_{ik})}{\sigma_{ik}^2} \right] f_{ik}^{(1)} \tag{B.8}$$

$$\Delta \sigma_{ik} = \eta \delta_{ik}^{(1)} \left[\frac{2(x_i - w_{ik})^2}{\sigma_{ik}^3} \right] f_{ik}^{(1)} \tag{B.9}$$

Avec : $w_{kj} = w_{kj} + \Delta w_{kj}$, $\sigma_{kj} = \sigma_{kj} + \Delta \sigma_{kj}$, $\sigma_{kj} = \sigma_{kj} + \Delta \sigma_{kj}$ où η est le pas d'apprentissage.

De cette manière, l'algorithme d'apprentissage peut être employé pour déterminer les paramètres des systèmes flous. Ceci revient à créer ou améliorer un système flou de manière automatique, au moyen des méthodes spécifiques aux réseaux neuronaux. Un aspect important est que le système reste toujours interprétable en termes de règles floues.

C. Evaluation de la vraisemblance d'une observation/MMC

La probabilité d'observation de *O* sachant un modèle λ est la somme sur tous les chemins d'états *E* des probabilités conjointes de *O* et de *E* par rapport à ce modèle [27].

$$p(O/\lambda) = \sum_{x_1,x_2,\ldots,x_T} p(O,X/\lambda) = \sum_{x_1,x_2,\ldots,x_T} p(O/X,\lambda).p(X/\lambda) \qquad \text{(C.1)}$$

où $X=\{x_1,x_2,\ldots,x_T\}$ désigne une séquence d'états quelconque.

Selon l'hypothèse d'indépendance (1.39), on a :

$$p(O/X,\lambda) = \prod_{t=1}^{T} p(o_t/x_t,\lambda) = \prod_{t=1}^{T} b_{x_t}(o_t) \qquad \text{(C.2)}$$

De plus notre MMC est supposé du premier ordre.

$$p(X/\lambda) = \pi_{x_1}.a_{x_1x_2}.a_{x_2x_3}\ldots a_{x_{T-1}x_T} \qquad \text{(C.3)}$$

Dès lors, (C.1) se calcule selon :

$$p(O/\lambda) = \sum_{x1,x2,\ldots,xT} \pi_{x_1} b_{x_1}(o_1).a_{x_1x_2}.b_{x2}(o_2)\ldots a_{x_{T-1}x_T}.b_{x_T}(o_T) \qquad \text{(C.4)}$$

Cette méthode s'avère cependant très coûteux en temps de calcul : de l'ordre de $2T.N^T$ multiplications sont requises, car il y a N^T séquence d'états possibles (figure C.1).

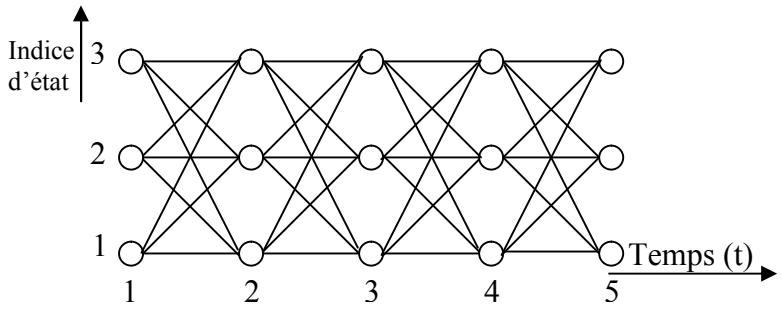


Fig C. 1 – Treillis d'états complètement connectés.

De ce fait, le calcul devient irréalisable pour de longues séquences d'observations.

Une évaluation optimale de cette probabilité est obtenue par les fonctions forward-backward [28].

C.1 Variable "forward"

Soit $\alpha_t(i) = p(o_1, o_2, ..., o_t, x_t = e_i / \lambda)$ la probabilité de générer l'observation partielle (o_1,.., o_t) sachant que le modèle λ se trouve dans l'état e_i à l'instant t; $\alpha_t(i)$ est appelée la variable "forward" ou "avant" et se calcule itérativement.

Cette variable se calcule de manière inductive, en trois étapes.

1. Initialisation : $\alpha_1(j) = \pi_j b_j(o_1)$, $1 \le j \le N$

C'est la probabilité conjointe de l'état initial e_j et de l'observation o_1.

2. Récursion (Induction) : $\alpha_t(j) = \left[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(o_t)$, $1 \le j \le N$, t=2, 3,….,T

La figure C.2 montre l'induction des variables directes.

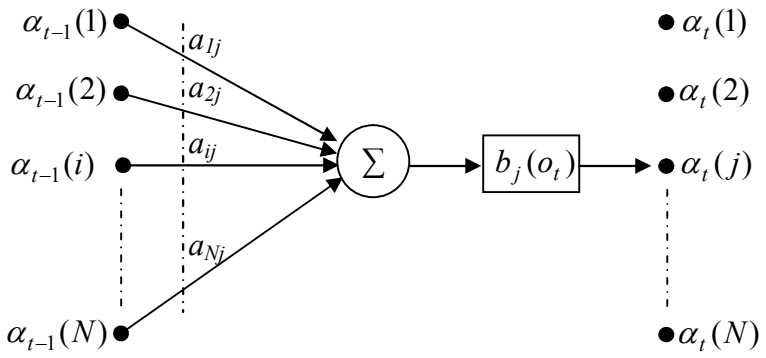


Fig C. 2 – Opération de base du calcul direct.

3. Terminaison : $p(O / \lambda) = \sum_{i=1}^{N} \alpha_T(i)$

On remarque que cet algorithme exploite le fait que, ce sont les N mêmes variables $\alpha_{t-1}(i)$ de l'étape précédente, calculées une fois pour toutes, qui sont recyclées dans le calcul des $\alpha_t(i)$ et quelle que soit la taille de la séquence d'observations, il n'y a, à chaque instant, que N états possibles.

Si l'on considère la formule d'induction, on se rend compte que l'algorithme ne nécessite que de l'ordre de N^2.T multiplications, ce qui permet une implémentation plus rapide que le calcul direct mentionné plus haut.

C.2 Variable " backward "

De la même façon l'algorithme introduit une variable, dite variable "backward", "arrière" ou "inverse" $\beta_t(i) = p(o_{t+1}, o_{t+2}, ..., o_T, x_t = e_i / \lambda)$ qui représente la probabilité d'observer la séquence partielle (o_{t+1}, . . . , o_T) sachant qu'on est à l'état e_i à l'instant t, pour le modèle λ.

Les variables inverses peuvent également se calculer de manière inductive, en trois étapes.

1. Initialisation : $\beta_T(i) = 1,, \quad 1 \le i \le N$

2. Récursion (Induction): $\beta_t(i) = \sum_{j=1}^{N} a_{ij} . \beta_{t+1}(j) . b_j(o_{t+1})$, $1 \le j \le N$, t=T-1, T-2,….3, 2, 1

Le détail de calcul de cette variable est illustré dans la figure C.3.

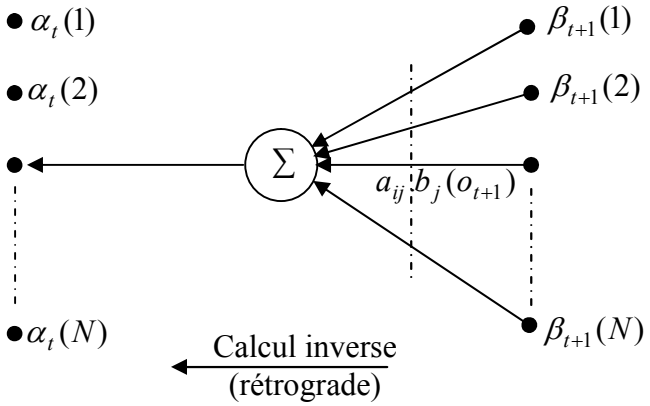


Fig C. 3 – Induction des variables inverses.

3. Terminaison : $p(O / \lambda) = \sum_{i=1}^{N} \pi_i . b_i(o_1) . \beta_1(i)$

D. Machines à vecteurs de supports (MVS)

Nous présentons les machines à vecteurs de support qui sont utilisées pour la classification supervisée. Ces machines sont basées sur le calcul de fonctions noyau et ont été très populaires pour des applications de régression et de classification.

Nous allons ici montrer la dérivation du SVM binaire de sa forme primale à sa forme duale, cette dernière étant un problème quadratique sous contraintes de boîtes.

Pour un problème de discrimination à deux classes, on cherche une frontière de décision entre les exemples de chaque classe. Ainsi, étant donné un ensemble de données d'apprentissage étiquetées $\{(x_k, y_k) : k = 1,\ldots,n\}$, où $x_k \in R^d$ et $y_k \in \{1,-1\}$. Pour simplifier les expressions par la suite, les deux classes sont donc étiquetées -1 et 1.

Pour le cas d'un problème linéairement séparable, il est plus courant d'utiliser le critère de la maximisation de la marge (figure D.1).

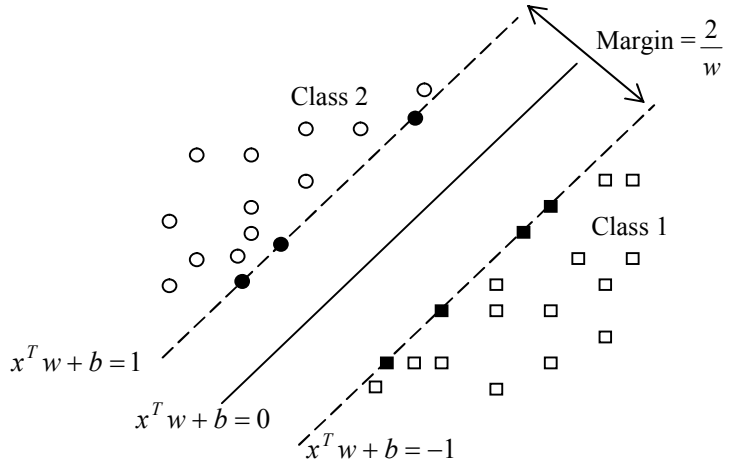


Fig D. 1 – Critère de la maximisation de la marge.

On suppose que toutes les données d'apprentissage satisfassent les contraintes suivantes :

$$x_i^T w + b \geq +1 \quad si \quad y_i = +1 \tag{D.1}$$

$$x_i^T w + b \leq -1 \quad si \quad y_i = -1. \tag{D.2}$$

Alors, l'hyperplan $(x^T w + b)$ sépare les données si et seulement si:

$$y_i (x_i^T w + b) \geq 1, \quad \forall\, i. \tag{D.3}$$

L'hyperplan séparateur optimal est un classifieur basé sur un critère de la maximisation de la marge dont la sortie est donnée par :

$$f(x) = sign(x^T w + b) \tag{D.4}$$

Où *x* est la donnée d'entrée, *w* est le vecteur de poids et *b* est le biais. Le biais et les poids sont calculés par maximisation de la marge $1/\|w\|$. Comme on cherche également un vecteur w de norme minimum, l'hyperplan cherché est la solution du problème de minimisation suivant :

$$\begin{cases} \min \frac{1}{2}\|w\|^2 \\ avec \quad y_i(x_i^T w + b) \geq 1, \quad i = 1,....., N. \end{cases} \tag{D.5}$$

Où $y_i \in \{-1, 1\}$ représente l'étiquette de la forme d'apprentissage *xi*.

La résolution d'un tel problème s'effectue à l'aide des multiplicateurs de Lagrange, on affecte à chaque contrainte le coefficient α_i, il s'agit alors de minimiser l'expression :

$$Lp = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i(x_i^T w + b) - 1 \right] \tag{D.6}$$

Où les α_i sont des multiplicateurs de Lagrange. En annulant les dérivées partielles de *Lp* par rapport aux paramètres *w* et *b*, ce problème conduit à la maximisation de la forme duale du problème par rapport aux α_i :

$$\begin{cases} L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j}^{N} \alpha_i \alpha_j y_i y_j \langle x_i^T, x_j \rangle \\ s.t. \ \ \alpha_i \geq 0, \ \ i = 1,.....N, \\ et \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases} \tag{D.7}$$

Ce problème est un problème de minimisation quadratique sous contraintes [168] pour lequel plusieurs algorithmes d'optimisation ont été proposés.

La solution *w* est une combinaison linéaire des exemples x_i :

$$w = \sum_{i=1}^{N} \alpha_i y_i x_i \tag{D.8}$$

Chaque point x_i de l'ensemble d'apprentissage est associé à un multiplicateur $\alpha_i \geq 0$. Seuls les points tel que $\alpha_i \neq 0$ intervient dans la solution, ce sont les ***vecteurs de support***.

La fonction f s'écrit alors :

$$f(x) = sign(\sum_{i \in SV} y_i \alpha_i (x_i^T x) + b) \tag{D.9}$$

où SV est l'ensemble des indices des vecteurs de support (vecteurs pour lesquels les coefficients de Lagrange sont non nuls).

Les SVs se trouvant sur les bords de la marge sont représentés par des cercles et carrés noirs (figure D.1).

Lorsque les données ne sont pas linéairement séparables, il faut relâcher les contraintes en introduisant des variables d'écart $\xi_i \geq 0$

$$\begin{aligned} &\min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} \xi_i \\ &s.t.\ \ y_i(x_i^T w + b) \geq 1 - \xi_i, \quad i = 1,.....,N. \end{aligned} \tag{D.10}$$

Les écarts permettent à certains points de se situer du mauvais coté de la frontière. Il faut alors minimiser $\sum_{i=1}^{N} \xi_i$, et on montre que l'on obtient alors la même solution que précédemment, avec une contrainte supplémentaire :

$$0 \leq \alpha_i \leq C, \quad i = 1,.....N, \tag{D.11}$$

Où C est une constante positive qui permet de doser l'importance de que l'on accorde a priori aux écarts.

Il est possible d'étendre les SVM au cas non linéaire d'après [169] en remplaçant le produit scalaire <*xi,x*> par une fonction noyau symétrique $K(x_i,x)$ et la fonction f s'écrit alors :

$$f(x) = sign(\sum_{i \in SV} y_i \alpha_i K(x_i, x) + b) \tag{D.12}$$

Où $K(x, x_i) = <\phi(x), \phi(x_i)>$ avec $<.,.>$ est l'opérateur produit scalaire et $\phi(x)$ est une projection de x dans un espace de grande dimension. Cette projection peut permettre de rendre séparables des données initialement non linéairement séparables, voir figure D.2.

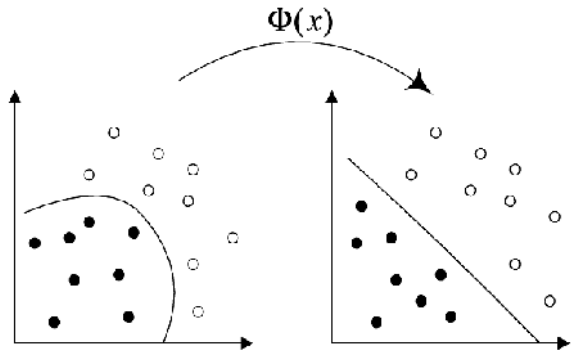


Fig D. 2 – Projection dans un espace de grande dimension.

La fonction de projection $\phi(x)$ n'est pas obligatoirement explicite. Les fonctions noyau typiques sont :

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

Noyau Sigmoïde : $K(x_i, x) = \tanh(\gamma(x^T x_i) + c)$.

Noyau Polynomial : $K(x_i, x) = (\gamma\, x^T x_i + c)^d$.

Bibliographie

[1] R. O. Duda, P. E. Hart, & D. G. Stork. Pattern Classification. (Second edition). New York: Wiley-Interscience, 2001.

[2] J. Milgram, R. Sabourin, M. Cheriet. Système de classification à deux niveaux de décision combinant approche par modélisation et machines à vecteurs de support. Traitement du Signal, Vol. 22, N°3, Traitement automatique des documents pp. 293-304, 2005.

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

[4] C. K. Chow. Statical Independence and Threshold Functions. IEEE Transactions on Electrical Computer, 14:66-68, 1965.

[5] L .Lebart, A. Morineau, J.-P. Fénelon. Traitement des données statistiques. Ed. Dunod, Paris, 1982.

[6] Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, Angela Y. Wu. An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions. JACM 45(6), pp 891-923, 1994.

[7] J.C. Bezdek, "Pattern Recognition with Fuzzy Objective Function Algorithms," Plenum Press, New York, 1981.

[8] B. Widrow, M. Lehr. 30 years of adaptive neural networks: Perceptron, madaline and back-propagation. IEEE., Proc., Vol. 87, N°9, September 1990.

[9] H. Abdi. Les réseaux de neurones. Press Universitaires de Grenoble, 1994.

[10] E. Davala, P. Main. Des réseaux de neurones, Editions Eyrolles, 1993.

[11] H. Borland, C. J. Mellekms, Multi-layer perceptrons and automatic speech recognition. IEEE, First Annual International Conference on Neural Networks, San Diago, California, 21-24 June 1987.

[12] CG. Looney. Advances in feedforward neural networks: Demystifying knowledge acquiring black boxes. IEEE Trans. Knowl. Data Eng. 8 (2):211-226, 1996.

[13] J. S. Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. in Neuro-computing: Algorithms, Architectures and Applications, Eds: Fougelman-Soulie and Hérault, Springer Verlag, pp. 227-236, 1989.

[14] H. Bourlard, N. Morgan. Continuous Speech Recognition by Connectionist Statistical Methods," IEEE trans. on Neural Networks, 4(6), pp. 893-909, 1993.
[15] D. R. Hush, B. G. Horne. Progress in supervised neural networks. IEEE signal processing magazine, pp 8-39, Janvier 1993.
[16] B. Gosselin. Application de réseaux de neurones artificiels a la reconnaissance automatique de caractères manuscrits. Thèse de Doctorat, Faculté Polytechnique de Mons, juin 1996.
[17] D.S. Broomhead, D. Lowe. Multi-variable functional interpolation and adaptive networks. Complex Sysytem, Vol.2, pp.269-303, 1988.
[18] S. Furui. Digital speech processing, synthesis, and recognition. Marcel Deller Inc., New York, 1990.
[19] R. E. Bellman. Dynamic Programming. Princeton University Press, 1957.
[20] J. S. Bridle. Optimization and search in speech and language processing. Survey of the state of the art in human language technology, NSF, CE-DG13E & OGI-CSLU éds, art. 11.7, pp 423-428, 1995.
[21] L.R. Rabiner, A.E. Rosenberg, S.E. Levinson. Considerations In Dynamic Time Warping Algorithm for Dicrete Word Recognition, IEEE Trans, ASSP, vol.ASSP-26, pp.575-582, 1978.
[22] R.A. Wagner et M.J. Fisher, "The String-to-String Correction Problem", Journal of the ACM, vol. 21, no. 1, pp. 260-265, 1974.
[23] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of IEEE 77 (2) p. 257-286, 1989.
[24] N. Ben Amara, A. Belaïd, N. Ellouze. Modélisation pseudo bidimensionnelle pour la reconnaissance de chaînes de caractères arabes imprimés. Proc. 1er Colloque international francophone sur l'écrit et le document (CIFED'98), Québec, Canada, p. 131-140, 1998.
[25] A.W, Senior, A.J, Robinson. An off-line cursive handwriting recognition system. IEEE Trans. Pattern Anal. Mach. Intelli; 20 (3):309-321, 1998.
[26] H. Bourlard, N. Morgan. Connectionist speech recognition – a hybrid approach. Kluewer Academic Publishers, 1994.
[27] A. Kriouile, J. F. Mari, J. P. Haton . Some Improvements in Speech Recognition Algorithms Based on HMM. In IEEE-ICASSP, p. 545-548, 1990.
[28] L. E. Baum, J. A. Egon. An Inequality with Applications to Statistical Estimation for Probabilistic Functions of a Markov Process and to a Model for Ecology. Bull, Ants . 73 : 360-363, 1967 .

[29] L. Rabiner et B. H. Juang. Fundamentals of Speech Recognition. Prentice Hall, Englewood Cliffs, New Jersey, 1993.
[30] A. P. Dempster, N. M. Laird, D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). Journal of the Royal Statistical Society, volume B 39, pages 1–38, 1977.
[31] P.A. Devijver, M . Dekesel. Champs aléatoires de Pickard et modélisation d'images digitales. Traitement du Signal, 5(5) : 131-150, 1988.
[32] M.Y. Chen, A. Kundu, S.N. Srihari. Variable duration hidden Markov and morphological segmentation for handwritten word recognition. IEEE Trans. Image Process; 4 (12):1675-1688, 1995.
[33] G. D. Forney. The Viterbi Algorithm . Proceedings of the IEEE, 661(3), 1973.
[34] R. Bakis. Continuous speech recognition via centisecond acoustic states. In 91st Meeting of the Acoustical Society of America, 1976.
[35] H. Bourlard, CJ. Wellekens. Links between Markov models and multilayer perceptrons. Proceedings of IEEE Conference on Neural Information Processing Systems, Denver, CO, Touretzky, D. (ed.), Morgan-Kaufmann, p. 502-510, 1989.
[36] J.B. Hampshire, H. Pearlmutter. Equivalence proofs for multi-layer perceptron classifiers and the Bayesian discriminant function. Proceedings of In Connectionist Models: Proceedings of the Summer School, Morgan-Kauffmann, p. 159-172, 1990.
[37] N. Morgan, H. Bourlard. Continuous Speech Recognition Using Multilayer Perceptrons with Hidden Markov Models, Proceedings of ICASSP-90, p.413-416, 1990.
[38] C.L. Liu, K. Nakashima, H. Sako, H. Fujisawa. Handwritten digit recognition: benchmarking of state-of-the-art techniques. Pattern Recognition, 36 (10): 2271-2285, 2003.
[39] J.C. Platt. Probabilities for SV Machines. Advances in Large Margin Classifiers, MIT Press, p. 61-74, 1999.
[40] C.C. Chang, C.J. Lin. LIBSVM: a library for support vector machines. Technical rapport, Department of Computer Science and Information Engineering, National Taiwan University, 2001.
[41] E. Herwijnen. Practical SGML. Kluwer Academic Publisher, 1990.
[42] G. Nagy. Twenty years of document image analysis in PAMI. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22 (1): 38-62, 2000.
[43] F. Wahl, K. Wong, R. Casey. Block Segmentation and Text Extraction in Mixed Text/Image Documents. Computer Vision Graphics, and Image Processing, 20, 375-390, 1982.

[44] G. Nagy, J. Kanai, M. Krishnamoorthy, M. Thomas, M. Viswanathan. Two Complementary Techniques for Digitized Document Analysis. Proceedings ACM Conference on Document Processing Systems, Santa Fe, New Mexico, USA, December 1988.
[45] D. Wang, S. N. Srihari. Classification of newspaper image blocks using texture analysis. Computer Vision Graphics and Image Processing, 47 (3): 327-352, September 1989.
[46] J. Fisher, S. Hinds, K. D'Amato. A Rule-Based System for Document Image Segmentation. Proceedings of the 10th International Conference on Pattern Recognition, Atlantic City, USA, p. 113-122, 1990.
[47] T. Saitoh, T. Pavlidis. Page Segmentation without Rectangle Assumption. Proceedings of the 11th International Conference on Pattern Recognition, The Hague, USA, p. 277-280, 1992.
[48] T. Pavlidis, J. Zhou. Page segmentation and classification. Graphical models and image processing, 54: 484-496, 1992.
[49] R. Kasturi, L. O'Gorman. Document Image Analysis. IEEE Computer Society Press, ISBN 0-8186-7802-X, 1997.
[50] A.K. Jain, B. Yu. Document representation and its application to page decomposition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20: 294-308, 1998.
[51] S.W. Lee, L. Lam, C.Y. Suen. Performance Evaluation of Skeletonization Algorithms for document Image Processing. First Intern. Conf. on Document Analysis, ICDAR, Saint-Malo p. 260-271, , sept. 1991.
[52] T. Paquet, T. Vallee, Y. Lecourtier. Extraction de primitives par suivi de traits dans l'image binarisée d'un mot manuscrit. BIGRE, N°68, p. 179-188, Mai, 1990.
[53] A. Belaid, Y. Belaid. Reconnaissance des formes : méthodes et application. InterEdition 1992.
[54] A. Bagdanov, J. Kanai. Projection Profile Based Skew Estimation Algorithm for JBIG Compressed Images. In: Proceedings of the 4th International Conference on Document Analysis and Recognition, p. 401-405, 1997.
[55] E. Kavallieratou, N. Fakotakis, G. Kokkinakis. Skew angle estimation for printed and handwritten documents using the Wigner-Ville distribution. Image and Vision Computing, 813–824, 2002.
[56] S. J. Perantonis, B. Gatos, N. Papamarkos. Block decomposition and segmentation for fast Hough transform evaluation. Pattern Recognition, 811–824, 1999.
[57] A. Amin, R. Shiu. Page Segmentation and Classification Utilizing Bottom-Up Approach. International Journal of Image and Graphics, 1(2): 345-361, 2001.

[58] A. Jain, B. Yu. A Robust and Fast Skew Detection Algorithm for Generic Documents. Pattern Recognition, 29 (10):1599-1629, October 1996.

[59] P.Y. Yin. Skew Detection and Block Classification of Printed Documents. Image and Vision Computing, 567-579, 2001.

[60] L. O'Gorman. The document Spectrum for page layout analysis. IEEE Transactions on Page Layout Analysis and Machine Intelligent, 1162-1173, 1993.

[61] A. Antonacopoulos. Local Skew Angle Estimation from Background Space in Text Regions. Proceedings of the 4th International Conference on Document Analysis and Recognition, p. 684–688, August 18–20, 1997.

[62] H. K. Kwag, S. H. Kim, S. H. Jeong, G. S. Lee. Efficient skew estimation and correction algorithm for document images. Image and Vision Computing, 25-35, 2002.

[63] C. A. Glasbey. An Analysis of Histogram-Based Thresholding Algorithms. CVGIP-Graphical Models and Image Processing, 55(6), pp. 532–537, 1993

[64] S. A. Lee, S. Y. Chung, R. H. Park. A Comparative Performance Study of Several Global Thresholding Techniques for Segmentation. CVGIP, 52, pp. 171–190, 1990

[65] P. K. Sahoo, S. Soltani, A. K. C. Wong, , Y. C. Chen. Survey of Thresholding Techniques, CVGIP, 41(2), pp. 233–260, 1988

[66] N. Otsu. A Threshold Selection Method from Gray-Level Histograms. IEEE Trans. on SMC, 9(1): 62–66, 1979

[67] C.C. Tappert, C.Y. Suen T. Wakahara. The State of the Art in On-Line Handwriting Recognition. IEEE Trans. on PAMI, 12 (8), 787-808, 1990.

[68] A. Belaid. La reconnaissance automatique de l'écriture. Dossier Pour la Science, 33, octobre 2001.

[69] E. Lethelier, M. Leroux, M. Gilloux. Traitement des montants numériques de cheques postaux. Une méthode de segmentation basée sur la reconnaissance. Traitement du signal, 12 (6), 1995.

[70] C. Parisse. Global word shape processing in off-line recognition of handwriting. I.E.E.E., Trans. on An. and Mach. Int., 18 (4), April 1996.

[71] E. Anquetil, G. Lorette. Reconnaissance en ligne de caractères manuscrits basée sur une approche qualitative par la logique floue. In Actes du 4e Colloque National sur l'Écrit et le Document, (CNED'96), Nantes, France, p. 23-30, 1996.

[72] S. Bercu, B. Delyon, G. Lorette. Segmentation pour une méthode de reconnaissance d'écriture cursive en ligne. Bigre, N°80 – CNED 92 : Au Colloque National sur l'écrit et le document, Nancy, Juillet 1992.

[73] Yi Lu, "Machine-Printed Character Segmentation", Pattern Recognition, 28(1), 67-80, January 1995

[74] T. S. Elsheich, R. M. Guindi. Automatic recognition of isolated Arabic characters. Signal Processing, p. 177-184, 1988.

[75] A. N. Azockly, A. Zramdani, R. Ingold. Reconnaissance de la structure physique de documents composites, Bigre, N°80 – CNED 92 : Au Colloque National sur l'écrit et le document, Nancy, Juillet 1992.

[76] Y. H. Liu, H. N. Pham, B. Dubuissen. Reconnaissance de la structure logique d'un document scientifique, Bigre, N°80 – CNED 92 : Au Colloque National sur l'écrit et le document, Nancy, Juillet 1992.

[77] A. Vinciarelli. A survey on off-line cursive word recognition. Pattern Recognition, 35(7):1433–1446, juillet 2002.

[78] C. Parisse. Reconnaissance hors ligne de mots manuscrits sans contraintes d'écriture, ni de vocabulaire. Actes de CNED'92, p. 136-143, Nancy, juil. 1992.

[79] S. Madhvanath, V. Krpasundar. Pruning large lexicons using generalized word shape descriptors. In 4th International Conference on Document Analysis and Recognition, Ulm, Allemagne, page Poste, 1997.

[80] E. Cohen, J.J. Hull, S.N. Srihari. Control-structure for interpreting handwritten addresses. IEEE Trans. Pattern Analysis and Machine Intelligence, 16(10):1049–1055, octobre 1994.

[81] K.M. Sayre. Machine recognition of handwritten words: A project report. Pattern Recognition, 5(3):213–228, septembre 1973.

[82] R.G. Casey, E. Lecolinet. A survey of methods and strategies in character segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence, 18(7):690–706, juillet 1996.

[83] G.M. Reicher. Perceptual recognition as a function of meaningfulness of stimulus material. Journal of Experimental Psychology, 81:275–280, 1969.

[84] A. Vinciarelli. A survey on off-line cursive word recognition. Pattern Recognition, 35(7):1433–1446, juillet 2002.

[85] Oivid Due Trier, Anil K. Jain, & Torfinn Taxt. (1996). Feature extraction methods for character recognition – a survey. Pattern recognition, 29(4), 641-662.

[86] L. Heutte, T. Paquet, J.V. Moreau, Y. Lecourtier, C. Olivier. A structural/statistical feature based vector for handwritten character recognition. Pattern recognition letters, 19, 629-641, 1998.

[87] I. Guyon, I. Poujaud, L. Personnaz, G. Dreyfus, J. Denker, Y. Le Cun. Comparing different neural networks architectures for classifying handwritten digits2. Int. J. Conf. on Neural Networks, 2, 127-132. Washington, USA, 1989.

[88] H. C. Yau, J. M. Marry, Iterative improvement of a nearest neighbor classifier. Neural Networks, 4 : 517-524, 1991.

[89] S. Collin. Grammaire pour la reconnaissance de la cotation dans les dessins techniques. Bigre, N°80 – CNED 92 : Au Colloque National sur l'écrit et le document, Nancy, Juillet 1992.

[90] M. Gilloux, M. Leroux. Approche markoviennes en reconnaissance de l'écriture manuscrit, Bigre, N°80 – CNED 92 : Au Colloque National sur l'écrit et le document, Nancy, Juillet 1992.

[91] B. Lemarié, M. Gilloux, M. Leroux. Coopération neuro-markovienne pour la reconnaissance des montants littéraux des cheques, srtP/rd/ coopération neuro-markovienne, 2ème Semestre, Transitions Numéro 7, 1994

[92] G. Yahiaoui. Une approche pour la modélisation de chiffres manuscrites dédiée à l'apprentissage pour réseaux de neurones, Bigre, N°80 – CNED 92 : Au Colloque National sur l'écrit et le document , Nancy, Juillet 1992.

[93] M. J. Martin-Bautista, M. A. Vila. A survey of Genetic Feature Selection in Mining Issues. Evolutionary Computation, CEC, Proceedings of the 1999 Congress on, 2: 1314-1321, 1999.

[94] N. Chaikla, Y. Qi. Genetic Algorithms in Feature Selection. Systems, Man and Cybernetics, IEEE SMC Conference Proceedings, 5, 538-540, 1999.

[95] M.L. Raymer, W.F. Punch, E.D. Godman, L.A. Kuhn, A.K. Jain. Dimensionality Reduction Using Genetic Algorithms. Evolutionary Computation, IEEE Transaction, 4(2), 164-171, 2000.

[96] L. Lebart, A. Morineau, J.-P. Fénelon traitement des données statistiques Ed. Dunod, Paris, 1982

[97] R. O. Duda, P. B. Hart. Pattern classification and scene analysis. Wiley, New York, 1973.

[98] K. Fukunaga Introduction to Statistical Pattern Recognition Academic Press, 1990

[99] N. Essoukri Ben Amara, F. Bouslama. Classification of Arabic Script Using Multiple Sources of Information: State of the Art and Perspectives. Int. Journal on Document Analysis and Recognition, 5(4), 195-212, 2003.

[100] S. Gazzah, N. Essoukri, Ben Amara. Utilisation des Ondelettes en Caractérisation des Fontes Arabes. Int. Conf. On Image and Signal Processing, ICISP'2003, Maroc, June, 2003.

[101] C. Aloulou, Y. Bahou, L. Hadrich Belguith, A. Ben Hamadou. Adaptation et implémentation des grammaires hpsg pour l'analyse de textes arabes non voyellés. In Actes du 15e Congrès de Reconnaissance des Formes et Intelligence Artificielle (RFIA'2006), Tours, January 2006.

[102] A. Amin, H.B. Al-Sadoun, S. Fisher. HandPrinted Arabic Character Recognition system using an artificial network. Pattern Recognition, 29 (4) : 663-675, 1996.

[103] B. Al-Badr, S.A. Mahmoud. Survey and bibliography of Arabic Optical Text Recognition, Signal processing, 41: 49-77, 1995.

[104] M. S. Khorsheed. Off-line Arabic character recognition– a review, Pattern Anal, Appl. 5 : 31–45, 2002.

[105] A. Amin. Off line Arabic character recognition - a survey. In ICDAR '97. Proceedings of the 4th International Conference on Document Analysis and Recognition, pages 596–599. IEEE Computer Society, 1997.

[106] L. M. Lorigo, V. Govindaraju. Offline arabic handwriting recognition: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(5) :712–724, 2006.

[107] A. Belaïd, C. Choisy. Human reading based strategies for off-line arabic word recognition. Summit on Arabic and Chinese Handwriting Recognition 2006 - SACH'06, 2006.

[108] Pavithra Babu Sargur Srihari, Harish Srinivasan, Chetan Bhole. Handwritten arabic word spotting using the cedarabic document analysis system. In 2005 Symposium on Document Image Understanding Technology, The Marriott Inn and Conference Center, University Maryland University College, Adelphi, Maryland, November 2-4, 2005.

[109] T. Sari, L. Souici, M. Sellami. Off-line handwritten arabic character segmentation algorithm: Acsa. In IWFHR '02: Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR'02), page 452. IEEE Computer Society, 2002.

[110] M. Cheriet. Strategies for visual arabic handwriting recognition: issues and case study. In ISSPA 2007, International Symposium on Signal Processing and its Applications, 12 - 15 February 2007, Sharjah, United Arab Emirates, Feb 2007.

[111] M. Côté, E. Lecolinet, M. Cheriet, C. Y. Suen. Automatic reading of cursive scripts using a reading model and perceptual concepts: The PERPECTO system, International Journal on Document Analysis and Recognition, 1(1) : 3–17, 1998.

[112] M. T. Laskri, A. Sehad, L. Mezai, M. Cheriet. Détection de l'inclinaison des documents arabes imprimés. In 8ème colloque international francophone sur l'écrit et le document (CIFED'2004), June 2004.

[113] P. Burrow. Arabic handwriting recognition. M.Sc. Thesis, University of Edinburgh England, 2004.

[114] M. Pechwitz, V. Märgner. Baseline estimation for arabic handwritten words. In IWFHR '02: Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR'02), page 479. IEEE Computer Society, 2002.

[115] F. Farooq, V. Govindaraju, M. Perrone. Preprocessing methods for handwritten Arabic documents. Proceedings of the 2005 Eight International Conference on Document Analysis and Recognition (ICDAR'05) IEEE.1, p. 267-271, 2005.

[116] S. Masmoudi, H. Amiri. Reconnaissance de mots arabes manuscrits par modélisation markovienne. In Proceedings of 2ème Colloque International Francophone sur l'Ecrit et le Document (CIFED'2000), Lyon (France), July 3 – 5, 2000.

[117] Badr Al-Badr, Sabri A. Mahmoud. Survey and bibliography of Arabic optical text recognition. Signal Processing 41 : 49-77, 1995.

[118] L. Zheng, A. H. Hassin, X. Tang. A new algorithm for machine printed Arabic character segmentation. Pattern Recognition Letters 25 :1723–1729, 2004.

[119] A. Ameur. Reconnaissance de l'écriture arabe. Thèse de doctorat, Rouen la 3i, In French, 1992.

[120] A. Bennasri, A. Zahour, B. Taconet. Extraction des lignes d'un texte manuscrit Arabe, vision Interface '99, Trois-Rivières, Canada, 19-21 May, 1999.

[121] A. Boukharouba, A. Bennia. Recognition of Handwritten Arabic words using a neuro-fuzzy network. 1st Mediterranean conference on intelligent systems and automation CISA'08. Proceeding in American Institute of Physics (AIP). 1019, p. 254-259, June 12, 2008.

[122] K. Romeo, A. Ameur, C. Olivier, Y. Lecourtier. Structural analysis of Arabic Handwriting: segmentation and recognition. Machine Vision and Application, 8 : 232-240, 1995

[123] Pavithra Babu Sargur Srihari, Harish Srinivasan and Chetan Bhole. Spotting words in handwritten arabic documents. In In Procs. of SPIE, San Jose, CA, USA, Jan 2006.

[124] H. Miled, M. Cheriet, C. Olivier. Multi-level Arabic handwritten words recognition. In SSPR '98/SPR '98: Proceedings of the Joint IAPR International Workshops on Advances in Pattern Recognition, pages 944–951, London, UK, 1998. Springer-Verlag.

[125] H. Miled, C. Olivier, M. Cheriet. Modélisation de la notion de pseudo-mots en reconnaissance de mots manuscrits arabes. In CIFED, 2000.
[126] F. Menasri, N. Vincent, E. Augustin, M. Cheriet. Shape-based Alphabet for Off-line Arabic Handwriting Recognition. Proceedings of the 9th International Conference on Document Analysis and Recognition ICDAR, Curitiba, Brazil, p. 969-973, 2007.
[127] F. Menasri. Contributions à la reconnaissance de l'écriture arabe manuscrite. Thèse de doctorat, Université paris Descartes, 2008.
[128] E. Augustin. Reconnaissance de mots manuscrits par systèmes hybrides Réseaux de Neurones et Modèles de Markov Cachés. PhD thesis, Université Rene Descartes - Paris V, 2001.
[129] D. Motawa, A. Amin, R. Sabourin. Segmentation of arabic cursive script. In ICDAR '97: Proceedings of the 4th International Conference on Document Analysis and Recognition, pages 625–628. IEEE Computer Society, 1997.
[130] G. Olivier, H. Miled, K. Romeo, Y. Lecourtier. Segmentation and coding of arabic handwritten words. In ICPR96, pages III : 264–268, 1996.
[131] A. Amin, J.F. Mari. Machine recognition and correction of printed Arabic texts. IEEE Transactions on systems, man, and cybernetics, 19 (5) : 1300-1304, September/October 1989.
[132] M.A. Mahjoub : "Apport de la modélisation de la durée d'état dans la reconnaissance en ligne des caractères arabes par HMMs (1)". 17èmes Journées tunisiennes en électrotechnique et automatique (JTEA'97), Nabeul, Tunisie, p. 341-348, 1997.
[133] M.A. Mahjoub, N. Ellouze. Nonstationary hidden Markov model- application to on-line arabic character recognition. Computational engineering in systems applications (CESA'98), Nabeul-Hammamet, Tunisie, p.111-116, 1998.
[134] M.A. Mahjoub. Application des modèles de Markov cachés stationnaires et non stationnaires à la reconnaissance en-ligne des caractères arabes. Thèse de doctorat, Université des sciences, des techniques et de médecine de Tunis II, Tunisie, 1999.
[135] M.C. Fehri, M. Ben Ahmed. Off-line Arabic handwriting recognition. Computational engineering in systems applications (CESA'98), Nabeul-Hammamet, Tunisie, p.1-3, 1998.
[136] M.C. Fehri : "Reconnaissance de textes arabes multifontes à l'aide d'une approche hybride neuro-markoviennes". Thèse de doctorat, Université des sciences, des techniques et de médecine de Tunis II, Tunisie, 1999.
[137] H. Miled, C. Olivier, M. Cheriet, Y. Lecourtier . Coupling observation/letter for a Markovian modelisation applied to the recognition of arabic handwriting. IEEE Proc. 4th

International conference on document analysis and recognition (ICDAR'97), Ulm, Germany p. 580-583, 1997.
[138] H. Miled, M. Cheriet, C. Olivier, Y. Lecourtier. Modélisation markovienne de l'écriture arabe manuscrite: une approche analytique. Proc. 1er Colloque international francophone sur l'écrit et le document (CIFED'98), Québec, Canada, p. 50-59, 1998.
[139] H. Miled. Stratégies de reconnaissance de l'écriture semi cursive : application aux mots manuscrits arabes. Thèse de doctorat, Université de Rouen, 1998.
[140] M. S. Khorsheed. Recognising handwritten arabic manuscripts using a single hidden markov model. Pattern Recogn. Lett., 24(14) :2235–2242, 2003.
[141] T. Y. Zhang, C. Y. Suen. A fast parallel algorithm for thinning digital patterns. Communications of the ACM, 27(3) :236–239, 1984.
[142] R. El-Hajj, L. Likforman-Sulem, C. Mokbel. Arabic handwriting recognition using baseline dependant features and hidden markov modeling. In ICDAR, p. 893–897, 2005.
[143] R. El-Hajj, C. Mokbel, L. Likforman-Sulem. Reconnaissance de l'écriture arabe cursive : combinaison de classifieurs MMCs à fenêtres orientées. In CIFED, 2006.
[144] A. Benouareth, A. Ennaji, M. Sellami. Semi-continuous HMMs with explicit state duration applied to arabic handwritten word recognition. In The Tenth International Workshop on Frontiers in Handwriting Recognition (IWFHR 10), La Baule, France, 2006.
[145] A. Benouareth, A. Ennaji, M. Sellami. Semi-continuous HMMs with explicit state duration for unconstrained Arabic word modeling and recognition. Pattern Recognition Letters 29:1742–1752, 2008.
[146] R. Schwartz, C. LaPre, J. Makhoul, C. Raphael, Y. Zhao. Language-independent OCR using a continuous speech recognition system. IEEE Proc. 13th International conference on pattern recognition (ICPR'96), Vienne, Autriche, p. 99-103, 1996.
[147] H. Miled, N. E. B. Amara. Planar Markov modeling for arabic writing recognition: Advancement state. In ICDAR '01: Proceedings of the Sixth International Conference on Document Analysis and Recognition, p. 69–73, 2001.
[148] V. Märgner, M. Pechwitz, H. El Abed. Arabic handwriting recognition competition. In ICDAR, p.70–74, 2005.
[149] A. Boukharouba, A. Bennia. Recognition of Handwritten Arabic Literal Amounts Using a Hybrid Approach. Cognitive Computation, 3(2) : 382–393, 2011
[150] H. Freeman. Computer processing of line-drawing images. Computing Surveys, 6(1):57–97, March 1974.

[151] J. Iivarinen, A. Visa. Shape recognition of irregular objects. In David P. Casasent, editor, Intelligent Robots and Computer Vision XV: Algorithms, Techniques, Active Vision, and Materials Handling, Proc. SPIE 2904, p. 25–32, 1996.

[152] M. Shridhar, A. Badreldin. Recognition of isolated and connected handwritten numerals. Proc. IEEE International Conference on Systems, Man and Cybernetics, p. 142 146, 1984.

[153] E. Lethelier, M. Leroux, M. Gilloux. Traitement des montants numériques des chèques postaux, approche d'une méthode de segmentation basée sur la reconnaissance. Actes de CNED 94 (3ème Colloque National sur l'Ecrit et le Document), Rouen, France, p. 315-323, 1994.

[154] JM. Naik, DM. Lubensky. A hybrid HMM-MLP speaker verification algorithm for telephone speech. Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) p. 153-156, 1994.

[155] Ha TM, Bunk H. Off-line handwritten numeral recognition by perturbation method. IEEE Trans. Pattern Anal. Mach. Intell. 19(5) :535-539, 1997.

[156] LS. Oliveira, R. Sabourin, F. Bortolozzi, CY. Suen. Automatic recognition of handwritten numerical strings: a recognition and verification strategy. IEEE Trans. Pattern Anal. Mach. Intell. 24(11):1438-1454, 2002.

[157] J. Liu, P. Gader. Neural networks with enhanced outlier rejection ability for off-line handwritten word recognition. Pattern Recognition. 35: 2061-2071, 2002.

[158] N. Farah, L. Souici, M. Sellami. Classifiers combination and syntax analysis for Arabic literal amount recognition. Engineering Applications of Artificial Intelligence. 19 : 29–39, 2006.

[159] A. Benouareth, A. Ennaji, M. Sellami. Semi-continuous HMMs with explicit state duration for unconstrained Arabic word modeling and recognition. Pattern Recognition Lett. 29: 1742–1752, 2008.

[160] J. Makhoul, R. Schwartz, C. Lapre, I. Bazzi. A script independent methodology for optical character recognition. Pattern Recognition. 31 (9):1285-1294, 1998.

[161] M. Deghan, K. Faez, M. Ahmadi, M. Shridhar. Handwritten Farsi (Arabic) word recognition: A holistic approach using discrete HMM. Pattern Recognition. 34:057-1065, 2001.

[162] G. R. Ball, S. N. Srihari, H. Srinivasan. Segmentation-based and segmentation-free methods for spotting handwritten arabic words. In Guy Lorette and Suvisoft, editors, Tenth International Workshop on Frontiers in Handwriting Recognition, October 2006.

[163] H. Khosravi, E. Kabir. Introducing a very large dataset of handwritten Farsi digits and a study on the variety of handwriting styles. Pattern Recognition Letters, 28 (10): 1133-1141, 2007.

[164] A. Alaei, P. Nagabhushan, U. Pal. Fine classification of unconstrained handwritten Persian/Arabic numerals by removing confusion amongst similar classes. Proc. 10th International Conference on Document Analysis and Recognition, 2009.

[165] A. Alaei, U. Pal, P. Nagabhushan. Using modified contour features and SVM based classifier for the recognition of Persian/Arabic handwritten numerals. Proc. 7th International Conference on Advances in Pattern Recognition, p.391-394, 2009.

[166] H. Parvin, H. Alizadeh, B. Minaei-Bidgoli, M. Analoui. A scalable method for improving the performance of classifiers in multiclass applications by pairwise classifiers and GA. Proc. Fourth International Conference on Networked Computing and Advanced Information Management. 2008.

[167] H. Parvin, H. Alizadeh, M. Moshki, B. Minaei-Bidgoli, N. Mozayani. Divide & conquer classification and optimization by genetic algorithm. Proc. Third International Conference on Convergence and Hybrid Information Technology, 2008.

[168] V.N. Vapnik, “The Nature of Statistical Learning Theory,” Springer, New York, 1995.

[169] B. E Boser, I. M Guyon and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” Proc. 5th Annual Workshop on Computational Learning Theory. 1992, pp. 144–152.