ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

THÈSE PRÉSENTÉE À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

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

PAR
FRÉDÉRIC GRANDIDIER

UN NOUVEL ALGORITHME DE SÉLECTION DE CARACTÉRISTIQUES – APPLICATION À LA LECTURE AUTOMATIQUE DE L'ÉCRITURE MANUSCRITE

MONTRÉAL, LE 24 JANVIER 2003

CETTE THÈSE A ÉTÉ ÉVALUÉE

PAR UN JURY COMPOSÉ DE :

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

M. Ching Y. Suen, codirecteur
Centre for Pattern Recognition and Machine Intelligence, Concordia University

M. Mohamed Cheriet, président du jury
Département de génie de la production automatisée à l'École de technologie supérieure

M. Marc Parizeau, examinateur externe
Département de génie électrique et de génie informatique à l'Université Laval

M. Pierre Dumouchel, examinateur
Département de génie électrique à l'École de technologie supérieure

M. Richard Lepage, examinateur invité
Département de génie de la production automatisée à l'École de technologie supérieure

ELLE A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 19 DÉCEMBRE 2002

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

UN NOUVEL ALGORITHME DE SÉLECTION DE CARACTÉRISTIQUES – APPLICATION À LA LECTURE AUTOMATIQUE DE L'ÉCRITURE MANUSCRITE

Frédéric Grandidier

SOMMAIRE

La problématique abordée dans cette thèse est celle de la reconnaissance de l'écriture manuscrite hors-ligne, avec pour application industrielle le tri automatique du courrier. En effet le Service de Recherche Technique de La Poste (France) nous a donné pour mandat d'améliorer son système de reconnaissance de l'écriture manuscrite. Une analyse approfondie du système existant a permis de dégager une direction principale de recherche : l'amélioration de la représentation de l'information fournie au système de reconnaissance. Elle est caractérisée par deux ensembles finis de primitives, qui sont combinés avant intégration dans le système, au moyen d'un produit cartésien.

L'amélioration de la représentation de l'information passe par l'extraction de nouvelles primitives. Dans cette optique, trois nouveaux espaces de représentation ont été développés. L'utilisation d'un algorithme de quantification vectorielle permet de construire plusieurs ensembles de primitives. Afin d'augmenter le pouvoir discriminant de ces dernières, différentes stratégies ont été évaluées : l'analyse discriminante linéaire, la technique de *zoning* et en association avec cette dernière une stratégie de pondération des zones. La combinaison des espaces de représentation et des stratégies d'amélioration a conduit à la construction de plusieurs systèmes de reconnaissance obtenant de meilleures performances que système de base.

La technique permettant de combiner les ensembles de primitives dans le système de base ne peut pas être utilisée. Un nouvel algorithme a été développé afin d'intégrer de nouveaux ensembles de primitives. L'idée de base est de remplacer les primitives les moins discriminantes d'un ensemble de départ par de nouvelles. Une stratégie effectuant des regroupements de primitives non-discriminantes permet de décomposer la tâche globale de reconnaissance en sous-problèmes. La définition et la sélection dynamique de nouvelles primitives est alors orientée par cette décomposition. L'application de l'algorithme aboutit à une représentation de l'information améliorée caractérisée par une hiérarchie de primitives. Son déroulement automatique permet une adaptation rapide à de nouvelles données ou à la disponibilité d'un nouvel espace de représentation. Les performances du système de base, utilisant la combinaison de deux ensembles de primitives est de 89,5% lors de l'utilisation d'un lexique de taille 1 000. L'amélioration d'un des deux ensembles conduit à une performance de 94,3%, tout en diminuant de 20% le nombre de primitives utilisées.

A NEW FEATURE SELECTION ALGORITHM – APPLICATION TO AUTOMATIC READING OF CURSIVE HANDWRITING

Frédéric Grandidier

ABSTRACT

The global theme of this thesis is the off-line handwriting recognition dedicated to the industrial automatic mail sorting application. The authors have obtained a contract with SRTP (Service de Recherche Technique de La Poste) to improve their off-line handwriting recognition system, that is currently integrated in some mail sorting machines. An exhaustive analysis of the SRTP system shows that its main weakness is the representation of the image information provided to the recognition system. Two information sources are combined by Cartesian multiplication, thus allowing their integration in the system.

The improvement of the information representation can be achieved by extracting several new feature sets. With this goal, we develop three feature spaces, allowing the building of feature sets with the help of a vector quantization algorithm. In order to improve the discriminative power of these features, we propose a new strategy allowing the use of linear discriminant analysis. Moreover, several zoning strategies were used in order to take into account some contextual information. A weighting technique was also applied in association with zoning to integrate sample style information during feature extraction. The combination of feature spaces and the above strategies leads to a significant improvement in the SRTP system recognition rates.

In order to integrate the above information sources into the recognition system, we have developed a new algorithm. The main idea is to replace the less discriminating features of a given set by new ones. We propose to gather some non-discriminative features according to the given properties. The resulting groups can be viewed as several sub-problems of the global recognition task. The dynamic definition and selection of new feature sets, according to the properties of each group, allow the combination of the information sources in a relevant manner. Finally, the algorithm application results in a feature hierarchy. The automatic aspect of the algorithm gives it a fast adaptability to new data or information sources. Its application on a feature set used by the SRTP system shows an absolute improvement of more than 5% in the recognition rate, compared to those obtained by the basic system using two feature sets. Moreover, a reduction of 20% in the number of features can be achieved.

REMERCIEMENTS

En premier lieu je tiens à remercier Robert Sabourin pour être à l'origine de ce projet et bien sûr pour en avoir assumé la direction. Ma gratitude va également à Ching Y. Suen pour sa direction et ses conseils avisés.

Je voudrais également remercier Mohamed Cheriet pour l'honneur qu'il me fait en présidant ce jury de thèse. Je remercie très sincèrement Pierre Dumouchel, Richard Lepage et Marc Parizeau pour avoir accepté d'évaluer ce travail.

Mes remerciements vont aussi à Michel Gilloux du Service de Recherche Technique de La Poste pour avoir fourni un système de reconnaissance fonctionnel, une base de données valable et un financement à ce projet. Un grand merci va à Abdenaïm El-Yacoubi pour son aide et ses conseils éclairés.

Je tiens à remercier tous les membres passés et présents du *Laboratoire d'Imagerie de Vision et d'Intelligence Artificielle* et du *Center for Pattern Recognition and Machine Intelligence* avec qui j'ai partagé bien plus qu'un lieu de travail et plus particulièrement Alceu, Alessandro et Alessandro, Flavio, Guido, Hamanaka, Il-Seok, Jonathan, Luis et Luiz, Marisa, Nedjem, Patrick et Paulo. Un merci particulier va à Christine pour son amitié et sa disponibilité.

Enfin je voudrais remercier du fond du cœur Angèle et Elise pour avoir été courageuses et de m'avoir attendu si longtemps...

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS ET SIGLES

CCV	Concavités
CEDAR	Center of Excellence on Document Analysis and Recognition
DDD	Distributions de Distances Directionnelles
HD	Histogrammes de Directions
LDA	Linear Discriminant Analysis
MMC	Modèles de Markov cachés
SRTP	Service de Recherche Technique de La Poste
$X(t)$	Variable aléatoire de paramètre réel t.
$\mathcal{T}$	Espace des paramètres ou espace du temps.
$\mathcal{S} = \{s_i\}$	Ensemble des états d'un processus stochastique discret.
N	Nombre d'états d'un processus stochastique discret
$\Pr(\cdot)$	Probabilité.
$\mathcal{Q}$	Séquence d'états décrivant une évolution d'un processus stochastique discret.
$q_t = s_i$	État à l'instant t d'un processus stochastique discret.
$B = \{b_{ij}\}$	Matrice de probabilités de transition associées à un processus stochastique.
$\Pi = \{\pi_i\}$	Vecteur des probabilités initiales d'un processus stochastique.
$\mathcal{O}$	Séquence d'observations.
$\mathcal{V} = \{v_i\}$	Ensemble des symboles d'observation associés à un MMC.
M	Nombre de symboles d'observation possibles.
$A = \{a_{ijk}\}$	Matrice de probabilités de transition d'un MMC avec émission des symboles sur les transitions.

$A' = \{a'_{ij\Phi}\}$	Matrice de probabilités de transition nulle d'un MMC avec émission des symboles sur les transitions.
Φ	Symbole associé à une transition nulle.
Λ	Un modèle de Markov caché.
α_t	Variable *forward*.
β_t	Variable *backward*.
δ_t	Probabilité du meilleur chemin aboutissant à l'instant t.
Ψ_t	Mémorisation de l'état menant à la plus forte probabilité δ_t.
ζ_t	Variable indiquant si la transition est nulle ou non.
P^*	Meilleure probabilité de sortie.
ξ_t^1	Probabilité de transition avec émission d'observation.
ξ_t^2	Probabilité de transition avec émission d'observation nulle.
γ_t	Probabilité d'être à un état donné à l'instant t.
$\delta(\cdot,\cdot)$	Variable indiquant si deux paramètres sont identiques.
$\mathbf{O_{App}}$	Corpus de données d'apprentissage.
$\mathbf{O_{Val}}$	Corpus de données de validation.
λ	Paramètre de l'interpolation de MMC.
$h(\cdot)$	Quantité d'information apportée par un évènement.
$i(\cdot;\cdot)$	Quantité d'information apportée par un évènement par rapport à la réalisation d'un second.
$H(\cdot)$	Entropie d'une variable aléatoire.
$I(\cdot;\cdot)$	Information mutuelle de deux variables aléatoires.
$\mathbf{x}$	Un vecteur de mesures.
$\mathcal{F} = \{f_i\}$	Un ensemble de primitives.

N_F	Nombre de primitives d'un ensemble.
$\mathcal{C} = \{\omega_i\}$	L'ensemble des classes d'une modélisation.
N_C	Nombre de classes de la modélisation.
$PP(C)$	Perplexité d'une variable aléatoire.
$q(\cdot)$	Opérateur de quantification.
$\mathcal{Z} = \{\mathbf{z}_i\}$	Alphabet résultant d'une quantification.
$\mathcal{R}$	Sous espace associé à chaque symbole d'un alphabet.
$cent(\cdot)$	Centre de gravité.
$d(\cdot,\cdot)$	Mesure de distorsion entre deux vecteurs.
$\mathcal{D}$	Distorsion moyenne associée à un centre de gravité
$\mathfrak{D}$	Distorsion globale associée à un processus de quantification.
W	Matrice contenant les composantes discriminantes.
$\mathbf{m}$	Vecteur moyen associé à une classe de modélisation.
$\bar{\mathbf{m}}$	Vecteur moyen de l'ensemble des échantillons.
S_j	Matrice de dispersion intra-classe associée à la classe j.
S_W	Matrice de dispersion intra-classe globale.
S_B	Matrice de dispersion inter-classes.
(x, y)	Coordonnées d'un pixel dans une image.
α_H	Angle de la transformée de Hook.
Ig	Une imagette : une partie d'image délimitée par deux points spécifiques.
β_{Glob}	Inclinaison globale des caractères d'une image.
h_m	Hauteur moyenne des maxima restants après la phase de filtrage.
y_{lb}	Ordonnée de la ligne de base.
$(ent)(\cdot)$	Valeur entière d'un réel.

$\mathcal{L}(\mathbf{O}_{Val}, \Lambda)$	Vraisemblance du corpus de validation par le modèle Λ.
σ_i	Différence relative de la vraisemblance de la base de validation entre deux itérations de l'algorithme d'apprentissage.
ϵ	Seuil permettant d'arrêter la phase d'apprentissage.
$\mathfrak{L}$	Vocabulaire associé à une application.
w	Un nom de commune.
$\hat{w}$	Nom de commune qui maximise la probabilité *a posteriori*.
h_{Sup}	Hauteur de la zone des dépassements hauts d'un échantillon.
h_{Med}	Hauteur de la zone médiane d'un échantillon.
h_{Inf}	Hauteur de la zone des dépassements bas d'un échantillon.
ρ_H	Rapport de hauteur entre la zone des dépassements hauts et la zone médiane.
ρ_B	Rapport de hauteur entre la zone des dépassements bas et la zone médiane.
$\mathbf{p}_i = [p_i]^t$	Vecteur de pondération associé à un vecteur de mesures.
$d_E(\cdot, \cdot)$	Distance euclidienne entre deux points.
$d_{Ep}(\cdot, \cdot)$	Distance euclidienne pondérée entre deux points.
τ_i	Seuil de pouvoir discriminant au niveau perceptif i.
D_i	Sous-ensemble des primitives discriminantes au niveau perceptif i.
$\bar{D}_i$	Sous-ensemble des primitives non discriminantes au niveau perceptif i.
$C_{i,j}$	Regroupement (ou classe) de primitives au niveau perceptif i.
$E_{i,j}$	Ensemble de primitive permettant de substituer le regroupement $C_{i,j}$.
$I_P(\cdot)$	Indicateur du pouvoir discriminant d'un modèle.
r	Compte associé à un n-gram.

r^*	Compte de Good/Turing
n_r	Nombre de n-grams apparu r fois.
$p_{GT}(\cdot)$	Estimation de Good/Turing d'un n-gram.
$p_K(\cdot)$	Estimation de Katz d'un n-gram.
$\theta_K(\cdot)$	Indicateur de compte nul d'un n-gram d'après Katz.
$\alpha_K(\cdot)$	Facteur de décompte des n-grams d'après Katz.
$KL(\cdot\|\|\cdot)$	Distance de *Kullback-Leibler* ou divergence.
$L_k(\cdot,\cdot)$	Métrique de Minkowski.
$L_1(\cdot,\cdot)$	Norme L_1 basé sur la métrique de Minkowski.
η_j	Heuristique pour la définition du nombre de primitives nécessaires pour remplacer un regroupement.
$\mathfrak{N}_{max}$	Nombre de primitives maximum pour remplacer un regroupement.
$\mathfrak{N}_{min}$	Nombre de primitives minimum pour remplacer un regroupement.
δ_P	Gain en performance entre deux itérations.

INTRODUCTION

L'écriture est une "représentation de la parole et de la pensée par des signes"[1]. Son invention, aux environs du IVè millénaire avant Jésus Christ, en Mésopotamie, marque le passage de la Préhistoire à l'Histoire de l'humanité. Les premières traces d'une écriture ont été découvertes sur des tablettes d'argile, dans les temples d'Uruk, au pays de Sumer. Les tablettes d'Uruk comportent des inventaires de grains et de bétail. Le but premier de l'écriture était donc de tenir une comptabilité. Ces premiers signes sont en fait des pictogrammes, c'est-à-dire des représentations stylisées, bien loin de notre alphabet. La première évolution de l'écriture est celle qui a permis au pictogramme de ne plus représenter un objet mais un concept abstrait lié. Elle caractérise le début d'une représentation de la parole et de la pensée. L'étape suivante fut considérable puisqu'elle a conduit à l'utilisation de signes pour représenter les sons de la langue parlée. L'invention de l'alphabet vers le Xè siècle avant Jésus Christ, par les Phéniciens et les Grecs, a permis à l'écriture de se répandre. Cependant l'écriture est restée élitiste très longtemps, très certainement de par le pouvoir qu'elle conférait. L'invention de l'imprimerie, attribuée à Johannes Gensfleisch, dit "Gutenberg", vers le milieu du XVè siècle, a entraîné une véritable révolution. En effet elle a permis une plus large diffusion et donc une plus grande démocratisation du savoir et de la culture. Au cours du siècle passé, l'évolution de l'écriture s'est poursuivie avec l'apparition de l'électronique et de l'informatique ; elle est devenue numérique. Cette forme, de part sa nature, permet d'accroître considérablement la disponibilité et donc la diffusion du savoir contenu dans l'écriture.

L'écriture est restée, jusqu'au siècle dernier, le seul moyen matériel de transmission des connaissances. De ce fait, l'homme a toujours développé des techniques visant sa pérennité et sa diffusion. Aujourd'hui, la prolifération des ordinateurs dans notre société conduit à une dématérialisation du support de l'écriture, au profit de sa forme numérique. Ainsi sont nés les concepts de société sans papier et de bibliothèque virtuelle. Paradoxalement,

[1] Définition du dictionnaire Petit Robert.

notre société de consommation produit un nombre considérable de documents écrits : lettres, enveloppes, chèques, formulaires, mémentos,.... Il n'est alors pas surprenant de constater le développement de nombreuses techniques visant à la conversion de ces documents vers la forme numérique de l'écrit. Le résultat obtenu peut alors être diffusé plus largement (via le mass médium qu'est devenu internet), être stocké sous une forme numérique de durée de vie prétendue plus longue que le papier et évidemment permettre le traitement automatique de l'information par l'intermédiaire d'ordinateurs.

La problématique

C'est cette dernière problématique qui est abordée dans ce travail et plus particulièrement la lecture automatique de l'adresse présente sur les enveloppes de courrier. Le tri automatique du courrier est une des principales applications industrielles de la reconnaissance de l'écriture. L'enjeu économique est d'importance puisqu'en France, un pays de 60 millions d'habitants, la quantité de plis traités par La Poste[2] est de plus de 75 millions par jour. Le chiffre d'affaires correspondant à l'activité courrier au sein de ce groupe a atteint 10,06 milliards d'euros (16,8 milliards de dollars canadiens) en 2001 [131]. Étant donnée la quantité de courrier à traiter, l'automatisation de son tri est nécessaire, afin que cette entreprise puisse assurer le service public d'acheminement du courrier.

L'automatisation du tri du courrier est un processus complexe, comportant une composante mécanique importante. Le cerveau d'une telle machine est bien sûr la partie permettant la prise de décision quant à l'acheminement du pli traité. Cette dernière est composée de plusieurs modules dont celui qui effectue la reconnaissance de l'écriture. À ce niveau une différence de traitement est faite suivant la nature de l'écrit. En effet l'imprimé n'est pas traité de la même manière que le manuscrit. L'imprimé est caractérisé par une grande régularité, qui est bien sûr exploitée lors de sa reconnaissance. Par opposition, la grande

[2]http ://www.laposte.fr

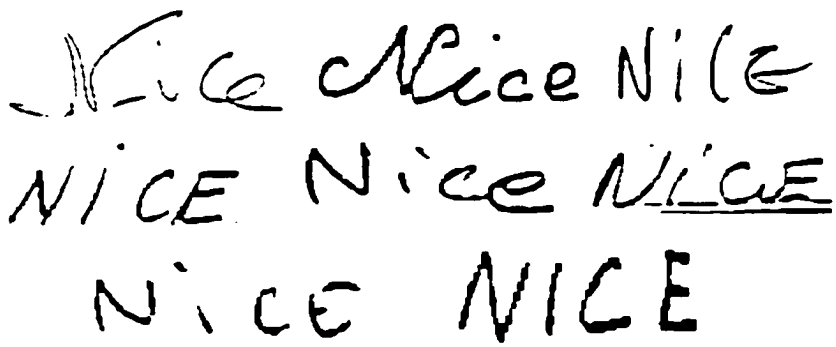


FIGURE 1 Plusieurs échantillons du nom de ville `Nice` présentant différents styles d'écriture.

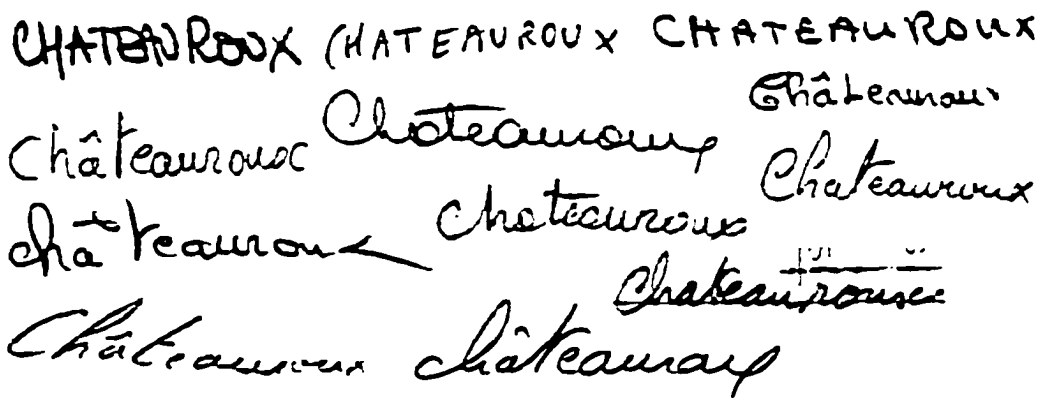


FIGURE 2 Plusieurs échantillons du nom de ville `Châteauroux` présentant différents styles d'écriture.

variabilité associée à l'écriture manuscrite complique fortement la tâche de classification. C'est cette problématique particulière qui est le sujet de cette thèse de doctorat.

Une des spécificités de la reconnaissance de l'écriture manuscrite est la grande variabilité associée à ce signal. Afin de visualiser ce phénomène, nous présentons aux figures 1 et 2 plusieurs exemples d'un même nom de ville. Les images sont extraites de notre base de données, composée d'enveloppes réelles du courrier français. Les sources de variation de

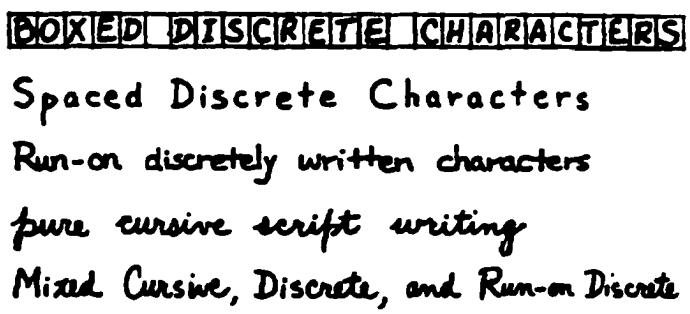


FIGURE 3 Les différents styles d'écriture manuscrite d'après Tappert *et al.* [149].

l'écriture sont multiples. La plus importante est bien sûr le scripteur. En effet chacun a une écriture propre. Cette singularité est exploitée au travers de la reconnaissance de signature afin d'identifier un scripteur. D'autres sources de variations sont l'outil d'écriture et le support utilisés. L'humeur de la personne au moment de la réalisation peut également conduire à une déformation du signal. Cependant, la reconnaissance d'adresses postales est principalement soumise à la première source de variabilité énoncée.

De l'analyse des figures 1 et 2 découle une conclusion : différents styles sont utilisés pour l'écriture des noms de ville sur des enveloppes de courrier. Certains scripteurs détachent tous les caractères alors que d'autres les lient. On parle pour le premier cas d'une écriture discrète par opposition à l'écriture cursive. La première est la plupart du temps composée de lettres majuscules encore appelées bâtons. La seconde quant à elle utilise une majorité de lettres minuscules. Une classification des styles d'écriture manuscrite a été proposée par Tappert *et al.* [149]. Elle comporte cinq styles différents présentés sur la figure 3. Les appellations utilisées peuvent se traduire de la manière suivante : discret pré-casé, discret avec caractères disjoints, discret avec certains caractères liés, purement cursif et finalement mixte.

Concernant l'écriture présente sur les enveloppes, les deux derniers styles sont les plus utilisés par les scripteurs. En fait ils correspondent à une écriture manuscrite naturelle sans contrainte. Le premier style est présent sur une quantité d'enveloppes grandissante, mais

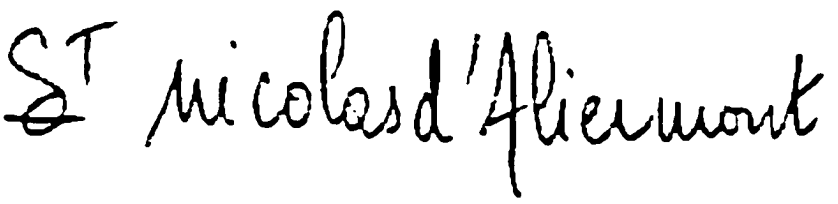


FIGURE 4 Échantillon cursif - Intitulé `St_nicolas_d'Aliermont`.

uniquement pour l'inscription du code postal. La contrainte de localisation des caractères imposée au scripteur facilite la lecture automatique. Cette propriété explique le nombre croissant d'enveloppes pré-casées proposées dans le commerce.

Intuitivement, la reconnaissance d'un mot passe par l'identification des différents caractères – entités de base de l'écriture – qui le composent. Lorsque l'échantillon traité est de type discret, cette idée est facilement exploitable. Par contre pour une écriture cursive, un problème survient : il faut segmenter le mot en caractères. Cette remarque permet de mentionner le paradoxe associé à la reconnaissance de l'écriture manuscrite énoncée par Sayre dans [137]. Il s'exprime de la manière suivante : pour reconnaître un caractère, on a besoin de savoir où il commence et où il se termine, mais pour isoler un caractère, il faut l'avoir reconnu auparavant. La figure 4 permet de bien visualiser ce problème. En effet il n'est pas évident de proposer une procédure automatique de décomposition d'un nom de ville en caractères. Une première difficulté est la segmentation en mots du nom de ville. Sur cet exemple le scripteur a placé un espace entre `ni` et `co` aussi important voire plus que celui entre `las` et `d'`. Ce phénomène est directement imputable au style d'écriture du scripteur. La seconde est la décomposition du mot en caractères. Si on ne regarde que la partie `iermon` de cet exemple, les caractères sont difficilement identifiables. En effet les lettres `r`, `m` et `n` peuvent très bien être décomposées ou combinées de manière à former des `u` ou des `i`.

Cette dernière remarque permet de mentionner l'importance du contexte lors de la reconnaissance de l'écriture manuscrite. En effet, lors de la lecture par l'humain de ce nom

FIGURE 5 Échantillon cursif - Intitulé Narbonne.

de ville, la présence du t final permet de lever l'ambiguïté associée aux caractères précédents. Notre connaissance de la langue française nous indique que la combinaison ont est beaucoup plus probable que out ou oirt, en particulier dans le vocabulaire des noms de communes françaises, où le suffixe mont est fréquent. L'homme effectue ce processus de manière quasiment inconsciente lors de la lecture d'écriture manuscrite. L'intégration d'information contextuelle et des concepts associés, à un système de lecture automatique, est un processus complexe.

Avant d'intégrer une quelconque information contextuelle dans un système de lecture automatique, il faut fournir à ce dernier une représentation adéquate de l'information présente dans l'image. Pour cela une étape de codage est réalisée. En plus de réduire la quantité d'informations, elle doit permettre d'extraire des caractéristiques pertinentes pour la phase de reconnaissance. L'utilisation d'algorithmes de traitement d'image permet par exemple d'évaluer la présence ou l'absence de boucles, mesurer leur superficie ou encore décomposer le contour de l'écriture à l'aide des descripteurs de Fourier. C'est une étape primordiale d'un système puisque la représentation obtenue conditionnera tout le reste du processus de reconnaissance. L'analyse des différents échantillons d'écriture manuscrite présentés dans cette section, ainsi que les différentes remarques les concernant, montrent clairement la difficulté d'extraire une information pertinente pour l'ensemble des caractères. La difficulté est accentuée pour les styles d'écriture cursive.

La problématique de notre projet est donc celle de la reconnaissance de l'écriture manuscrite et l'application visée est la lecture automatique des adresses postales. Plus précisé-

ment un travail précédent à notre thèse a conduit à la construction d'un système efficace [36, 38]. Il est intégré par le Service de Recherche Technique de La Poste (SRTP[3]) aux machines de tri du courrier français et traite les rejets du système de reconnaissance principal fourni par le constructeur de la machine. Ce système permet d'obtenir de bonnes performances, cependant elles chutent de manière non négligeable dans certaines conditions d'utilisation. Le SRTP nous a donnés pour mandat de développer de nouveaux algorithmes afin d'apporter une solution à ce problème.

Méthodologie et principales contributions

Afin d'orienter nos travaux, une analyse quasi exhaustive du système de base du SRTP a été réalisée [58]. Elle a permis de mettre en évidence son principal point faible : la relative faiblesse de la représentation de l'information utilisée. Nous avons montré que la reconnaissance d'échantillons de petite taille souffrait particulièrement du manque de caractéristiques discriminantes.

La solution apportée à ce problème est le développement de techniques d'extraction de caractéristiques différentes et complémentaires à celles déjà disponibles. Pour cela plusieurs espaces de représentation de l'information ont été mis en œuvre, permettant la construction de plusieurs ensembles de primitives. Afin d'augmenter le pouvoir discriminant de ces dernières, différentes techniques ont été testées. Premièrement une stratégie a été développée pour effectuer un étiquetage automatique des graphèmes par leur classe. L'exploitation de cette information passe par l'application de l'algorithme LDA (*Linear Discriminant Analysis*) qui permet la construction d'ensembles de primitives plus discriminants. Une autre possibilité étudiée est la prise en compte localisée de l'information présente dans les graphèmes par l'intermédiaire de la technique de *zoning*. Cette dernière consiste à diviser l'image en plusieurs zones et à effectuer une extraction de caractéristiques pour chacune d'elles, de manière indépendante. Cette technique permet de prendre

[3]http ://www.srt-poste.fr

en compte une information contextuelle locale. Différentes constatations au sujet des performances du système nous ont conduits à proposer une nouvelle stratégie de pondération des zones. Cette dernière permet de prendre en compte d'une certaine manière le style de l'échantillon traité directement au niveau de la définition des primitives. Un grand nombre de systèmes de reconnaissance sera construit afin d'évaluer la pertinence des espaces de représentation ainsi que celles des techniques développées pour l'amélioration du pouvoir discriminant des primitives.

Dans le système de base, l'intégration des deux sources d'information disponibles est réalisée en prenant en compte les ensembles de primitives conjointement. Cette stratégie consiste à construire un seul ensemble de primitives égal au produit cartésien des deux autres. Les nouvelles sources d'information peuvent être intégrées de la même manière. Cependant cette possibilité conduit à une augmentation exponentielle du nombre de primitives prises en compte par le système de reconnaissance. Cette dernière s'accompagne d'un accroissement proportionnel du nombre de paramètres. Dans une certaine limite, une telle éventualité est réalisable, mais elle nécessite, en contrepartie, une augmentation du nombre d'échantillons d'apprentissage, afin d'obtenir une estimation fiable de ces paramètres. Disposant d'une base de données de taille fixe, nous ne pouvons pas envisager cette stratégie. Nos travaux ont alors été orientés vers la recherche et le développement d'un nouvel algorithme visant l'intégration de différentes sources d'information à un système de reconnaissance.

Une évaluation du pouvoir discriminant des primitives utilisées par le système de base a permis de mettre en évidence que certaines sont suffisamment discriminantes pour la tâche de reconnaissance, alors que d'autres ne le sont pas. Une telle constatation peut être étendue à n'importe quel ensemble de primitives. Notre approche consiste à identifier les primitives non-discriminantes et de les substituer par de nouvelles. En partant d'un ensemble de primitives de départ, l'algorithme développé permettra d'améliorer la représentation de l'information dont dispose le système de reconnaissance. Un des points

fort de cet algorithme est le concept de regroupements (ou classes) de primitives. Leur identification au sein du sous-ensemble de primitives non-discriminantes conduit à une division en sous-problèmes de la tâche globale de classification. Notre algorithme permet alors de sélectionner dynamiquement les meilleurs ensembles de primitives pour chaque regroupement, de manière individuelle. Cette approche peut être interprétée comme une optimisation locale du problème global de reconnaissance. L'algorithme développé est de type itératif. L'amélioration de la représentation de l'information est effectuée en plusieurs étapes. L'ensemble de primitives utilisé par le système étant sa seule perception de l'information nous parlons alors de niveaux perceptifs. L'ensemble de départ est considéré comme le premier niveau perceptif. Chaque itération conduit à un nouveau niveau perceptif, caractérisé par un ensemble de primitives. Le résultat du déroulement de l'algorithme est une représentation hiérarchisée de l'information. Étant donnée l'approche proposée de définition et de sélection des primitives, celles des niveaux inférieurs contiennent implicitement l'information de celles situées plus haut dans la branche concernée de la hiérarchie. Un autre avantage de cette approche est que lors du fonctionnement en mode production du système, l'extraction d'information est guidée par la hiérarchie et uniquement les primitives nécessaires sont extraites. L'algorithme proposé sera appliqué sur plusieurs ensembles de primitives afin de prouver ses qualités.

Plan du manuscrit

Le domaine de la reconnaissance de l'écriture manuscrite fait appel à divers domaines de recherche. Au cours de notre travail, nous en avons abordé principalement deux, qui sont l'extraction de caractéristiques et la sélection de caractéristiques. Pour la rédaction de cette thèse, nous avons fait le choix éditorial de consacrer un chapitre pour chacun de ces domaines. Nous proposons alors une revue de littérature spécifique en début de chapitre. Afin de définir la notation utilisée et d'alléger le contenu de cette thèse, nous proposons de consacrer le premier chapitre à la description des différents outils et notions théoriques utilisés pour le développement de notre projet. Dans un premier temps nous présenterons

le formalisme associé aux modèles de Markov cachés. Un concept important est celui de la quantité d'information. Cette notion liée à la théorie de l'information sera décrite dans ce premier chapitre. Finalement une introduction au domaine de la quantification vectorielle y est également présente ainsi qu'une description de l'analyse discriminante linéaire.

Le second chapitre traite des systèmes de reconnaissance de l'écriture manuscrite. Après une brève revue de la littérature, le système de base fourni par le SRTP est présenté, ainsi que son évaluation. Les conclusions découlant de cette dernière ont permis d'orienter nos travaux. Le chapitre 3 est consacré à l'extraction de caractéristiques. La revue de littérature du domaine permet de constater la grande diversité des techniques permettant l'extraction d'information d'une image. Trois d'entre elles ont été mises en œuvre au cours de notre travail, elles sont plus longuement décrites. Une section est également consacrée aux stratégies permettant d'augmenter le pouvoir discriminant des primitives. Le quatrième chapitre traite de la sélection de caractéristiques. Il permet de présenter un nouvel algorithme permettant de définir et sélectionner dynamiquement des primitives afin d'améliorer la représentation de l'information caractérisée par un ensemble de primitives de départ.

La description des différentes expériences réalisées et l'analyse des résultats obtenus sont regroupées dans le chapitre 5. Finalement le dernier chapitre contient nos conclusions quant au travail réalisé. Rien n'étant parfait, une section est consacrée aux améliorations possibles et donc aux perspectives envisagées de ce travail.

CHAPITRE 1

OUTILS THÉORIQUES

Ce chapitre nous permettra d'exposer les différents outils théoriques utilisés au cours du développement de notre projet. Dans un premier temps la théorie de la modélisation Markovienne sera présentée. En effet le système de reconnaissance utilisé au cours de ce travail est basé sur une modélisation markovienne des caractères. La section suivante permettra d'introduire le vaste domaine de la théorie de l'information et plus particulièrement les différents indicateurs permettant de quantifier l'information. L'application de ces concepts au domaine de la classification sera également exposée dans cette section. Une définition de la quantification vectorielle sera donnée dans la section suivante, ainsi que deux algorithmes permettant de la mettre en œuvre. L'utilisation de cette technique a permis de discrétiser différents espaces de représentation de l'information continus utilisés lors de la construction d'ensembles de primitives discrètes. La dernière section sera consacrée à la technique d'analyse statistique de données appelée l'analyse discriminante linéaire (Linear Discriminant Analysis LDA en anglais), qui permet d'augmenter la capacité de discrimination d'un ensemble de primitives.

1.1 Les modèles de Markov cachés

1.1.1 Définitions

1.1.1.1 Processus stochastiques

Un processus stochastique $\{X(t): t \in \mathcal{T}\}$ est une suite de variables aléatoires où t est un paramètre réel. L'*espace des paramètres* ou *espace du temps* $\mathcal{T}$ peut être discret : $\mathcal{T} = \{0, 1, 2, \ldots\}$ ou continu : $\mathcal{T} = [0, \infty[$. Nous parlons alors de processus stochastiques à temps discret ou continu. L'*espace des états* $\mathcal{S}$ est l'ensemble dénombrable des valeurs prises par l'ensemble des variables aléatoires du processus stochastique. Dans le

cas où l'espace des états est discret : $\mathcal{S} = \{s_0, s_1, \ldots, s_{N-1}\}$, nous parlons de processus stochastiques à états discrets.

L'évolution d'un processus stochastique est une suite de transitions d'états : $s_0 s_1 \ldots s_T$, où dans cette notation s_0 correspond à l'état du processus à l'instant 0. Sa loi d'évolution est obtenue à l'aide de la probabilité $\Pr(s_0 \ldots s_T)$. Cette dernière est définie de proche en proche de la manière suivante :

$$\begin{aligned} \Pr(s_0 \ldots s_T) &= \Pr(s_0 \ldots s_{T-1}) \times \Pr(s_T | s_0 \ldots s_{T-1}) \\ &= \Pr(s_0) \times \Pr(s_1|s_0) \times \Pr(s_2|s_0 s_1) \times \cdots \times \Pr(s_T|s_0 \ldots s_{T-1}) \end{aligned} \tag{1.1}$$

La caractérisation du processus se résume donc à l'obtention des probabilités initiales $\Pr(s_0)$ et des probabilités des états conditionnés par les évolutions antérieures. La situation générale est celle où la loi de probabilité des états, à un certain instant, dépend de la totalité de l'histoire du processus. On dira que celui-ci garde la mémoire de son passé.

1.1.1.2 Propriétés de Markov du premier ordre

Un processus stochastique vérifie la *propriété de Markov* du premier ordre si pour tout instant t :

$$\Pr(q_t = s_i | q_{t-1} = s_j, q_{t-2} = s_k, \ldots) = \Pr(q_t = s_i | q_{t-1} = s_j) \tag{1.2}$$

L'équation 1.1 peut alors être ré-écrite de la manière suivante :

$$\Pr(s_0 \ldots s_T) = \Pr(s_0) \times \Pr(s_1|s_0) \times \Pr(s_2|s_1) \times \cdots \times \Pr(s_T|s_{T-1}) \tag{1.3}$$

Un processus ainsi défini est appelé *processus de Markov du premier ordre*. On dira parfois aussi qu'un tel processus stochastique est sans mémoire. Il est dit stationnaire si pour tout

instant t et tout décalage temporel k :

$$\Pr(q_t = s_i | q_{t-1} = s_j) \;=\; \Pr(q_{t+k} = s_i | q_{t+k-1} = s_j) \tag{1.4}$$

On définit dans ce cas une *matrice de probabilité de transitions* ou *matrice de transitions* $B = \{b_{ij}\}$ telle que :

$$b_{ij} \;=\; \Pr(q_t = s_i | q_{t-1} = s_j) \tag{1.5}$$

avec : $0 \leq i, j \leq N - 1$. De plus

$$\forall i, j \quad b_{ij} \geq 0 \quad \text{et} \quad \forall i \quad \sum_{j=1}^{N} b_{ij} \;=\; 1 \tag{1.6}$$

On définit également le vecteur des probabilités initiales $\Pi = \{\pi_i\}$ pour tout i :

$$\pi_i \;=\; \Pr(q_0 = s_i) \tag{1.7}$$

Une *chaîne de Markov* est un processus stochastique qui vérifie les propriétés énoncées dans les équations 1.2 et 1.4. Ce processus est entièrement caractérisé par la matrice de transition B et le vecteur des probabilités initiales Π.

1.1.1.3 Modèles de Markov cachés

Les modèles de Markov cachés (MMC) sont des modèles doublement stochastiques dont la première composante est un processus stochastique non observable, donc caché, mais qui peut l'être par l'intermédiaire d'un second processus stochastique.

La structure d'un MMC est la même que celle d'une chaîne de Markov. L'évolution dans le temps du premier processus stochastique produit une séquence d'états $\mathcal{Q} = q_0 q_1 \cdots q_{T-1}$ de la même manière qu'une chaîne de Markov. La différence entre chaîne de Markov

et MMC se situe au niveau des observations du phénomène analysé. Pour une chaîne de Markov, chaque état correspond à une observation unique. Pour les MMCs une fonction de densité de probabilité sur l'ensemble des observations possibles est associée à chaque état. Le second processus permet d'observer l'évolution du modèle à travers la séquence d'observations qu'il émet : $\mathcal{O} = o_0 o_1 \cdots o_{T-1}$. Étant donnée une séquence d'observations, il n'est pas possible de connaître directement la séquence d'états l'ayant produite car chaque état peut potentiellement émettre l'ensemble de ces observations. Généralement lors de la modélisation d'un phénomène évoluant dans le temps il est coutumier de considérer un état comme celui de départ et un comme fin du processus.

Il existe plusieurs types de MMCs. La première distinction se fait suivant la nature de la fonction de densité de probabilité utilisée pour la génération des observations. Lorsque la distribution est naturellement discrète ou obtenue par quantification, les MMCs sont qualifiés de *discrets*. L'utilisation d'une distribution continue, généralement approximée par une mixture de gaussienne, conduit à des *MMCs continus*. Il existe également un compromis entre ces deux familles appelé *MMCs semi-continus* [70]. En effet l'utilisation d'une quantification induit une perte d'information qui peut être préjudiciable aux modèles. D'un autre coté, le passage à une modélisation continue conduit à une augmentation importante du nombre de paramètres à estimer. Les MMCs semi-continus sont une alternative permettant d'optimiser le nombre global de paramètres du modèle.

Une autre distinction est effectuée entre les MMCs, suivant le mode d'émission des observations. Généralement les observations sont produites par les états du modèle, on parle alors de *modèles d'états*. Il est cependant possible de considérer l'émission des observations lors du franchissement des transitions il s'agit alors de *modèles d'arcs*. Le choix est guidé par l'application. Nous pouvons cependant mentionner qu'à nombre égal d'états, les modèles d'arcs permettent un plus grand nombre de possibilités quant à l'émission d'observations. Lors de la modélisation d'un phénomène par un modèle d'arcs, il peut être

intéressant de permettre le franchissement de transitions sans émission d'observation, en particulier pour modéliser l'absence d'un évènement.

Un autre paramètre permet de générer d'autres familles de MMCs : l'ordre du modèle. Ce dernier est relié à la propriété de Markov (équation 1.2). En augmentant la mémoire du processus, nous augmentons l'ordre du modèle.

1.1.2 Les paramètres des modèles de Markov cachés

Dans le cadre de notre projet, nous avons utilisé des MMCs du premier ordre, discret, avec émission des observations sur les arcs. Cette modélisation contient également un certain nombre de transitions nulles. De plus un état de départ et un état de fin sont utilisés. Le formalisme que nous allons présenter maintenant correspond à celui associé à de tels modèles. Pour un formalisme générique ou de plus amples détails sur les MMCs, le lecteur peut consulter entre autres les références suivantes : [31, 69, 76, 130, 135].

La définition de MMCs fait appel à un certain nombre de variables et paramètres :

- T : la longueur de la séquence d'observations,
- $\mathcal{O} = o_0 o_1 \cdots o_{T-1}$: une séquence d'observations, où o_t est l'observation à l'instant t,
- N : le nombre d'états du modèle,
- $\mathcal{S} = \{s_0, s_1, \ldots, s_{N-1}\}$: l'ensemble des états du modèle, s_0 étant l'état initial et s_{N-1} l'état final,
- $\mathcal{Q} = q_0 q_1 \cdots q_T$: une séquence cachée d'états du système, où q_t est l'état du système à l'instant t,
- M : le nombre de symboles d'observations possibles,
- $\mathcal{V} = \{v_1, v_2, \ldots, v_M\}$: l'ensemble des symboles d'observations possibles,
- $\Pi = \{\pi_i\}$ avec $\pi_i = \Pr(q_0 = s_i)$: la probabilité que le processus soit dans l'état s_i à l'instant 0.

- $A = \{a_{ijk}\}$ avec $a_{ijk} = \Pr(o_t = v_k, q_{t+1} = s_j | q_t = s_i)$: la probabilité d'être dans l'état s_i à l'instant t et à l'état s_j à l'instant $t+1$ et de produire lors du franchissement de la transition $s_i \rightarrow s_j$ le symbole d'observation v_k,
- $A' = \{a'_{ij}\} = \{a'_{ij\Phi}\}$ avec $a_{ij\Phi} = \Pr(o_t = \Phi, q_t = s_j | q_t = s_i)$: la probabilité de franchir la transition nulle de l'état s_i vers l'état s_j à l'instant t tout en émettant le symbole d'observation nul Φ. Il est à noter qu'il n'y a pas d'évolution du temps lors du franchissement de la transition puisqu'aucun symbole d'observation réel n'est produit.

Les éléments des deux matrices de transitions a_{ijk} et $a_{ij\Phi}$ doivent satisfaire la contrainte suivante :

$$\sum_{j=0}^{N-1} \left[a'_{ij\Phi} + \sum_{k=1}^{M} a_{ijk} \right] = 1 \quad \text{pour} \quad i = 0, 1, \ldots, N-1 \tag{1.8}$$

L'équation 1.8 traduit le fait que la somme des probabilités sortantes d'un état doit être égale à 1.

Par la suite nous adopterons la notation suivante pour désigner un MMC discret du premier ordre : $\Lambda = (T, N, M, \Pi, A, A')$ ou plus simplement $\Lambda = (\Pi, A, A')$.

1.1.3 Les trois problèmes associés aux modèles de Markov cachés

Afin de pouvoir exploiter le modèle Λ défini dans la section précédente avec des données réelles, trois problèmes de base doivent être résolus :

1. **Le problème d'évaluation** : soit une séquence d'observations $\mathcal{O} = o_0 o_1 \cdots o_{T-1}$ et un modèle $\Lambda = (\Pi, A, A')$, comment pouvons nous calculer efficacement la probabilité de cette séquence étant donné le modèle $\Pr(\mathcal{O}|\Lambda)$?
2. **Le problème de décodage ou de reconnaissance** : étant donnés une séquence d'observations $\mathcal{O} = o_0 o_1 \cdots o_{T-1}$ et un modèle $\Lambda = (\Pi, A, A')$, comment pouvons nous trouver la séquence optimale d'états $\mathcal{Q} = q_0 q_1 \cdots q_T$ qui a produit la séquence $\mathcal{O}$?

3. **Le problème d'apprentissage** : étant donnés un ensemble de séquences d'observations et un modèle initial Λ_0, comment ré-estimer les paramètres du modèle de manière à augmenter sa vraisemblance de génération de l'ensemble des séquences ?

1.1.3.1 Le problème d'évaluation : la procédure Forward

La solution la plus directe de ce problème consiste à énumérer toutes les séquences de longueur T possibles. Cependant cette technique nécessite un grand nombre de calculs (de l'ordre de $2TN^T$) et n'est pas réellement envisageable. Une procédure appelée *Forward* [135] permet grâce à une factorisation des chemins, de réduire considérablement le nombre d'opérations. Considérons la variable *forward* définie par :

$$\alpha_t(i) = \Pr(o_0 o_1 \cdots o_{t-1}, q_t = s_i|\Lambda) \tag{1.9}$$

c'est-à-dire la probabilité d'émission de la séquence partielle $o_0 o_1 \cdots o_{t-1}$ étant donné le modèle Λ, tout en atteignant l'état s_i à l'instant t. Cette variable peut être obtenue de manière itérative :

1. Initialisation, $t = 0$ et $0 \leq j \leq N-1$:

$$\begin{aligned} \alpha_0(0) &= 1.0 \\ \alpha_0(j) &= \sum_{i=0}^{N-1} a'_{ij\Phi}\alpha_0(i) \end{aligned} \tag{1.10}$$

2. Induction, $1 \leq t \leq T$ et $0 \leq j \leq N-1$:

$$\alpha_t(j) = \sum_{i=0}^{N-1} \left[a_{ijo_{t-1}}\alpha_{t-1}(i) + a'_{ij\Phi}\alpha_t(i)\right] \tag{1.11}$$

3. Terminaison, $t = T$

$$\Pr(\mathcal{O}|\Lambda) = \alpha_T(N-1) \tag{1.12}$$

Le formalisme ci-dessus considère qu'il existe un état de départ et un de fin, respectivement s_0 et s_{N-1}. L'étape d'induction explique comment l'état s_j peut être atteint à l'instant t à partir des N états possibles s_i atteint à l'instant $t-1$. De manière similaire nous pouvons définir la variable *backward* par :

$$\beta_t(i) = \Pr(o_t o_{t+1} \ldots o_{T-1} | q_t = s_i, \Lambda) \tag{1.13}$$

c'est-à-dire la probabilité d'émission de la séquence partielle $o_t o_{t+1} \ldots o_{T-1}$ étant donné le modèle Λ et qu'à l'instant t le modèle est dans l'état s_i que l'on emprunte une transition nulle ou non. Cette variable peut également être calculée de manière itérative :

1. Initialisation, $t = T$ et $0 \leq j \leq N-1$:

$$\begin{aligned} \beta_T(N-1) &= 1.0 \\ \beta_T(i) &= \sum_{j=0}^{N-1} a'_{ij\Phi} \beta_T(i) \end{aligned} \tag{1.14}$$

2. Induction, $T-1 \leq t \leq 0$ et $0 \leq j \leq N-1$:

$$\beta_t(i) = \sum_{j=0}^{N-1} \left[a_{ijo_t} \beta_{t+1}(j) + a'_{ij\Phi} \beta_t(j) \right] \tag{1.15}$$

3. Terminaison, $t = 0$:

$$\Pr(\mathcal{O}|\Lambda) = \beta_0(0) \tag{1.16}$$

À partir des variables $\alpha_t(i)$ et $\beta_t(i)$, nous pouvons obtenir également la vraisemblance de la séquence $\mathcal{O}$ par le modèle Λ à partir de n'importe quel instant t :

$$\Pr(\mathcal{O}|\Lambda) = \sum_{i=0}^{N-1} \alpha_t(i) \times \beta_t(i) \tag{1.17}$$

La fonction *forward* permet de calculer la somme des probabilités de tous les chemins possibles et réduit considérablement le nombre de calculs (de $2TN^T$ à TN^2).

1.1.3.2 Le problème de décodage : l'algorithme de Viterbi

Ce problème est généralement résolu par l'intermédiaire de l'*algorithme de Viterbi* [42, 135]. C'est une approximation de la procédure *Forward* qui calcule la probabilité du meilleur chemin, au lieu de la somme de l'ensemble des chemins. La recherche est décomposée en une succession d'optimisations locales, principe de base des approches de programmation dynamique. Nous cherchons donc le meilleur chemin caractérisé par la séquence d'états $\mathcal{Q} = q_0 q_1 \cdots q_T$, qui peut produire la séquence d'observations $\mathcal{O} = o_0 o_1 \cdots o_{T-1}$. Pour cela définissons la variable $\delta_t(i)$:

$$\delta_t(i) = \max_{q_0, q_1, \ldots, q_{t-1}} \Pr(q_0 q_1 \cdots q_t = s_i, o_0 o_1 \cdots o_{t-1} | \Lambda) \tag{1.18}$$

C'est-à-dire la probabilité du meilleur chemin aboutissant à l'état s_i à l'instant t, étant donné notre modèle Λ. Lors du déroulement de l'algorithme, nous devons conserver la trace du meilleur chemin. Dans ce but une fonction particulière est utilisée $\Psi_t(i)$ qui conserve l'état correspondant à la meilleure probabilité $\delta_t(i)$ à chaque instant t. La procédure de Viterbi peut être décrite de la manière suivante :

1. Initialisation, $t = 0$ et $0 \leq j \leq N-1$:

$$\begin{aligned} \delta_0(0) &= 1 \\ \Psi_0(0) &= 0 \\ \delta_0(j) &= \max_{0 \leq i \leq N-1} [a'_{ij\Phi} \delta_0(i)] \\ \Psi_0(j) &= \arg\max_{0 \leq i \leq N-1} [a'_{ij\Phi} \delta_0(i)] \end{aligned} \tag{1.19}$$

2. **Induction**, $1 \leq t \leq T$ et $0 \leq j \leq N-1$:

$$\delta_t(j) = \max_{0 \leq i \leq N-1} \left[a_{ij o_{t-1}} \delta_{t-1}(i) \, , \, a'_{ij\Phi} \delta_t(i) \right]$$

$$\Psi_t(j) = \begin{cases} \arg\max_{0 \leq i \leq N-1} \left[a'_{ij\Phi} \delta_t(i) \right] & \text{si la transition de } s_i \text{ à } s_j \text{ est nulle} \\ \arg\max_{0 \leq i \leq N-1} \left[a_{ij o_{t-1}} \delta_{t-1}(i) \right] & \text{sinon} \end{cases} \tag{1.20}$$

$$\zeta_t(j) = \begin{cases} 1 & \text{si la transition de } s_i \text{ à } s_j \text{ est nulle} \\ 0 & \text{sinon} \end{cases}$$

3. **Terminaison**, $t = T$:

$$\begin{aligned} P^* &= \delta_T(N-1) \\ q_T^* &= N-1 \end{aligned} \tag{1.21}$$

4. **Recherche arrière du meilleur chemin (*backtracking*)**, $T-1 \leq t \leq 0$:

$$q_t^* = \Psi_{t+1}(q_{t+1}^*) \tag{1.22}$$

Après l'estimation de q_t^* nous devons vérifier la présence de transitions nulles par l'intermédiaire de la variable ζ_t. Pour cela une nouvelle variable est définie q'_t :

$$q'_t = \Psi_t(q_t^*) \quad \text{si} \quad \zeta_t(q_t^*) = 1 \tag{1.23}$$

La présence d'une transition nulle lors de la procédure de *backtracking* conduit alors à modifier légèrement la variable q_t^* :

$$q_t^* = \Psi_{t+1}(q'_{t+1}) \quad \text{si} \quad \zeta_{t+1}(q_{t+1}^*) = 1 \tag{1.24}$$

Finalement la séquence d'états correspondant à la plus forte probabilité $\mathcal{Q}^*$ est obtenue de la manière suivante :

$$\mathcal{Q}^* = q^*_{T-1}q'_{T-1}q^*_{T-2}q'_{T-2}\cdots q^*_1 q'_1 q^*_0 q'_0 \tag{1.25}$$

Cet algorithme est similaire à la procédure *Forward*. La seule différence se situe au niveau de l'induction, où l'algorithme de Viterbi effectue une maximisation au lieu d'une somme. Du point de vue implémentation il est à noter qu'il est possible de passer en logarithme et ainsi effectuer des additions de probabilités au lieu de multiplications. Le passage au logarithme des probabilités est également réalisé pour une raison de représentation numérique. En effet, la multiplication d'un grand nombre de probabilités conduit à des valeurs numériques faibles, atteignant parfois la précision de certaine machine.

1.1.3.3 Le problème d'apprentissage : l'algorithme de Baum-Welch

Le dernier problème associé aux MMCs est celui de l'estimation des paramètres à partir d'un ensemble de séquences d'observations. Il peut être réalisé par l'intermédiaire de l'*algorithme de Baum-Welch* [11, 7, 135]. Il s'agit d'une adaptation de l'algorithme EM (*Expectation-Maximization*) [29] qui garantit la convergence vers un maximum local de la probabilité d'observation de l'ensemble des exemples d'apprentissage, au sens du critère de *maximum de vraisemblance* (MLE). Afin de décrire cet algorithme, nous allons définir deux nouvelles variables :

- $\xi^1_t(i,j) = \Pr(q_t = s_i, q_{t+1} = s_j|\mathcal{O}, \Lambda)$, la probabilité d'être à l'état s_i à l'instant t et à l'état s_j à l'instant $t+1$ tout en émettant une observation réelle o_t, étant donnés la séquence d'observations $\mathcal{O}$ et le modèle Λ,
- $\xi^2_t(i,j) = \Pr(q_t = s_i, q_t = s_j|\mathcal{O}, \Lambda)$, la probabilité d'être à l'état s_i à l'instant t et à l'état s_j à l'instant t tout en émettant l'observation nulle Φ, étant donnés la séquence d'observations $\mathcal{O}$ et le modèle Λ.

Après développement, ces quantités peuvent être exprimées en fonction des variables *forward* et *backward* (voir les équations 1.9 et 1.13) de la manière suivante :

$$\xi_t^1(i,j) = \frac{\alpha_t(i) a_{ijo_t} \beta_{t+1}(j)}{\Pr(\mathcal{O}|\Lambda)} \tag{1.26}$$

$$\xi_t^2(i,j) = \frac{\alpha_t(i) a'_{ij\Phi} \beta_{t+1}(j)}{\Pr(\mathcal{O}|\Lambda)} \tag{1.27}$$

Une autre variable est nécessaire pour le développement de l'algorithme. Il s'agit de la probabilité d'être à l'état s_i à l'instant t, étant donnés la séquence d'observations $\mathcal{O}$ et le modèle Λ :

$$\gamma_t(i) = \Pr(q_t = s_i|\mathcal{O}, \Lambda) \tag{1.28}$$

Cette dernière est simplement reliée aux variables $\xi_t^1(i,j)$ et $\xi_t^2(i,j)$ par la relation :

$$\gamma_t(i) = \sum_{j=0}^{N-1} \left[\xi_t^1(i,j) + \xi_t^2(i,j)\right] = \frac{\alpha_t(i)\beta_t(i)}{\Pr(\mathcal{O}|\Lambda)} \tag{1.29}$$

Maintenant si nous effectuons une somme de la variable $\gamma_t(i)$ pour tous les instants t, nous obtenons une quantité qui peut être interprétée comme le nombre de fois que l'état s_i a été visité. De même, la somme sur t de la variable $\xi_t^1(i,j)$ peut être interprétée comme le nombre de fois que la transition $s_i \rightarrow s_j$ a été franchie.

Une manière raisonnable d'estimer les paramètres a_{ijk} et $a_{ij\Phi}$ du modèle est donnée par les relations suivantes :

$$\bar{a}_{ijk} = \frac{\text{nombre estimé de transitions de } s_i \text{ vers } s_j \text{ en émettant le symbole } v_k}{\text{nombre estimé de visite de l'état } s_i} \tag{1.30}$$

$$\bar{a}'_{ij\Phi} = \frac{\text{nombre estimé de transitions de } s_i \text{ vers } s_j \text{ en émettant le symbole } \Phi}{\text{nombre estimé de visite de l'état } s_i} \tag{1.31}$$

En utilisant les interprétations des variables $\gamma_t(i)$ et $\xi_t^1(i,j)$ présentées ci-dessus, nous pouvons exprimer ces relations en fonction des différentes variables :

$$\bar{a}_{ijk} = \frac{\sum_{t=0}^{T} \delta(o_t, v_k)\xi_t^1(i,j)}{\sum_{t=0}^{T} \gamma_t(i)} = \frac{\sum_{t=0}^{T} \delta(o_t, v_k)\alpha_t(i)a_{ijo_t}\beta_{t+1}(j)}{\sum_{t=0}^{T} \alpha_t(i)\beta_t(i)} \tag{1.32}$$

$$\bar{a'}_{ij\Phi} = \frac{\sum_{t=0}^{T} \xi_t^2(i,j)}{\sum_{t=0}^{T} \gamma_t(i)} = \frac{\sum_{t=0}^{T} \alpha_t(i)a'_{ij\Phi}\beta_{t+1}(j)}{\sum_{t=0}^{T} \alpha_t(i)\beta_t(i)} \tag{1.33}$$

où la fonction $\delta(x,y)$ est définie de la manière suivante :

$$\delta(x,y) = \begin{cases} 1 & \text{si } x = y \\ 0 & \text{sinon} \end{cases} \tag{1.34}$$

permet de prendre en compte les différentes observations possibles.

Rabiner [135] mentionne le fait que l'apprentissage du modèle Λ doit se faire à l'aide d'un grand nombre de séquences d'observations, ceci de manière à obtenir une estimation fiable des différents paramètres. Cette considération conduit à une légère modification des formules de ré-estimation. Considérons un corpus de L exemples : $\mathbf{O} = [\mathcal{O}^0, \mathcal{O}^1, \ldots, \mathcal{O}^{L-1}]$. Nous assumons que chacune des séquences d'observations est indépendante. Le but de l'apprentissage consiste à ajuster les paramètres du modèle Λ de manière à maximiser :

$$\Pr(\mathbf{O}|\Lambda) = \prod_{l=0}^{L-1} \Pr(\mathcal{O}^l|\Lambda) \tag{1.35}$$

Comme les ré-estimations sont basées sur les fréquences d'occurrence des différents évènements, les formules de ré-estimations sont modifiées en prenant en compte les fré-

quences individuelles de chaque séquence :

$$\bar{a}_{ijk} = \frac{\sum_{l=0}^{L-1}\left[\frac{1}{\Pr(\mathcal{O}^l|\Lambda)}\sum_{t=0}^{T}\delta(o_t^l, v_k)\alpha_t^l(i)a_{ijo_t^l}\beta_{t+1}^l(j)\right]}{\sum_{l=0}^{L-1}\left[\frac{1}{\Pr(\mathcal{O}^l|\Lambda)}\sum_{t=0}^{T}\alpha_t^l(i)\beta_t^l(i)\right]} \tag{1.36}$$

$$\bar{a'}_{ij\Phi} = \frac{\sum_{l=0}^{L-1}\left[\frac{1}{\Pr(\mathcal{O}^l|\Lambda)}\sum_{t=0}^{T}\alpha_t^l(i)a'_{ij\Phi}\beta_{t+1}^l(j)\right]}{\sum_{l=0}^{L-1}\left[\frac{1}{\Pr(\mathcal{O}^l|\Lambda)}\sum_{t=0}^{T}\alpha_t^l(i)\beta_t^l(i)\right]} \tag{1.37}$$

Cette normalisation permet également de ne pas favoriser les séquences d'observations de forte probabilité.

La procédure d'apprentissage consiste, à partir du modèle de départ Λ_0, initialisé aléatoirement, de ré-estimer les paramètres à l'aide des équations 1.36 et 1.37, de manière itérative, jusqu'à l'obtention du critère d'arrêt.

1.1.4 L'interpolation de modèles

Lors de la modélisation statistique d'un phénomène réel, les évènements rares ne sont pas correctement pris en compte à cause du manque d'exemples. Dans [7] les auteurs présentent une technique d'interpolation permettant d'outrepasser ce problème. Elle est également exposée dans [135].

Le principe consiste à construire, à partir d'un modèle de base Λ_1, un second modèle Λ_2 de même structure mais pour lequel certains paramètres sont *liés*. Deux paramètres sont dits liés s'il existe une relation d'équivalence entre eux. Cette technique permet de réduire le nombre de paramètres indépendants du modèle de base. De ce fait le modèle sera moins précis, mais disposant de plus de données pour l'apprentissage, l'estimation des différents paramètres sera plus fiable. L'obtention des paramètres finaux est alors obtenue à l'aide d'un troisième modèle Λ, résultant de l'interpolation des deux premiers. Considérons un

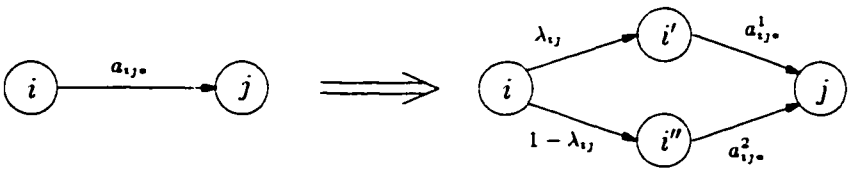


FIGURE 6 Représentation graphique de l'interpolation de paramètres des modèles de Markov cachés.

paramètre du premier modèle : $a^1_{ijk} = \Pr(o_t = v_k, q_{t+1} = s_j | q_t = s_i, \Lambda_1)$ et un du second : $a^2_{ijk} = \Pr(o_t = v_k, q_{t+1} = s_j | q_t = s_i, \Lambda_2)$, l'estimation des paramètres finaux se fera de la manière suivante :

$$a_{ijk} = \lambda_{ij} \times a^1_{ijk} + (1 - \lambda_{ij}) \times a^2_{ijk} \tag{1.38}$$

Ce formalisme peut être interprété comme la substitution d'une transition entre deux états par quatre transitions et deux états intermédiaires, comme présenté sur la figure 6. L'introduction des paramètres d'interpolation λ_{ij} permettent au système de choisir localement la meilleure estimation des différents paramètres. Afin d'estimer les paramètres d'interpolation, il faut réaliser un nouvel apprentissage. Pour cela les paramètres des deux premiers modèles Λ_1 et Λ_2 sont utilisés directement et l'apprentissage permet d'estimer les valeurs des λ_{ij} uniquement. Du fait que ces nouveaux paramètres sont utilisés afin de prédire lequel des deux modèles est le meilleur lors de l'observation de nouvelles données, un nouveau corpus est nécessaire afin d'obtenir une estimation correcte. Une autre alternative consiste à utiliser une technique de partition du corpus d'apprentissage appelée : *deleted interpolation* [69, 135].

Dans cette section, nous avons donné une définition des modèles de Markov cachés et présenté les algorithmes classiques utilisés pour les mettre en œuvre. Le formalisme présenté est celui correspondant aux modèles utilisés au cours du développement de notre projet. Cette section est loin de couvrir l'ensemble des possibilités offertes par la modélisation

markovienne. Le lecteur peut se reporter aux différentes références citées pour approfondir ce domaine.

1.2 La théorie de l'information

Claude E. Shannon est considéré comme le père de la théorie de l'information avec son article publié, en deux parties, en 1948 [141, 142]. Son but était de présenter un formalisme mathématique à la théorie générale de la communication. Le travail ou l'œuvre de Shannon a influencé un grand nombre de personnes. De plus nous pouvons dire qu'il est à la source des technologies de communication modernes, en plein essor en ce début de millénaire. Pour se rendre compte de l'impact de son travail, il est intéressant de lire l'article de Gallager publié en 2001 [50].

La théorie de l'information est basée sur le paradigme de Shannon qui est représenté par le schéma de la figure 7. La source et le destinataire sont deux entités séparées. Elle peuvent communiquer par l'intermédiaire du canal et d'appareillage d'émission et de réception du signal d'information. Le canal est le siège de phénomènes de propagation d'une part et de phénomènes perturbateurs d'autre part. À cause de ces derniers, l'excitation appliquée par la source ne suffit pas à déterminer la réponse du canal. Le destinataire quant à lui n'a que la réponse du canal pour percevoir le message émis. Ce schéma peut s'appliquer à tout type de communication : la source est par exemple une personne qui parle et le destinataire est une personne qui écoute, le canal est alors l'air ambiant, ou encore la ligne téléphonique reliant deux appareils téléphoniques. La source et le destinataire peuvent être distants dans l'espace, mais également dans le temps. Dans ce cas le canal serait un support magnétique. Il peut s'agir également d'un scripteur et d'un lecteur, le canal est alors une feuille de papier.

La théorie de l'information étudie la transmission de l'information ainsi que sa dégradation. Pour cela la source est considérée comme le siège d'événements aléatoires conduisant à l'émission du message. Le but de la théorie de l'information est de caractériser la

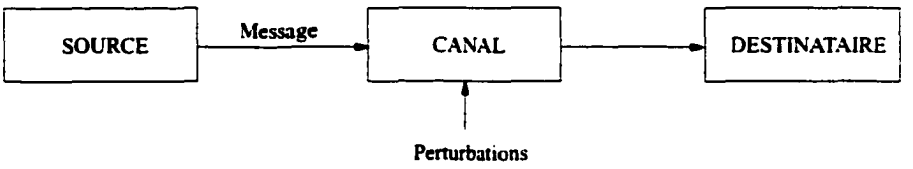


FIGURE 7 Schéma fondamental d'une communication : paradigme de Shannon.

source et le canal. Pour cela deux concepts principaux ont été formulés ; il s'agit de la quantité d'information moyenne produite par la source et de la capacité du canal. Cette dernière permet de caractériser la quantité d'information moyenne que la connaissance d'un message reçu en sortie du canal apporte sur le message émis à son entrée.

C'est du point de vue de la classification que ces concepts nous intéressent. Pour le domaine de la reconnaissance de formes, le canal peut-être considéré comme l'espace de représentation choisi pour décrire le signal envoyé par la source. En effet, quelle que soit l'application choisie, il est rare que le signal à reconnaître soit directement transmis au classificateur. Généralement une étape d'extraction est réalisée. Elle a pour but de compresser l'information en supprimant la redondance et les composantes non pertinentes au classificateur pour réaliser sa tâche. Donc après une étape de pré-traitements, le signal de départ est projeté dans un "espace d'information". Il est certain que cet espace est choisi par l'expérimentateur parmi un grand nombre de possibilités. C'est la pertinence de ce support de l'information que nous désirons quantifier par l'intermédiaire des mesures de la théorie de l'information.

1.2.1 Une mesure quantitative de l'information

Le premier concept de la théorie de l'information est celui de mesure quantitative de l'information. Rappelons que la source est considérée comme aléatoire, ce qui induit que le destinataire ne peut faire que des hypothèses quant au message émis et choisir la plus vraisemblable. Cette remarque conduit à mettre en relation la mesure de la quantité d'information avec la mesure de l'inattendu et de l'improbable. La quantité d'information

$h(x)$ apportée par la réalisation d'un événement x de probabilité $\Pr(x)$ est mesurée par une fonction croissante de son improbabilité $1/\Pr(x)$. Le choix de cette fonction a été guidé par le respect de certaines propriétés :

- un événement certain apportera une quantité d'information nulle, soit $f(1) = 0$,
- la réalisation de deux événements indépendants x et y apportera la somme de leurs quantités individuelles d'information, soit $h(x, y) = h(x) + h(y)$.

Ces propriétés ont conduit à choisir la fonction logarithme. La quantité d'information associée à la réalisation d'un événement x est définie par :

$$h(x) = \log\left[\frac{1}{\Pr(x)}\right] = -\log \Pr(x) \tag{1.39}$$

Le choix de la base du logarithme définit l'unité d'information. Shannon a proposé de prendre la base égale à 2 et de nommer l'unité le "bit" pour *binary unit*. Comme cette unité est également utilisée comme abréviation de chiffre binaire, l'ISO a proposé d'utiliser le "shannon", dont l'abréviation est Sh, comme unité binaire d'information. Si le logarithme est népérien, l'unité est le "nat" et si il est décimal le "dit" ou "Hartley".

Considérons maintenant deux événements x et y. On peut associer à leur couple (x, y) la quantité d'information :

$$h(x, y) = -\log \Pr(x, y) \tag{1.40}$$

où $\Pr(x, y)$ désigne la probabilité conjointe des deux événements. On peut également mesurer la quantité d'information associée à x conditionnellement à la réalisation de y :

$$h(x|y) = -\log \Pr(x|y) \tag{1.41}$$

où $\Pr(x|y)$ est la probabilité conditionnelle de x étant donné y.

À partir de la règle de Bayes :

$$\Pr(x,y) \;=\; \Pr(x|y)\Pr(y) \;=\; \Pr(y|x)\Pr(x) \tag{1.42}$$

on en déduit :

$$h(x,y) \;=\; h(x|y) + h(y) \;=\; h(y|x) + h(x) \tag{1.43}$$

Dans le cas où les deux événements x et y sont indépendants on a : $h(x|y) = h(x)$ et donc on retrouve la condition d'additivité de la quantité d'information mentionnée plus haut : $h(x,y) = h(x) + h(y)$.

Une autre mesure intéressante est la quantité d'information qu'une donnée, y par exemple, apporte sur l'autre, x en l'occurrence. Dans le cas où l'on identifie x au signal appliqué à l'entrée du canal et y à celui observé en sortie, $\Pr(x)$ correspond à la probabilité *a priori* que x soit émis et $\Pr(x|y)$ à la probabilité *a posteriori* que x ait été émis, sachant que y a été reçu. Cette mesure est obtenue de la manière suivante :

$$i(x;y) \;=\; \log\left[\frac{\Pr(x|y)}{\Pr(x)}\right] \tag{1.44}$$

La quantité $i(x;y)$ mesure l'accroissement de la probabilité de x que l'observation de y apporte. À partir de la règle de Bayes (équation 1.42), cette quantité peut s'écrire :

$$i(x;y) \;=\; \log\left[\frac{\Pr(x,y)}{\Pr(x)\Pr(y)}\right] \;=\; i(y;x) \tag{1.45}$$

Cette quantité est appelée information mutuelle par opposition à $h(x)$ qui est appelée information propre de x.

1.2.2 Entropie et information mutuelle moyenne

Nous venons de voir des mesures d'informations d'événements individuels. Nous allons maintenant nous intéresser aux estimations moyennes de tels événements. Considérons une source discrète, finie et stationnaire, les événements x_i peuvent alors être interprétés comme le choix d'un symbole parmi un alphabet $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$. Supposons de plus que les événements successifs sont mutuellement indépendants. Chaque émission de la source est alors décrite par une variable aléatoire X prenant ses valeurs dans l'alphabet considéré. À chacun de ces symboles est associée une probabilité :

$$p_i = \Pr(X = x_i), \quad i = 1, 2, \ldots, N. \qquad \text{avec} \quad \sum_{i=1}^{N} p_i = 1$$

La quantité d'information moyenne associée à chaque symbole est la moyenne de l'information propre de chacun des événements $X = x_i$, c'est-à-dire :

$$H(X) \;=\; E[h(X)] \;=\; \sum_{i=1}^{N} p_i \log\left[\frac{1}{p_i}\right] \;=\; -\sum_{i=1}^{N} p_i \log p_i \tag{1.46}$$

où $E[\cdot]$ dénote l'espérance mathématique. Cette quantité est également appelée entropie de la source. La variable aléatoire X n'étant pas l'argument réel de l'entropie, ce sont les probabilités associées aux différents événements possibles, une autre notation de cette fonction est utilisée : $H(p_1, p_2, \ldots, p_N)$. Une des propriétés de l'entropie est qu'elle est non négative : $H(p_1, p_2, \ldots, p_N) \geq 0$. L'égalité intervenant si une des probabilités p_i est égale à 1 et donc toutes les autres égale à 0. Ceci correspond au fait qu'un des événements est certain et que sa réalisation n'apporte alors aucune information. Par contre l'entropie atteint sa valeur maximum lorsque tous les événements sont équiprobables et donc égaux à $1/N$. Elle vaut alors :

$$H(p_1, p_2, \ldots, p_N) \;\leq\; \log N \tag{1.47}$$

La fonction entropie $H(X)$ est une fonction concave de X. Une autre de ses propriétés est d'être croissante avec l'augmentation du nombre de partitions de l'espace de probabilité associé à la variable aléatoire considérée : soit deux distributions de probabilités $\{p_1, p_2, \ldots, p_N\}$ et $\{q_1, q_2, \ldots, q_M\}$ d'une même variable aléatoire, avec $N \leq M$, on a :

$$H(p_1, p_2, \ldots, p_N) \leq H(q_1, q_2, \ldots, q_M) \tag{1.48}$$

Considérons maintenant deux variables aléatoires X et Y prenant leur valeurs dans deux alphabets distincts : $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$ et $\mathcal{Y} = \{y_1, y_2, \ldots, y_M\}$, on obtient alors l'entropie conjointe moyenne à partir de la définition précédente :

$$H(X,Y) = E[h(X,Y)] = -\sum_{i=1}^{N}\sum_{j=1}^{M} \Pr(x_i, y_j) \log \Pr(x_i, y_j) \tag{1.49}$$

De la même manière nous obtenons l'entropie conditionnelle d'une variable aléatoire X étant donnée la seconde Y :

$$H(X|Y) = E[h(X|Y)] = -\sum_{i=1}^{N}\sum_{j=1}^{M} \Pr(x_i, y_j) \log \Pr(x_i|y_j) \tag{1.50}$$

Il est à noter que dans cette dernière équation, la probabilité en argument du logarithme est une probabilité conditionnelle. Elle est obtenue à partir des probabilités conjointes de x_i et y_j et de la probabilité marginale de y_j :

$$\Pr(x_i|y_j) = \frac{Pr(x_i, y_j)}{\Pr(y_j)} \quad \text{avec} \quad \Pr(y_j) = \sum_{i=1}^{N} \Pr(x_i, y_j) \tag{1.51}$$

Pour un couple de variables, une autre mesure peut être obtenue, il s'agit de l'information mutuelle moyenne, définie de la manière suivante :

$$I(X;Y) = E[i(X;Y)] = -\sum_{i=1}^{N}\sum_{j=1}^{M} \Pr(x_i, y_j) \log\left[\frac{\Pr(x_i, y_j)}{\Pr(x_i)\Pr(y_j)}\right] \tag{1.52}$$

où $\Pr(x_j) = \sum_{j=1}^{M} \Pr(x_i, y_j)$ et $\Pr(y_j) = \sum_{i=1}^{N} \Pr(x_i, y_j)$

L'information mutuelle est bien sûr non-négative comme l'entropie. De plus l'information mutuelle est symétrique $I(X;Y) = I(Y;X)$, ceci peut être vérifié directement à partir de sa définition ci-dessus.

À partir de la définition de l'entropie conjointe $H(X,Y)$ (équation 1.49) et de la règle de Bayes (équation 1.42), nous obtenons directement deux relations entre les différentes entropies :

$$H(X,Y) = H(X) + H(Y|X) \tag{1.53}$$

$$= H(Y) + H(X|Y) \tag{1.54}$$

Elles caractérisent le fait que la quantité d'information apportée conjointement par X et Y est supérieure ou égale à celle apportée par X ou Y. Ceci est dû au fait que l'entropie est non négative et peut se traduire également par :

$$H(X,Y) \geq \max[H(X), H(Y)] \tag{1.55}$$

L'entropie conditionnelle est quant à elle bornée par l'entropie propre de la variable considérée de la manière suivante :

$$H(X|Y) \leq H(X) \tag{1.56}$$

L'égalité étant vérifiée si X et Y sont indépendantes. Cette équation traduit le fait que la réalisation d'une variable aléatoire en conditionnant une seconde ne peut que diminuer la quantité d'information apportée par cette dernière. L'équation 1.56 permet alors d'écrire :

$$H(X,Y) \leq H(X) + H(Y) \leq 2H(X,Y) \tag{1.57}$$

Il existe également un certain nombre de relations entre les différentes entropies et l'information mutuelle :

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) && (1.58)\\ &= H(Y) - H(Y|X) && (1.59)\\ &= H(X) + H(Y) - H(X,Y) && (1.60) \end{aligned}$$

Les deux premières équations traduisent le fait que l'information mutuelle entre deux variables est égale à l'apport d'information de la première moins la quantité d'information apportée par cette même variable lorsqu'elle est conditionnée par la seconde. Comme nous l'avons mentionné ci-dessus, l'information mutuelle est positive ou nulle. Les équations 1.56 et 1.58 permettent de vérifier l'inégalité $I(X;Y) \geq 0$, l'égalité étant vraie lorsque les variables sont indépendantes. À partir de ces mêmes équations, nous pouvons déterminer la borne supérieure de l'information mutuelle :

$$I(X;Y) \leq \min\left[H(X), H(Y)\right] \tag{1.61}$$

Elle est bornée par les valeurs d'entropie propre des variables aléatoires X et Y.

Le diagramme de Venn présenté sur la figure 8 symbolise les relations entre les différentes entropies et l'information mutuelle. Il permet également de visualiser les bornes de l'information mutuelle.

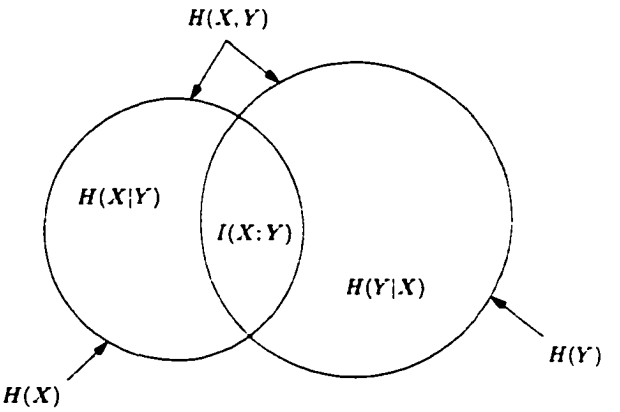


FIGURE 8 Diagramme de Venn caractérisant les relations entre les différents indicateurs entropiques.

1.2.3 Application à la classification

Dans un système de reconnaissance de formes intervient une étape de classification. Une forme est caractérisée par deux attributs : une série de mesures et un sens ou un message. La classification consiste à inférer le sens à partir des mesures, c'est-à-dire établir la relation entre l'espace de représentation des formes et l'espace des sens possibles. L'espace de représentation choisi a donc une importance non négligeable dans la détermination de cette relation. Il est alors judicieux d'évaluer sa qualité et pour cela quantifier l'information qu'il apporte. La quantité la plus intéressante est bien sûr la mesure de l'information apportée par l'espace de représentation sur le sens de la forme à reconnaître.

Nous allons maintenant poser le formalisme associé à la classification. Une forme à reconnaître est caractérisée par un vecteur $\mathbf{x} = [x_1, x_2, \ldots, x_N]^t$ dans un espace de représentation à N dimensions. Dans le cas discret, on associe à ce vecteur une primitive f_i appartenant à un alphabet fini $\mathcal{F} = \{f_1, f_2, \ldots, f_{N_F}\}$, encore appelé ensemble de primitives. L'extraction de primitives peut être vu comme une application associant à chaque forme en entrée du système de reconnaissance une primitive unique appartenant à l'ensemble $\mathcal{F}$. La classification, quant à elle, peut être considérée comme une application

de cet ensemble de primitives vers un ensemble de classes $\mathcal{C} = \{\omega_1, \omega_2, \ldots, \omega_{N_C}\}$. Le nombre de ces classes N_C dépend de la modélisation choisie par le concepteur du système de reconnaissance.

À partir de cette définition, deux variables aléatoires peuvent alors être associées à un système de reconnaissance. L'extraction de primitives peut être vue comme la réalisation d'une variable aléatoire F, qui prend ses valeurs dans un alphabet fini $\mathcal{F}$. Le résultat de la classification peut également être vu comme la réalisation d'une autre variable aléatoire C. À l'aide de ces deux variables aléatoires, plusieurs mesures d'information peuvent être réalisées. Dans un premier temps nous pouvons obtenir l'entropie de l'ensemble de primitives $H(F)$, conformément à l'équation 1.46, de la manière suivante :

$$H(F) = -\sum_{i=1}^{N_F} \Pr(f_i) \log \Pr(f_i) \tag{1.62}$$

Cette fonction mesure la quantité d'information moyenne apportée pour chaque primitive de l'ensemble $\mathcal{F}$. Elle peut être obtenue avant la classification puisqu'elle ne fait appel qu'aux probabilités des primitives.

De la même manière nous pouvons obtenir une mesure de la quantité d'information des classes :

$$H(C) = -\sum_{i=1}^{N_C} \Pr(\omega_i) \log \Pr(\omega_i) \tag{1.63}$$

Cette mesure est utilisée pour quantifier la difficulté d'une tâche de reconnaissance. Comme nous l'avons vu dans la section précédente, l'entropie atteint son maximum lorsque toutes les probabilités, $\Pr(\omega_i)$ en l'occurrence, sont égales. Cela signifie que toutes les classes de notre modélisation sont équiprobables. Par contre si une seule des classes est probable : $\Pr(\omega_i) = 1$ et $\Pr(\omega_j) = 0 \quad \forall j \neq i$, l'entropie est nulle et le problème de classification est résolu. L'entropie permet donc bien de mesurer la difficulté d'une tâche de reconnais-

sance. Ce concept a été introduit par Bahl [7] afin de mesurer la difficulté d'une tâche de reconnaissance de la parole. En fait il n'utilise pas l'entropie mais plutôt la perplexité PP. Ces deux mesures sont directement reliées par la relation :

$$PP(C) = 2^{H(C)} = 2^{-\sum_{i=1}^{N_C} \Pr(\omega_i) \log \Pr(\omega_i)} \tag{1.64}$$

Cette relation est bien sûr valable uniquement si la base du logarithme est 2. Dans le cas du logarithme népérien on utiliserait la fonction exponentielle $e^{(\cdot)}$. La perplexité est une grandeur sans dimension qui varie entre 1 et la taille de l'alphabet dans lequel la variable aléatoire prends ses valeurs, N_C en l'occurrence. Cette mesure permet de quantifier objectivement la difficulté de la tâche de reconnaissance.

Une mesure d'information relative aux deux variables aléatoires est l'entropie conditionnelle des classes étant données les primitives. Nous pouvons obtenir un indicateur pour chaque primitive de l'ensemble $\mathcal{F}$ de la manière suivante :

$$H(C|F = f_j) = -\sum_{i=1}^{N_C} \Pr(\omega_i|f_j) \log \Pr(\omega_i|f_j) \tag{1.65}$$

L'évaluation de l'ensemble $\mathcal{F}$ globalement est alors obtenu de la manière suivante :

$$H(C|F) = -\sum_{j=1}^{N_F} \Pr(f_j) H(C|F = f_j) \tag{1.66}$$

$$= -\sum_{j=1}^{N_F} \sum_{i=1}^{N_C} \Pr(f_j) \Pr(\omega_i|f_j) \log \Pr(\omega_i|f_j) \tag{1.67}$$

$$= -\sum_{j=1}^{N_F} \sum_{i=1}^{N_C} \Pr(\omega_i, f_j) \log \Pr(\omega_i|f_j) \tag{1.68}$$

En utilisant la relation entre perplexité et entropie (équation 1.64) nous pouvons obtenir les quantités $PP(C|F = f_j)$ et $PP(C|F)$. Ces deux dernières mesures sont directement reliées au nombre de classes associées au problème. La perplexité conditionnelle des classes étant donnée une primitive a été proposée par El-Yacoubi pour évaluer objectivement le pouvoir discriminant d'une primitive [35]. En effet, si une primitive est toujours utilisée pour caractériser une même classe : $\{f_j \mid \Pr(\omega_i|f_j) = 1, \Pr(\omega_k|f_j) = 0 \quad \forall k \neq i\}$, cette primitive permet d'obtenir directement la classe associée, elle est donc très discriminante du point de vue de la classification. Par contre si les probabilités $\Pr(\omega_i|f_j)$ sont équiprobables, la primitive ne permet pas d'effectuer la discrimination entre les classes et son pouvoir discriminant est faible. $H(C|F)$ permet de quantifier la variabilité de l'ensemble des classes $\mathcal{C}$ lorsque l'ensemble de primitives $\mathcal{F}$ est connu.

Du point de vue de la classification, les quantités $H(F|C)$ et $H(C, F)$ ne sont pas très intéressantes. Par contre l'information mutuelle moyenne, qui mesure la quantité d'information que l'une des deux variables aléatoires apporte sur l'autre, peut être utilisée pour évaluer un ensemble de primitives. On obtient cet indicateur de la manière suivante :

$$I(C;F) = \sum_{i=1}^{N_C} \sum_{j=1}^{N_F} \Pr(\omega_i, f_j) \log \left[\frac{\Pr(\omega_i, f_j)}{\Pr(\omega_i) \Pr(f_j)} \right] \tag{1.69}$$

Cette quantité permet de mesurer également le degré de dépendance entre les deux variables aléatoires. En effet si les deux variables sont indépendantes la probabilité $\Pr(\omega_i, f_j)$ est alors égale à $\Pr(\omega_i) \Pr(f_j)$ et donc $I(C; F) = 0$. Nous pouvons également obtenir une mesure individuelle de l'information mutuelle. À l'aide de la règle de Bayes (équation 1.42), l'information mutuelle entre les deux variables peut s'écrire :

$$I(C;F) = \sum_{j=1}^{N_F} \Pr(f_j) \sum_{i=1}^{N_C} \Pr(\omega_i|f_j) \log \left[\frac{\Pr(\omega_i|f_j)}{\Pr(\omega_i)} \right] \tag{1.70}$$

L'évaluation de l'information mutuelle relative à une seule primitive peut alors s'obtenir de la manière suivante :

$$I(C; F = f_j) = \sum_{i=1}^{N_C} \Pr(\omega_i|f_j) \log \left[\frac{\Pr(\omega_i|f_j)}{\Pr(\omega_i)} \right] \tag{1.71}$$

Pour plus d'information sur le domaine de la théorie de l'information, le lecteur peut se reporter au livre de Cover et Thomas [26], ou en français à celui de Battail [10].

1.2.4 Mesure du pouvoir discriminant des primitives

La classification consiste à attribuer une classe ou une étiquette à un vecteur de mesures effectuées sur le phénomène que l'on désire modéliser. Dans le cas d'une modélisation discrète, un symbole, généralement appelé primitive, sert à représenter ce vecteur. Dans la plupart des applications il est possible d'effectuer un grand nombre de mesures du phénomène et donc d'obtenir un nombre de primitives potentielles important. De manière à réduire la complexité du système, il est conseillé de conserver uniquement les meilleures primitives. Cette phase de sélection nécessite la définition d'un indicateur quantitatif de la qualité des primitives.

Pour cela il est possible d'utiliser les concepts de la théorie de l'information. Premièrement l'entropie conditionnelle des classes étant données les primitives $H(C|F)$ permet de quantifier l'information moyenne qu'apporte la connaissance d'une primitive sur les classes. Comme mentionné dans la sous section précédente, El-Yacoubi a proposé d'utiliser la perplexité conditionnelle pour quantifier le pouvoir discriminant des primitives. Un indicateur individuel peut être obtenu pour chaque primitive f_j en utilisant la définition de l'équation 1.65.

Une seconde possibilité consiste à utiliser l'information mutuelle entre les classes et les primitives $I(C; F)$. Cette dernière mesure la quantité d'information moyenne que la connais-

sance d'une des deux variables apporte sur la seconde. Elle traduit d'une certaine manière la dépendance entre les deux variables aléatoires. Par l'intermédiaire des équations 1.70 et 1.71 nous obtenons respectivement une mesure globale pour l'ensemble de primitives et une mesure individuelle pour chaque primitives f_j.

Il existe une relation entre l'entropie conditionnelle et l'information mutuelle, qui s'écrit en fonction des variables aléatoires C et F :

$$I(C;F) = H(C) - H(C|F) \tag{1.72}$$

En comparant les équations 1.68 et 1.69 nous pouvons constater que les définitions de l'entropie conditionnelle et l'information mutuelle sont similaires. La seule différence se situe au niveau de l'argument de la fonction $\log(\cdot)$. Pour l'entropie conditionnelle il est égal à $\Pr(\omega_i|f_j)$ alors que pour l'information mutuelle il est égal à $\frac{\Pr(\omega_i|f_j)}{\Pr(\omega_i)}$. En fait la différence est introduite uniquement par la distribution des classes ω_i et plus particulièrement par les probabilités marginales $\Pr(\omega_i)$, ce que traduit l'équation 1.72.

Pour un problème de classification donné, le nombre de classes dépend uniquement de la modélisation choisie. La distribution de ces classes, étant donnée la modélisation, ne dépend alors que des données utilisées pour réaliser l'estimation des probabilités. De ce fait l'utilisation d'une base de donnée de taille fixe va conduire à une distribution des classes unique. Cela signifie que pour l'évaluation de différents ensembles de primitives extraits de cette même base, les indicateurs d'entropie conditionnelle $H(C|F)$ et d'information mutuelle $I(C;F)$ varieront proportionnellement.

Afin de valider cette conclusion, nous avons réalisé une série d'expériences au cours de laquelle les différents indicateurs entropiques ont été évalués. À partir d'un espace de représentation continu, différents ensembles de primitives ont été créés, en augmentant le nombre de classes de la partition lors de la quantification vectorielle. L'algorithme LBG

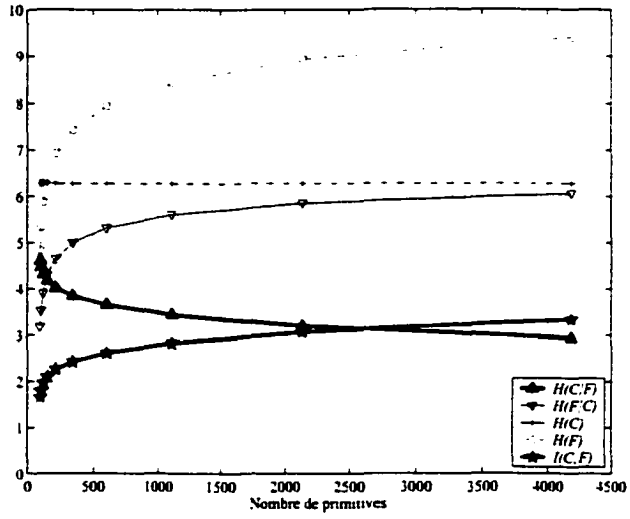


FIGURE 9 Comparaison des différentes entropies et de l'information mutuelle.

(voir section 1.3) a été utilisé, ce qui conduit à des ensembles de primitives de taille 2^n. Les résultats obtenus sont présentés à la figure 9.

L'analyse des différentes courbes permet de confirmer notre première conclusion, à savoir que l'entropie des classes est constante quelle que soit la taille de l'ensemble de primitives utilisé. Nous pouvons également constater que l'information mutuelle varie de manière inversement proportionnelle à l'entropie des classes, comme le mentionne l'équation 1.72.

L'augmentation du nombre de primitives conduit à une diminution de la grandeur $H(C|F)$. L'interprétation est la suivante : l'augmentation du nombre de primitives conduit à réduire la quantité d'information moyenne qu'une primitive individuelle f_j apporte sur les classes. Du fait de la relation qui lie l'entropie conditionnelle et l'information mutuelle, cette dernière augmente lors de l'accroissement du nombre de primitives. L'interprétation de ce phénomène est que l'accroissement du nombre de primitives conduit à augmenter la quantité d'information moyenne que la connaissance d'une primitive ou d'une classe apporte sur l'autre variable.

Nous pouvons également constater la propriété de croissance de l'entropie énoncée par l'équation 1.48. En effet l'augmentation de la taille de l'ensemble de primitives $\mathcal{F}$ conduit à un accroissement de l'entropie moyenne associée $H(F)$, ainsi que de l'entropie conditionnelle moyenne des primitives étant données les classes $H(F|C)$. Cette propriété interdit l'utilisation des indicateurs entropiques moyens pour la comparaison d'ensembles de primitives ne contenant pas le même nombre de primitives.

Du point de vue de l'évaluation d'ensembles de primitives de tailles différentes, à l'aide d'une même base de données et d'une même modélisation, les quantités $H(C|F)$ et $I(C;F)$ permettront d'obtenir une information sur leurs pouvoirs discriminants, mais elles ne permettront pas de d'énoncer infailliblement qu'un ensemble est plus discriminant que l'autre. Par contre l'évaluation individuelle de la qualité d'une primitive à l'aide des indicateurs $H(C|F = f_j)$ et $I(C; F = f_j)$ ne sera pas influencée par cette propriété. De ce fait la comparaison de primitives va être possible, tant au sein d'un même ensemble qu'entre ensembles de tailles différentes.

Pour le développement de notre projet, afin de quantifier le pouvoir discriminant des primitives, nous avons utilisé l'entropie conditionnelle des classes étant données les primitives plutôt que l'information mutuelle. Notre choix a été guidé par le fait que l'entropie conditionnelle caractérise plus directement la variable aléatoire F que l'information mutuelle, qui, elle, caractérise le couple de variables (C, F).

Dans cette section, nous avons présenté une introduction à la théorie de l'information. Le concept principal de cette théorie est celui de la mesure quantitative de l'information apportée par une variable aléatoire. Cette mesure est appelé entropie. Par la suite, nous avons montré l'application de ce concept au domaine de la classification. Une telle mesure est utilisée afin de quantifier l'information apportée par la représentation symbolique sur les différentes classes de la modélisation.

1.3 La quantification vectorielle

1.3.1 Définition

La quantification est le processus qui permet de fragmenter une grandeur physique en valeurs discrètes, multiples d'un quantum et exclusives de toute autre valeur. Cette étape est réalisée de manière à compresser l'information associée à la grandeur physique, généralement avant de la transmettre. Dans le cas de signaux multiples, chacun peut être quantifié indépendamment, on parle alors de *quantification scalaire*. Par contre si la quantification se fait simultanément pour l'ensemble des signaux, on parle de *quantification vectorielle*.

Considérons $\mathbf{x} = [x_1, x_2, \ldots, x_N]^t$ un vecteur de dimension N, dont toutes les composantes $\{x_i, 1 \leq i \leq N\}$ sont des variables aléatoires à valeurs réelles et amplitude continue. La quantification vectorielle consiste à faire correspondre à $\mathbf{x}$ un nouveau vecteur $\mathbf{z}$, de même dimension N, à valeurs réelles mais d'amplitude discrète. En notant $q(\cdot)$ l'opérateur de quantification, nous pouvons écrire :

$$\mathbf{z} = q(\mathbf{x}) \tag{1.73}$$

Le vecteur résultant $\mathbf{z}$ prend sa valeur parmi un ensemble fini de solutions $\mathcal{Z} = \{\mathbf{z}_i, 1 \leq i \leq N_S\}$ où $\mathbf{z}_i = [z_1^i, z_2^i, \ldots, z_N^i]^t$. L'ensemble $\mathcal{Z}$ est appelé alphabet et N_S est la taille de cet alphabet. Sa construction est obtenue en divisant l'espace à N dimensions du vecteur aléatoire $\mathbf{x}$ en M régions ou cellules et en associant à chacune de ces régions $\mathcal{R}_i$ un vecteur $\mathbf{z}_i$. La quantification consiste alors à attribuer au vecteur $\mathbf{x}$ la valeur $\mathbf{z}_i$ suivant sa position dans l'espace :

$$q(\mathbf{x}) = \mathbf{z}_i \quad \text{si} \quad \mathbf{x} \in \mathcal{R}_i \tag{1.74}$$

Une telle opération entraîne naturellement une perte d'information qui est appelée généralement *erreur de quantification*. Une *mesure de distorsion* $d(\mathbf{x}, \mathbf{z})$ est alors utilisée afin de

quantifier cette perte. Du fait que cette mesure peut être assimilée à une distance, il existe un grand nombre de possibilités lors de sa mise en œuvre [31]. La mesure de distorsion la plus utilisée est celle des moindres-carrés :

$$d(\mathbf{x}, \mathbf{z}) = \frac{1}{N} \sum_{j=1}^{N} (x_j - z_j)^2 \tag{1.75}$$

L'utilisation de cette mesure de distorsion suppose que chaque composante des vecteurs a la même importance. Il est possible que pour une application donnée cette contrainte ne soit pas satisfaisante, en particulier si les différentes composantes du vecteur ont des moyennes et/ou des variances hétérogènes. Il est alors conseillé de centrer et réduire les données et d'utiliser l'inverse de la matrice de covariance des données Σ pour pondérer les différentes composantes :

$$d(\mathbf{x}, \mathbf{z}) = (\mathbf{x} - \mathbf{z})^t \Sigma^{-1} (\mathbf{x} - \mathbf{z}) \tag{1.76}$$

Cette mesure de distorsion est connue sous le nom de *distance de Mahalanobis*.

Le processus de quantification est optimal lorsque la distorsion globale $\mathfrak{D}$, c'est-à-dire pour les L classes, est minimale. Cette dernière peut être définie par :

$$\mathfrak{D} = E[d(\mathbf{x}, \mathbf{z})] = \sum_{i=1}^{N_S} \mathcal{D}_i \tag{1.77}$$

où $E[\cdot]$ dénote l'espérance mathématique et $\mathcal{D}_i$ la distorsion moyenne pour la région $\mathcal{R}_i$.

1.3.2 L'algorithme *k-means*

Si la distorsion globale $\mathfrak{D}$ est utilisée comme critère lors de la définition de l'alphabet $\mathcal{Z}$, deux conditions sont nécessaires à l'optimalité de la solution. La première est que le processus de quantification doit être réalisé en utilisant la loi de sélection du plus proche

voisin. Cette dernière s'écrit :

$$q(\mathbf{x}) = \mathbf{z}_i, \text{ sssi } \quad d(\mathbf{x}, \mathbf{z}_i) \leq d(\mathbf{x}, \mathbf{z}_j), \text{ pour } \quad j \neq i \text{ et } 1 \leq j \leq N_S \tag{1.78}$$

Cela signifie que le processus de quantification choisit le vecteur $\mathbf{z}_i$ qui conduit à la plus petite distorsion étant donné le vecteur $\mathbf{x}$. La deuxième condition nécessaire concerne l'obtention des vecteurs $\mathbf{z}_i$ de l'alphabet $\mathcal{Z}$. Ces derniers doivent être choisis de manière à minimiser la distorsion moyenne de chaque région $\mathcal{R}_i$. Considérons un ensemble de N_V vecteurs d'apprentissage $\{\mathbf{x}^k, 1 \leq k \leq N_V\}$ et N_{V_i} le nombre de ces vecteurs contenu dans la région $\mathcal{R}_i$. La distorsion moyenne associée à chaque région est définie de la manière suivante :

$$\mathcal{D}_i = \frac{1}{N_{V_i}} \sum_{\mathbf{x} \in \mathcal{R}_i} d(\mathbf{x}, \mathbf{z}_i) \tag{1.79}$$

Le vecteur qui minimise la distorsion au sein d'une région $\mathcal{R}_i$ est appelé *centre de gravité* de la région $\mathcal{R}_i$ et est noté :

$$\mathbf{z}_i = cent(\mathcal{R}_i) \tag{1.80}$$

La technique la plus utilisée pour minimiser itérativement la mesure de distorsion moyenne est l'algorithme des *k-means* [31]. L'idée de base est de diviser l'ensemble des vecteurs d'apprentissage en L régions $\mathcal{R}_i$ de telle manière que les deux conditions nécessaires décrites ci-dessus soit respectées. L'algorithme peut être décrit de la manière suivante :

1. **Initialisation :** choisir un ensemble de vecteurs initiaux $\mathbf{z}_i, 1 \leq i \leq N_S$.
2. **Classification :** attribuer chaque élément de l'ensemble d'apprentissage $\{\mathbf{x}^k, 1 \leq k \leq N_V\}$ à une des régions $\mathcal{R}_i$ en choisissant le centre de gravité $\mathbf{z}_i$ le plus proche (loi de sélection du plus proche voisin, voir équation 1.78).

3. **Mise à jour des centres de gravité :** mettre à jour les centres de gravité $\mathbf{z}_i$ de chaque région $\mathcal{R}_i$ en calculant sa nouvelle position à l'aide des échantillons d'apprentissage attribués à cette région (équation 1.80).
4. **Fin :** si la différence de distorsion globale $\mathfrak{D}$ entre cette itération et la précédente est inférieure à un seuil prédéfini τ, alors **FIN** ; sinon aller à l'étape **Classification**.

Cette procédure divise la tâche de minimisation de la distorsion globale en deux étapes. La première assume que les centres de gravité sont connus et distribue les échantillons d'apprentissage parmi les L régions possibles en fonction de la mesure de distorsion. La seconde assume que les régions ou classes sont connues et consiste à calculer la nouvelle position du centre de gravité qui minimise la distorsion intra-classe. Il a été démontré que cet algorithme ne conduit qu'à un minimum local de la fonction de distorsion [105]. Une solution optimale peut être approximée en appliquant l'algorithme plusieurs fois avec des valeurs initiales différentes et en choisissant l'alphabet qui conduit à la plus faible distorsion.

1.3.3 L'algorithme LBG

Un autre algorithme utilisé pour la quantification vectorielle est celui de Linde, Buzo et Gray appelé LBG [105]. C'est une version étendue de l'algorithme *k-means*. Il permet de résoudre l'un des problèmes associés à l'algorithme des *k-means*, celui de l'initialisation des centres de gravité. En fait, l'algorithme LBG divise itérativement l'ensemble des échantillons d'apprentissage en $2, 4, \ldots, 2^p$ régions, et calcule le centre de gravité de chacune.

1. **Initialisation :** attribuer la valeur 1 à N_S (nombre de classes ou régions). Calculer le centre de gravité associé à l'ensemble des échantillons d'apprentissage.
2. **Division :** diviser les N_S régions en 2 : $N_S \times 2 \rightarrow N_S$.
3. **Classification :** attribuer chaque échantillon d'apprentissage $\mathbf{x}^k$ à une des régions $\mathcal{R}_i$ en choisissant le centre de gravité $\mathbf{z}_i$ le plus proche (idem étape 2 du *k-means*).

4. **Mise à jour des centres de gravité :** mettre à jour les centres de gravité $\mathbf{z}_i$ de chaque région $\mathcal{R}_i$ (idem étape 3 du *k-means*).
5. **Fin Classification :** si la différence de distorsion globale $\mathfrak{D}$ entre cette itération et la précédente est inférieure à un seuil prédéfini, alors aller à l'étape **Fin** ; sinon aller à l'étape **Classification**.
6. **Fin :** si N_S est égal au nombre de centres de gravité désiré alors **FIN** ; sinon aller à l'étape **Division**.

Il existe différentes méthodes heuristiques pouvant être utilisées lors de l'étape de division permettant de trouver deux vecteurs éloignés dans la même partition.

Nous venons de présenter deux algorithmes de quantification vectorielle classique et simple de mise en œuvre. Il existe de nombreux autres algorithmes, certains étant d'autres variantes de l'algorithme des *k-means*. Le domaine de recherche de la quantification vectorielle est toujours très actif, comme en témoigne le nombre de publications important sur le sujet : [9, 19, 64, 56, 68, 115, 119, 124] entre autres.

1.4 L'analyse discriminante linéaire

Comme nous l'avons mentionné dans la section 1.2.3, la classification consiste à inférer le sens à partir des mesures, c'est-à-dire établir la relation entre l'espace de représentation des formes et l'espace des sens possibles. La quantification vectorielle permet d'effectuer un partitionnement de l'espace d'un vecteur aléatoire x en un nombre fini de régions M, correspondant chacune à un symbole de l'alphabet résultant. Cette procédure s'effectue de manière non-supervisée, c'est-à-dire qu'à aucun moment on ne connaît le sens (ou classe) de l'échantillon traité. Il peut être judicieux, lors de la construction de l'alphabet, de prendre en compte cette information, lorsqu'elle est disponible. L'*analyse discriminante linéaire* (LDA pour Linear Discriminant Analysis en anglais) permet de prendre en compte cette information. Cette technique est également appelée analyse discriminante de *Fisher*, qui en a été l'instigateur dans les années 1930.

1.4.1 Définition

L'analyse discriminante linéaire permet d'obtenir une transformation linéaire d'un espace, garantissant une séparation des échantillons en fonction des modalités d'une variable qualitative. Considérons $\omega_1, \omega_2, \ldots, \omega_{N_C}$ comme les N_C étiquettes correspondant aux différentes modalités de cette variable ou encore comme les N_C classes d'une modélisation. Soit les N_V échantillons $\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^{N_V}$ de la base de données dans un espace à N dimensions : $\mathbf{x}^i = [x_1^i, x_2^i, \ldots, x_N^i]^t$. Ils se décomposent en fonction de la variable qualitative en N_C sous-ensembles notés $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_{N_C}$, chacun contenant respectivement $n_1, n_2, \ldots, n_{N_C}$ exemples. L'analyse discriminante permet de déterminer les directions permettant de réaliser une discrimination efficace par rapport aux classes ω_i. L'ensemble de ces composantes discriminantes est regroupé dans une matrice généralement notée W. Cette dernière permet de projeter les échantillons $\mathbf{x}^i$ dans un nouvel espace :

$$\mathbf{y}^i = \mathrm{W}^t \mathbf{x}^i \tag{1.81}$$

Chaque échantillon de la base de données $\mathbf{x}^i$ est caractérisé par un nouveau vecteur $\mathbf{y}^i = [y_1^i, y_2^i, \ldots, y_{N'}^i]^t$ où chacune des composantes y_k^i est une combinaison linéaire des x_k^i de départ. Il est à noter qu'une réduction de l'espace de représentation peut également être effectuée à l'aide de cet algorithme, c'est-à-dire que $N' \leq N$.

1.4.2 Description de l'algorithme

Dans la suite de cette section, nous allons décrire les variables utilisées et le cheminement conduisant à l'obtention de la matrice W. Pour avoir une description complète du processus, le lecteur peut se reporter à de nombreux ouvrages dans la littérature, comme [31] par exemple. Dans un premier temps nous allons définir le vecteur moyen associé à chacune

des classes ω_j. Il est noté $\mathbf{m}_j = [m_1^j, m_2^j, \ldots, m_N^j]^t$ et défini de la manière suivante :

$$\mathbf{m}_j = \frac{1}{n_j} \sum_{\mathbf{x} \in \mathcal{D}_j} \mathbf{x} \tag{1.82}$$

Chaque composante de ce vecteur est bien sûr obtenue à partir de celles des échantillons associés à la classe ω_i :

$$m_k^j = \frac{1}{n_j} \sum_{\mathbf{x} \in \mathcal{D}_j} x_k \tag{1.83}$$

Un autre vecteur moyen est utilisé, c'est le vecteur moyen total obtenu à partir de l'ensemble des échantillons $\bar{\mathbf{m}} = [\bar{m}_1, \bar{m}_2, \ldots \bar{m}_N]^t$. Il est obtenu de la manière suivante :

$$\bar{\mathbf{m}} = \frac{1}{N_V} \sum_{\mathbf{x}} \mathbf{x} = \frac{1}{N_V} \sum_{j=1}^{N_C} n_j \mathbf{m}_j \tag{1.84}$$

Ces deux vecteurs moyens sont utilisés pour la définition de deux matrices de dispersion (*scatter matrices*). La première est la matrice de *dispersion intra-classes* S_W, appelée en anglais *within-class scatter*. Pour chaque classe ω_j, une matrice de dispersion intra-classes S_j est calculée à l'aide de l'équation suivante :

$$S_j = \sum_{\mathbf{x} \in \mathcal{D}_j} (\mathbf{x} - \mathbf{m}_j)(\mathbf{x} - \mathbf{m}_j)^t \tag{1.85}$$

La matrice de dispersion intra-classes globale S_W est alors obtenue à partir des matrices S_j de la manière suivante :

$$S_W = \sum_{j=1}^{N_C} S_j \tag{1.86}$$

La seconde matrice nécessaire à l'obtention des composantes discriminantes est celle de la *dispersion inter-classes* S_B, appelée en anglais *between-class scatter*. Elle est obtenue

à l'aide des différents vecteurs moyens :

$$S_B = \sum_{j=1}^{N_C} n_j(\mathbf{m}_j - \bar{\mathbf{m}})(\mathbf{m}_j - \bar{\mathbf{m}})^t \tag{1.87}$$

La matrice de dispersion totale associée aux données, notée S_T, peut être obtenue de la manière suivante :

$$S_T = S_W + S_B \tag{1.88}$$

L'obtention des composantes discriminantes consiste alors à trouver la matrice W qui permet de maximiser la mesure de dispersion inter-classes tout en minimisant la mesure de dispersion intra-classes (voir [31] pour plus de détails). Ce problème peut se traduire par la fonction objective :

$$J(W) = \frac{|W^t S_B W|}{|W^t S_W W|} \tag{1.89}$$

Les colonnes $\mathbf{w}_i$ de la matrice W correspondent alors aux vecteurs propres généralisés de l'équation suivante :

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i \tag{1.90}$$

où les λ_i sont les valeurs propres de la matrice W. Cette équation peut également s'écrire de la manière suivante :

$$S_W^{-1} S_B \mathbf{w}_i = \lambda_i \mathbf{w}_i \tag{1.91}$$

L'obtention des composantes discriminantes s'obtient alors en résolvant ce problème. Comme nous l'avons mentionné auparavant, une réduction de dimension de l'espace de

représentation peut être effectuée. Elle consiste à conserver uniquement les composantes discriminantes w_i correspondantes aux valeurs propres les plus grandes.

L'application de cette technique permet donc de réduire la dispersion intra-classe des échantillons tout en augmentant la dispersion inter-classe. Idéalement lors de la construction d'un alphabet par quantification vectorielle, chaque symbole devrait correspondre à une classe et à une seule. L'obtention d'un tel alphabet réduirait la tâche de classification à l'extraction des primitives. L'application de l'analyse discriminante linéaire, avant la quantification vectorielle, permet de réduire la distorsion moyenne associée à chaque région (voir équation 1.79) et ainsi d'obtenir une meilleure solution. En d'autres termes, cette technique va permettre d'augmenter la corrélation entre les primitives et les classes de la modélisation, ce qui permet de faciliter la tâche de reconnaissance. L'application de l'algorithme LDA avant la quantification vectorielle permet d'obtenir des ensembles de primitives plus discriminants.

1.5 Résumé

Dans ce chapitre nous avons présenté les principaux outils théoriques utilisés au cours du développement de notre projet. Les modèles de Markov cachés, utilisés pour la modélisation de l'écriture, ainsi que les différents algorithmes permettant leurs mises en œuvre, ont été décrits. Nous avons ensuite introduit le domaine de la théorie de l'information et en particulier les différents indicateurs permettant de quantifier l'information associée à une variable aléatoire. L'application de ces concepts à la classification et à l'estimation du pouvoir discriminant de primitives a également été présentée. Nous avons poursuivi en énonçant le formalisme de la quantification vectorielle et en présentant deux algorithmes. Au cours de notre projet, nous avons utilisé le second afin de construire nos ensembles de primitives provenant d'espaces de représentation continus. Finalement nous avons exposé l'analyse discriminante linéaire. Cette technique a été utilisée dans le but d'augmenter le pouvoir discriminant de nos différents ensembles de primitives.

CHAPITRE 2

LES SYSTÈMES DE RECONNAISSANCE DE L'ÉCRITURE MANUSCRITE

Dans ce chapitre, nous allons présenter dans une première section notre domaine d'intérêt, à savoir celui de la reconnaissance de l'écriture manuscrite (REM). Le but de notre projet étant l'amélioration des performances d'un système de reconnaissance de l'écriture manuscrite, nous poursuivrons par une deuxième section consacrée à la description du système existant. Afin de dégager les différentes voies possibles pour son amélioration, une évaluation approfondie de ce système a été effectuée. Nous en discuterons dans la troisième section. Nous terminerons en exposant nos conclusions et la direction choisie pour nos travaux.

2.1 Introduction

De nos jours, l'écriture est toujours le moyen de communication visuelle le plus utilisé par l'homme. Il n'est donc pas surprenant de voir que de nombreux travaux scientifiques portent sur sa reconnaissance automatique. L'écriture est en fait la réalisation d'un message à transmettre, c'est-à-dire la représentation physique d'un contenu sémantique. Le média ou support généralement utilisé est le papier. Le but de la reconnaissance de l'écriture est de prendre une décision quant au contenu sémantique du message transmis à partir de sa représentation physique. Les applications de systèmes capables de remplir cette tâche sont nombreuses ; nous pouvons citer entre autres la lecture automatique de bons de commande, le traitement automatique des chèques, la vérification de signatures ou encore le tri automatique du courrier.

La reconnaissance de l'écriture est rattachée au vaste domaine de la reconnaissance de formes. Sa spécificité vient bien sûr des données à analyser et de leurs diverses sources de variation. En effet, un même mot écrit par plusieurs personnes peut avoir des formes assez différentes. Pour cette raison, les caractéristiques extraites de l'écriture sont très

importantes pour la suite du processus de reconnaissance. Ce dernier quant à lui, peut être mis en œuvre à l'aide de presque toutes les techniques développées en reconnaissance de formes.

2.2 Revue des principales méthodes de reconnaissance

Du signal écriture sous ses différentes formes, à la prise de décision par un système, il existe un certain nombre d'étapes à mettre en œuvre. La figure 10 représente globalement le processus de reconnaissance de l'écriture manuscrite. Sur cette dernière, les rectangles représentent une action par opposition aux ellipses qui elles caractérisent des données.

Dans un premier temps une phase de pré-traitement est réalisée. Elle permet de réduire au maximum la variabilité intrinsèque à l'écriture ainsi que les bruits possiblement introduits lors de l'acquisition. Une seconde étape, optionnelle, est celle de la segmentation. L'écriture étant une concaténation de caractères, il est normal lors de la reconnaissance d'essayer de segmenter l'écriture à reconnaître en caractères. La dernière étape a être réalisée directement sur les données présentées en entrée du système est l'extraction de caractéristiques. Son but est la réduction de la quantité d'information et l'extraction des caractéristiques les plus pertinentes pour la reconnaissance. Le chapitre 3 est consacré à l'extraction de caractéristiques pour la reconnaissance de l'écriture manuscrite. Après celle-ci, le système utilise une représentation symbolique de l'information, représentée par une chaîne de caractères sur la figure 10.

Lorsque le système est en phase d'apprentissage, nous avons besoin de l'étiquette, c'est-à-dire la chaîne de caractères exacte correspondant à l'exemple traité. Cette étape permet alors d'estimer les différents paramètres de la modélisation choisie, à partir du corpus de données d'apprentissage.

Par contre, lors de la phase de test, nous devons obtenir la séquence de caractères correspondant à l'exemple traité. Généralement le système de reconnaissance est dédié à une

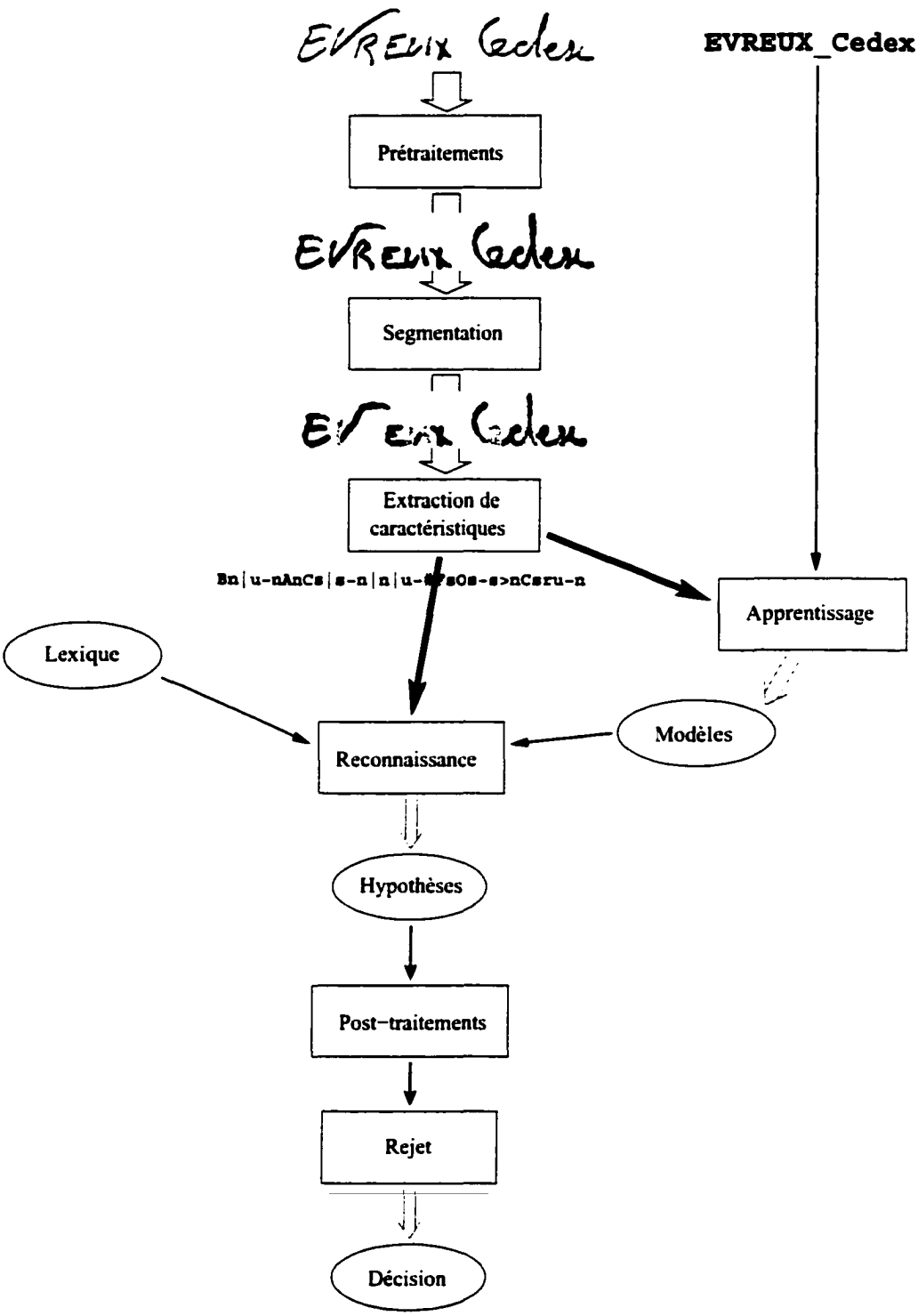


FIGURE 10 Synopsis d'un système de reconnaissance de l'écriture.

application particulière, ce qui restreint l'espace de recherche à un lexique donné. De manière plus ou moins directe, cette étape permet d'obtenir une liste de candidats possibles, aussi appelée hypothèses. Généralement chacune d'elles est caractérisée par un indice de confiance. Le choix de la solution finale peut être réalisée directement à l'aide de cet indicateur.

Cependant, un module de post-traitement peut être ajouté. Ce dernier procède généralement par intégration d'informations supplémentaires et/ou complémentaires. Cette dernière peut être de différentes natures : contextuelle, lexicale, sémantique,.... Les post-traitements doivent permettre de faciliter la prise de décision finale. Finalement un dernier module intégrable est celui du rejet. Il s'agit d'une procédure permettant au système de spécifier qu'il n'est pas capable de prendre une décision. En effet, pour certaines applications, cette alternative est préférable à une mauvaise décision.

Nous venons de présenter succinctement les différents modules possibles d'un système de reconnaissance de l'écriture manuscrite. Différents critères permettant de les caractériser plus précisément existent. Nous allons les passer en revue dans la suite de cette section.

2.2.1 Systèmes en-ligne / hors-ligne

Une différence fondamentale entre les systèmes de reconnaissance de l'écriture est liée au moyen utilisé pour l'acquisition des données. Une distinction est faite entre les systèmes en-ligne et les systèmes hors-ligne.

L'écriture en ligne est obtenue lors de sa réalisation par une saisie en continu du tracé. Les données se présentent alors sous la forme d'une séquence de points ordonnés dans le temps. Dans ce cas, le signal est de type 1D et le système de reconnaissance peut bénéficier de la représentation temporelle. De ce fait il existe une analogie avec la reconnaissance de la parole. Il n'est donc pas surprenant de voir des chercheurs appliquer les techniques

développées pour la parole à l'écriture [146]. Un état de l'art des principaux systèmes de reconnaissance de l'écriture en-ligne est présenté dans [149].

La seconde catégorie de systèmes, celle qui nous intéresse plus particulièrement, effectue la reconnaissance une fois que l'écriture est présente sur un support en papier. L'acquisition des données se fait à l'aide d'un scanner ou d'une caméra qui permet de convertir l'écriture en images binaires ou en niveaux de gris. L'information est alors bidimensionnelle. Par rapport aux systèmes en-ligne, les systèmes hors-ligne ne disposent plus de l'information temporelle et dynamique du tracé. De plus l'épaisseur du tracé devient une contrainte supplémentaire à prendre en compte.

Les principales applications des systèmes hors-ligne sont : le tri postal [23, 25, 38, 41, 47, 53, 81, 82, 97, 100] et la lecture des montants manuscrits sur les chèques [30, 33, 52, 62, 63, 88, 87, 102, 126].

Cette deuxième catégorie est celle qui nous intéresse plus particulièrement. Notre exposé portera uniquement sur les systèmes de ce type.

2.2.2 Grands vocabulaires / vocabulaires restreints

Comme nous l'avons déjà mentionné, le développement d'un système de reconnaissance est généralement guidé par une application. Quelle qu'elle soit, il existe un lexique, ou vocabulaire associé, c'est-à-dire un ensemble de mots qui peuvent être rencontrés. La taille de ce vocabulaire est également un critère de distinction entre les différents systèmes de reconnaissance de l'écriture. Le lexique est considéré comme petit s'il contient moins de 100 mots et grand s'il en comporte plus de 500.

La lecture des montants manuscrits des chèques utilise un lexique de petite taille. En effet le vocabulaire complet de ce type d'application ne dépasse pas trente mots. Par contre, la reconnaissance des noms de ville pour le tri postal entraîne l'utilisation de grands lexiques contenant tous les noms de ville possibles.

2.2.3 Approche globale / approche analytique

Une autre différence importante entre les systèmes de reconnaissance de l'écriture est liée à la nature de la modélisation mise en œuvre. L'approche globale consiste à modéliser les mots comme des entités globales, sans essayer d'identifier les caractères qui le composent. Pour une application donnée, il faudra générer autant de modèles que de mots présents dans le vocabulaire. L'approche analytique, fondamentalement opposée, consiste à reconnaître un mot après avoir identifié les lettres le constituant. Cette technique conduira à la création de modèles caractères et non plus de modèles mots.

Le choix de l'approche globale ou analytique est généralement dicté par la taille du vocabulaire. Dans le cas de petits lexiques, il est envisageable de créer autant de modèles que de mots possibles [52, 62, 88, 127, 144]. Par contre pour un vocabulaire de plusieurs milliers de mots, il n'est pas concevable de créer un modèle pour chaque classe ; il est plus pertinent de modéliser au niveau d'entités plus petites comme les caractères [22, 37, 51, 53]. Dans ce cas, le système doit segmenter l'écriture en entités de base.

2.2.4 Segmentation implicite / explicite

Sur la figure 10, nous avons indiqué la présence d'une étape de segmentation. Il existe deux techniques permettant sa mise en œuvre. La première consiste à effectuer un découpage *a priori* de l'image en intervalles de grandeur régulière [51, 118, 140]. Nous parlons alors de segmentation implicite. Cette technique est similaire à celle utilisée en reconnaissance de la parole, où le signal est divisé en intervalle de temps régulier. Par opposition, la segmentation explicite [13, 22, 53, 97] consiste à utiliser des points caractéristiques dans le mot, tels que les minima locaux du contour supérieur, les espaces ou encore les points d'intersection. Le résultat de cette étape est la segmentation du mot en entités de base appelées graphèmes.

2.2.5 Stratégies de reconnaissance (mots isolés/phrases)

Les principales applications de la reconnaissance de l'écriture manuscrite sont la lecture des montants littéraux de chèques et la lecture des enveloppes postales. Dans les deux cas le système doit reconnaître un ensemble de mots. À ce moment deux stratégies peuvent être envisagées : reconnaître les mots séparément ou alors essayer de reconnaître le groupe de mots [138].

Dans la plupart des systèmes existants la reconnaissance des mots est tout d'abord mise en œuvre, puis la combinaison des différents mots permet d'obtenir un résultat pour le groupe de mots. Dans le cas de la reconnaissance des montants littéraux de chèques cette dernière étape permet d'effectuer une vérification grammaticale et ainsi de proposer seulement des montants grammaticalement corrects [63, 127]. Dans le cas de l'application à la reconnaissance des adresses le fonctionnement est similaire : la reconnaissance se fait au niveau du mot et non pas de la phrase entière ; diverses approches sont possibles [38, 41, 82].

2.2.6 Méthodes de reconnaissance

La reconnaissance est en fait la prise de décision par le système quant à la nature des données qui lui sont présentées en entrée. Pour cela il existe un grand nombre de techniques classées généralement en deux grandes catégories :

2.2.6.1 Les approches statistiques

La reconnaissance est alors l'étude statistique de mesures effectuées sur les formes à reconnaître. L'étude de leur répartition dans un espace métrique et la caractérisation statistique des classes permet de prendre une décision du type : plus forte probabilité d'appartenance à une classe. Ces méthodes s'appuient généralement sur des hypothèses concernant la description statistique des familles d'objets analogues dans l'espace de représentation. Utilisant cette approche, nous pouvons citer différentes méthodes : la distance de Mahala-

nobis, les réseaux de neurones [41, 46], la méthode des k plus proches voisins [114], les fenêtres de Parzen, les méthodes d'appariement de masques.

2.2.6.2 Les approches structurelles

Ces approches consistent à mettre en relation la structure des formes analysées et la syntaxe d'un langage formel. La description des formes est réalisée par l'intermédiaire de phrases et le problème de classification est ramené à un problème d'analyse de grammaire (*parsing*). De manière générale, les approches syntaxiques ou structurelles permettent la description de formes complexes à partir de formes élémentaires. Ces dernières, encore appelées caractéristiques, sont extraites directement des données présentes en entrée du système. La différence principale entre ces méthodes et les méthodes statistiques est que ces caractéristiques sont des formes élémentaires et non pas des mesures. Une autre différence est qu'elles introduisent la notion d'ordre dans la description d'une forme. Les méthodes les plus répandues utilisent le calcul de distance d'édition entre deux chaînes [63, 127] et la programmation dynamique [47, 81, 118].

2.2.6.3 Approche hybride

Une dernière approche est envisageable : l'approche hybride. La reconnaissance par modèles de Markov cachés en est une. En effet ils utilisent une approche statistique tout en ayant la possibilité d'utiliser des descriptions structurelles. Leur application dans le domaine de la reconnaissance de l'écriture est de plus en plus présente [16, 23, 38, 51, 53, 98, 140]. Leur succès en reconnaissance de la parole en est la cause principale [69, 135]. Une description complète de l'application de ces modèles à la reconnaissance de mots manuscrits est présentée dans [96].

Il existe également des approches couplant différentes techniques présentées ci-dessus : modèle de Markov et réseaux de neurones [6, 87], modèles de Markov cachés et k plus proches voisins [62].

Nous venons de passer en revue les principales différences existant entre les systèmes de REM. Dans la suite de ce chapitre, nous allons décrire le système standard du SRTP, utilisé au cours de notre projet. Pour plus de détails, le lecteur peut se reporter à différents articles présentant une revue de différents aspect du domaine [90, 129, 147, 153].

2.3 Le système standard du SRTP

Le SRTP étant en charge du choix de la technologie utilisée pour le tri du courrier au niveau de la France, son personnel connaît bien la problématique associée à la reconnaissance de l'écriture manuscrite. Le système standard du SRTP est l'aboutissement de nombreuses années de travail. Son développement a été l'objet de nombreuses publications : [34, 38, 35]. En particulier, El-Yacoubi fait une description complète du système dans [36]. La version du système présentée dans cet article est le point de départ de notre projet. Dans la suite de cette section nous allons décrire les différents modules mis en œuvre.

2.3.1 Les pré-traitements

La première étape est celle des pré-traitements. Elle a pour but de réduire au maximum les bruits dans l'image et d'éliminer autant que possible les variabilités liées au style d'écriture. En effet ce type de signal peut être sujet à un grand nombre de variations. Elles peuvent provenir de différentes sources : le matériel, le scripteur ou encore le contexte. La figure 11 représente un exemple d'image traitée par notre système. Il s'agit de la composante nom de ville d'une enveloppe de courrier réel. Cet exemple permettra d'illustrer les différentes étapes des pré-traitements.

Pour éliminer ou tout au moins réduire au maximum ces variabilités, le système actuel procède en plusieurs étapes. Avant d'appliquer une procédure quelconque de pré-traitement, un épaississement du tracé est réalisé. Il a pour but de rétablir la connexité des composantes connexes qui peut être perdue au cours des phases d'acquisition et de binarisation de l'image. Il est réalisé en ajoutant à l'image d'un mot donné son contour externe.

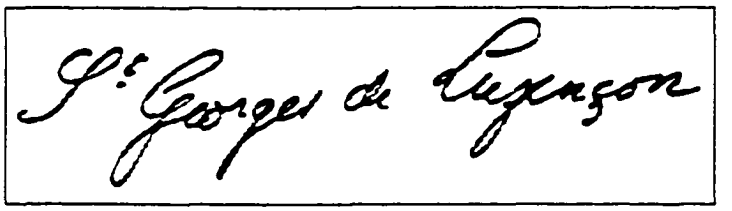


FIGURE 11 Champs "nom de ville" d'une image de la base de données SRTP.

2.3.1.1 La normalisation de la ligne de base

La ligne de base est définie comme la ligne sur laquelle reposent les lettres ne possédant pas de dépassement bas. Le but de cette procédure est de ramener cette droite à l'horizontale.

La méthode mise en œuvre procède en deux temps. Une première phase permet d'estimer l'inclinaison globale de cette ligne. Pour cela les minima du contour inférieur de l'écriture sont détectés. Une phase de filtrage, basée sur des heuristiques, permet d'éliminer les minima indésirables (ceux situés sur les dépassements bas). Ensuite la méthode de régression linéaire (moindres carrés) est utilisée afin d'ajuster une droite parmi les minima restants et ainsi obtenir l'angle d'orientation de l'écriture. Le redressement est alors exécuté par l'intermédiaire de la transformation de Hook. La transformation de Hook d'angle α_H permet de faire une rotation du mot d'un angle α_H. Elle est réalisée en remplaçant les coordonnées (x, y) de chaque pixel noir par les coordonnées (x', y') données par :

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\alpha_H & -\sin\alpha_H \\ \sin\alpha_H & \cos\alpha_H \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.1}$$

et en se limitant à la valeur entière.

Une seconde phase est alors mise en œuvre afin d'affiner le résultat obtenu. Les mêmes étapes de détection et de filtrage des minima sont réalisées sur l'image résultante du premier traitement. La ligne de base n'est alors plus considérée comme une droite, mais

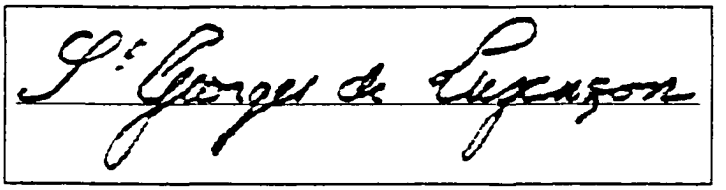


FIGURE 12 Résultat de la normalisation de la ligne de base.

comme une succession de segments, chacun reliant deux minima consécutifs. Ces derniers permettent de délimiter une partie de l'image que nous appelons imagette. Pour chacune d'elles une correction individuelle est effectuée, c'est une transformation colonne par colonne. Pour chaque imagette Ig_i délimitée par les minima i et $i+1$ la transformation permet de remplacer les coordonnées (x, y) de chaque pixel noir par les coordonnées (x', y') obtenues de la manière suivante :

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\tan\alpha_i & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ y_1 - y_i \end{bmatrix} \tag{2.2}$$

où α_i est l'angle de la pente entre l'axe horizontal et la droite reliant les minima délimitant l'imagette Ig_i et où y_1 et y_i sont respectivement les ordonnées du premier minimum et du minimum d'indice i.

Après cette transformation les minima sont tous positionnés sur une ligne horizontale. Cette méthode permet de tenir compte de la présence de plusieurs inclinaisons dans une image. Le résultat de cette étape est présenté sur le figure 12. Nous pouvons constater que le nom de ville est correctement positionné sur une droite horizontale. Un certain nombre d'heuristiques étant utilisé, le résultat de cette étape de pré-traitements correspond à nos attentes dans une grande majorité des images traitées. Il est certain que dans certains cas la transformation réalisée conduit à une déformation non désirée de l'image, qui peut introduire une certaine confusion dans le système de reconnaissance.

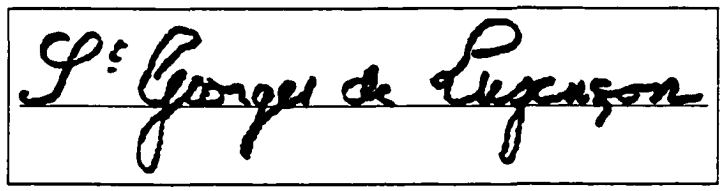


FIGURE 13 Résultat de la correction de l'inclinaison des caractères.

2.3.1.2 La correction de l'inclinaison des caractères

L'inclinaison des caractères est définie comme étant l'angle entre l'axe correspondant à la direction moyenne des caractères et l'axe vertical. L'objectif de ce pré-traitement est de transformer le mot de façon à ce que cet axe de direction principale devienne vertical. Ceci permet de réduire considérablement la variabilité de l'écriture.

La méthode mise en œuvre est semblable à celle développée par Kim [80]. De la même manière que pour la normalisation de la ligne de base, le contour extérieur de l'écriture est utilisé. Il est échantillonné avec un pas de 8 pixels. Pour chaque segment reliant deux points adjacents une inclinaison est obtenue. Les segments horizontaux ou pseudo-horizontaux sont éliminés. L'inclinaison globale des caractères β_{Glob} est alors obtenue en calculant la moyenne des inclinaisons de tous les segments restants.

La normalisation est effectuée à l'aide d'une transformation ligne par ligne où chaque pixel noir de coordonnées (x, y) est remplacé par les coordonnées (x', y') données par :

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & -\tan\theta \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.3}$$

avec $\theta = \arctan(\beta_{Glob})$. Après cette phase l'écriture est positionnée sur une droite horizontale et l'inclinaison des caractères est nulle ou presque.

2.3.1.3 La normalisation du corps des minuscules

Cette étape des pré-traitements a deux objectifs. Le premier est la réduction de la variabilité de l'écriture en normalisant la hauteur des lettres minuscules et le second la définition de zones d'écriture : la zone médiane ou corps du mot, la zone des dépassements hauts et la zone des dépassements bas. Ces informations sont pertinentes pour un système de reconnaissance de l'écriture manuscrite comme l'a montré Madhvanath dans [111].

La méthode développée est fondée sur l'analyse des maxima du contour supérieur de l'écriture. Après une étape de détection, un filtrage heuristique est appliqué sur les maxima de manière à éliminer les indésirables (ceux positionnés sur les dépassements hauts). Cette étape permet également de caractériser l'écriture comme étant cursive ou bâton, en calculant le rapport entre le nombre de maxima après et avant la phase de filtrage. La ligne de base supérieure est alors modélisée par la succession de segments reliant deux maxima consécutifs. La normalisation du corps des minuscules est réalisée sur chaque imagette délimitée verticalement par deux maxima successifs. Pour cela une transformation non linéaire continue des pixels est mise en œuvre. Elle permet de ramener les deux maxima à la hauteur h_m : hauteur moyenne des maxima restant après la phase de filtrage, tout en maintenant la ligne de base inférieure horizontale. Cette transformation est effectuée en remplaçant les coordonnées (x, y) de chaque pixel noir par les coordonnées (x', y') telles que :

$$\begin{cases} x' &= x \\ y' &= (ent)\left(\frac{h_m}{y_1 - y_{lb}} + \frac{x - x_1}{x_2 - x_1}\left(\frac{h_m}{y_2 - y_{lb}} - \frac{h_m}{y_1 - y_{lb}}\right)\right)(y - y_{lb}) \end{cases} \tag{2.4}$$

où (x_1, y_1) et (x_2, y_2) sont les coordonnées du premier et du deuxième maximum de l'imagette, y_{lb} est l'ordonnée de la ligne de base et (ent) désigne la valeur entière. En fait cette transformation n'est pas effectuée lorsque l'exemple traité est considéré comme bâton. En effet pour ce type d'écriture seule la zone médiane doit exister. Forcer la détection

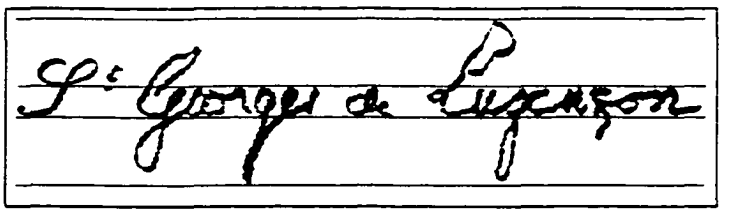


FIGURE 14 Résultat de la normalisation du corps des minuscules.

de la zone des dépassements hauts conduirait à introduire une certaine confusion dans le système de reconnaissance.

Après cette transformation les lettres minuscules sont comprises entre les lignes de base inférieure et supérieure. Du fait de l'aspect non-linéaire de la transformation, l'image résultante peut être distordue comme nous pouvons le constater sur la figure 14. La distorsion est particulièrement visible sur le L du dernier mot.

2.3.1.4 Le lissage

La dernière phase des pré-traitements est un lissage. Il permet de réduire au maximum les discontinuités introduites dans l'image au cours des différentes transformations et ainsi de rétablir la régularité et la continuité du contour du mot. Le lissage consiste à examiner le voisinage d'un pixel et de lui attribuer la valeur 1 si le nombre de pixel noir dans cette zone est supérieur à un seuil. Dans notre cas, un voisinage de 3×3 et un seuil de 4 ont été utilisés. Ceci nous permet, compte tenu de l'épaississement du tracé réalisé au début de la phase de pré-traitement, d'accomplir un lissage modéré qui éliminera seulement les bruits résiduels apparaissant aux bords de l'image. Le résultat de cette étape est présenté sur le figure 15.

L'image du mot ne peut pas être directement utilisée par le système de reconnaissance. Nous devons extraire des primitives qui résument l'information et réduisent la redondance présente dans les images. Pour cela le système du SRTP, utilisant une approche analytique, c'est-à-dire une modélisation au niveau du caractère, procède en deux étapes : première-

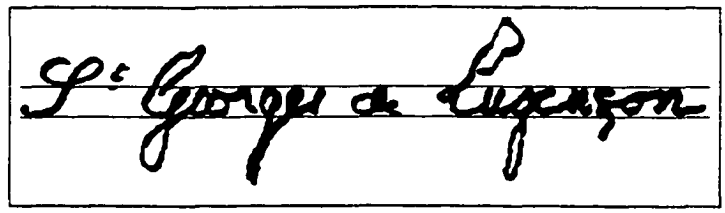


FIGURE 15 Résultat de l'étape de lissage de l'image.

ment la segmentation du mot en entités élémentaires appelées graphèmes, deuxièmement l'extraction de l'information pertinente pour la reconnaissance de chacun de ces segments.

2.3.2 La segmentation des noms de ville

Le vocabulaire associé à l'application visée est de grande taille. Cette contrainte implique l'utilisation d'une approche analytique et donc une modélisation au niveau du caractère. De manière à extraire l'information en fonction des caractères, une étape de segmentation est mise en œuvre. Cependant la segmentation des mots en caractères est une opération très difficile. À ce jour il n'existe pas d'algorithme permettant de la réaliser de manière correcte. De ce fait, la stratégie adoptée consiste à segmenter les mots en entités plus petites que le caractère et ensuite utiliser la phase de reconnaissance pour choisir les points de segmentation valides parmi l'ensemble proposé. Durant cette dernière étape le contexte local des graphèmes peut être pris en compte.

La technique utilisée par ce système est donc à rattacher aux méthodes de segmentation explicite. Elle est inspirée de celle développée par Leroux [103] pour la lecture automatique des montants littéraux des chèques postaux. La détection du contour du tracé de l'écriture est nécessaire. La technique est basée sur les deux hypothèses suivante :

- il existe des points de segmentation naturels associés aux caractères non liés,
- les liens entre caractères se situent en général à proximité d'un minimum du contour supérieur du tracé.

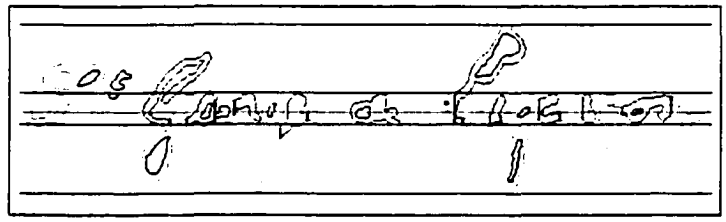


FIGURE 16 Résultat de l'étape de segmentation.

La détection des composantes connexes permet d'identifier les espaces dans la phrase à reconnaître. L'extrémité de chacune de ces composantes est alors considérée comme un point de segmentation sûr. Pour chacune d'elles une recherche des minima locaux du contour supérieur est réalisée. Un tri est alors réalisé en éliminant certains points et en en déplaçant d'autres, de manière à ne pas trop sur-segmenter l'image. La procédure consiste à définir une zone de recherche pour chaque minimum, bornée par les minima adjacents. Le point est alors déplacé de manière à satisfaire les deux contraintes suivantes :

- l'épaisseur du tracé doit être minimale,
- la valeur de l'histogramme de transitions verticales doit être minimale.

Il est possible que suite à cette phase, deux minima consécutifs soient très proches. Une dernière étape permet de les regrouper à l'aide d'un simple critère de distance horizontale.

Le résultat de l'étape de segmentation est présenté sur la figure 16. En plus des droites permettant de délimiter les zones d'écriture, nous avons représenté en vert clair celle correspondant au maximum de l'histogramme horizontal des pixels.

A partir de chacun des graphèmes maintenant définis, l'extraction de caractéristiques pertinentes pour la reconnaissance sera réalisée.

2.3.3 Les caractéristiques adaptées à l'écriture cursive et bâton

Les caractéristiques utilisées doivent permettre de résumer l'information contenue dans chaque segment et ainsi de réduire la quantité de données. Du fait que le système uti-

lise une modélisation discrète, le résultat de cette phase est une suite de symboles. Cette séquence constitue les données d'entrée du module de reconnaissance.

En fait le système de reconnaissance standard fonctionne avec deux jeux de primitives. Le premier a été défini de manière à caractériser l'écriture cursive alors que le second est plutôt dédié à la reconnaissance des échantillons bâtons. Afin d'intégrer en partie le contexte local des graphèmes, les points de segmentation sont pris en compte. Pour cela un jeu de primitives spécifique a été développé. Il contient cinq symboles distincts. À la fin de l'étape d'extraction de caractéristiques, une image est représentée par deux séquences de primitives. Elles sont composées d'une succession de couples de primitives, la première caractérisant le segment considéré et la seconde le point de segmentation séparant ce dernier du segment suivant.

2.3.3.1 Les primitives perceptuelles

Le premier jeu de primitives est fondé sur des caractéristiques topologiques et géométriques. Son développement est basé sur la perception humaine de l'écriture lors de la lecture. Les primitives sont globales et comprennent les boucles fermées, les dépassements hauts, les dépassements bas et les espaces inter-lettres ou inter-mots. Leur détection est réalisée par l'analyse des différents contours associés à un segment-lettre ou graphème. La justification de leur utilisation a fait l'objet de nombreux travaux, nous pouvons citer entre autres [110], [139].

La détection des dépassements hauts (respectivement bas) dans un segment est obtenue en comparant la position du maximum du contour supérieur (respectivement le minimum du contour inférieur) de ce segment par rapport à la ligne de base supérieure (respectivement inférieure) du mot. Les primitives associées sont codées de deux façons différentes, suivant que le dépassement est jugé important ou faible. Les boucles fermées sont détectées grâce à l'analyse des contours. Elles sont différemment codées suivant leur position (zone supérieure, zone médiane ou zone inférieure). Nous tenons compte également de la taille

de la boucle lorsque celle-ci est positionnée dans la zone médiane (petite ou grande). De plus, une dernière information est prise en compte au sujet des boucles médianes. En effet, afin de différencier par exemple un b d'un d, nous tenons compte de l'ordre d'apparition des primitives : boucle médiane avant ou après le dépassement.

Nous avons alors un ensemble de primitives de base qui peuvent être combinées afin de produire différents symboles. Cependant, l'utilisation de ce jeu de primitives a conduit à ne considérer que 26 combinaisons possibles, celles-ci permettant de modéliser tous les cas rencontrés. De plus il existe un symbole joker : "-" correspondant à aucune occurrence des primitives de base (voir annexe 1.1).

Comme nous l'avons mentionné auparavant, ce système prend en compte la nature des points de segmentation à l'aide d'un ensemble de cinq primitives (voir annexe 1.3). Deux cas sont à considérer : le point de segmentation a été généré par l'algorithme ou il est naturel. Pour le premier, les points de segmentation sont modélisés à l'aide de deux primitives. La position verticale de la coupure par rapport à la ligne de base est prise en compte. Le point de segmentation peut alors être proche ou éloigné de la ligne de base. Pour les points de segmentation naturels, trois cas sont envisagés, en fonction de l'espace séparant les segments (pas d'espace, petit ou grand espace). Ces cinq symboles sont utilisés par les deux jeux de primitives du système.

Nous obtenons finalement un alphabet de taille 32 : 27 symboles sur les formes des segments-lettre et 5 symboles sur la nature des points de segmentation.

2.3.3.2 Les primitives bâtons

Les caractéristiques que nous venons de décrire ne sont pas réellement performantes pour l'écriture bâton. En effet pour ce type d'écriture, les dépassements ne sont pas informants. Afin de compléter la caractérisation des segments, un deuxième jeu a été développé.

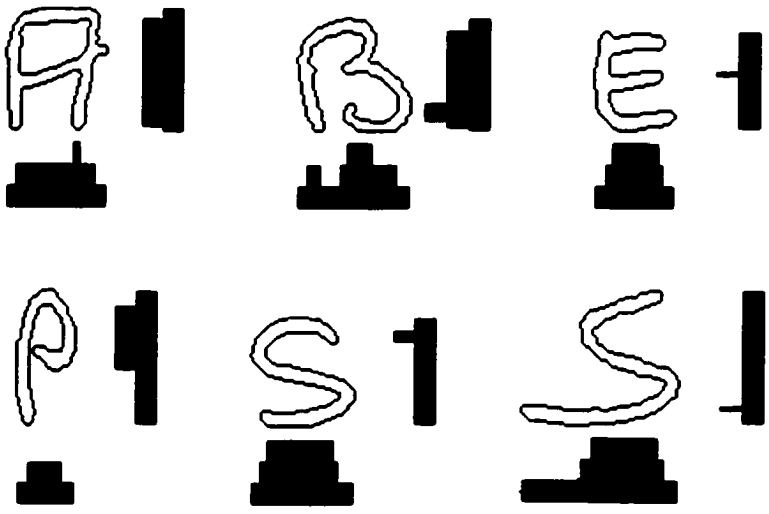


FIGURE 17 Histogrammes de transitions verticaux et horizontaux, extraits pour différentes lettres.

L'extraction des primitives bâtons est fondée sur l'analyse du contour bidimensionnel de chaque forme segmentée. Les histogrammes de transition selon les directions verticale et horizontale sont calculés (voir figure 17). Un filtrage est ensuite réalisé, de manière à remplacer chaque valeur par la moyenne calculée sur une fenêtre centrée de largeur cinq pixels. Chaque histogramme est alors divisé en cinq zones de même largeur. Afin de ne tenir compte que de la partie stable des graphèmes, seules les trois cellules centrales sont conservées. Nous évaluons alors le nombre de transitions dominant dans les deux directions (2, 4 ou 6). À chaque couple de valeurs est associée une primitive, ce qui conduit à $3 \times 3 = 9$ symboles ou classes. Cependant certaines caractéristiques supplémentaires, comme le nombre de points de contours positionnés sur la ligne de base ou la position du nombre de transitions dominants, ont permis de diviser certaines classes. Ces considérations conduisent à un ensemble de 14 symboles (voir annexe 1.2). Pour obtenir le jeu de primitives complet, les cinq symboles qui permettent d'intégrer la nature du point de segmentation sont ajoutés.

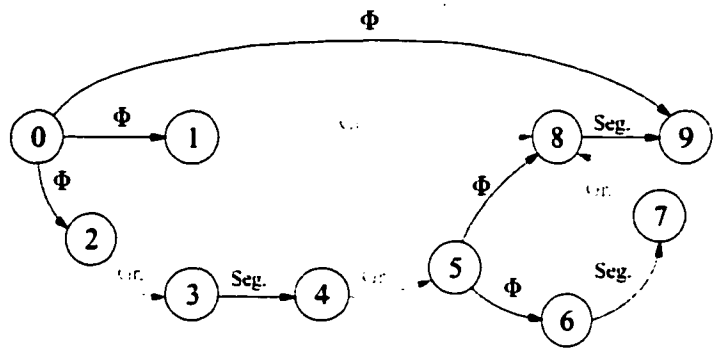


FIGURE 18 Architecture du modèle caractère prenant en compte trois segments [37].

Suite à cette étape d'extraction de primitives, le système standard dispose de deux séquences de symboles représentant le contenu de l'image du mot qui doit être reconnu. Elles sont l'entrée du module de reconnaissance.

2.3.4 La modélisation markovienne utilisée

Comme nous l'avons mentionné, le système de reconnaissance doit utiliser une approche analytique, c'est-à-dire que la modélisation est réalisée au niveau du caractère. La lecture du courrier conduit à modéliser en plus de l'ensemble des lettres majuscules et minuscules et des chiffres, un certain nombre d'autres caractères rencontrés sur les enveloppes : "- / \ ' , . espace". Nous disposons alors d'un total de 69 modèles de caractère. Mis à part le modèle espace, tous les autres ont la même architecture, présentée sur la figure 18.

La modélisation permet de prendre en compte les cas de sous-segmentation et de sur-segmentation d'un caractère jusqu'à trois segments par l'intermédiaire de quatre chemins. Le premier allant directement de l'état de départ 0 à l'état final 9 permet de modéliser la sous-segmentation du caractère. Les transitions marquée "Φ" symbolise les transitions nulles, c'est-à-dire celles n'émettant aucun symbole.

Le second chemin $0 \rightarrow 1 \rightarrow 8 \rightarrow 9$ modélise les cas de segmentation d'un caractère en un segment. Les deux derniers, $0 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 8 \rightarrow 9$ et $0 \rightarrow 2 \rightarrow$

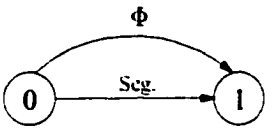


FIGURE 19 Architecture du modèle associé aux espaces.

$3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8 \rightarrow 9$ permettent de considérer les cas de sur-segmentation des caractères en deux ou trois segments respectivement. Sur la figure 18 les transitions marquées "Gr." correspondent à celles émettant une primitive caractérisant les graphèmes (primitives perceptuelles et/ou bâton). Suivant chacune d'elles, une transition marquée "Seg." est obligatoirement présente. Elle permet d'émettre la primitive associée au point de segmentation séparant le graphème en cours du suivant. Cette technique permet la prise en compte d'une partie du contexte des caractères directement au niveau de la modélisation. Elle permet une meilleure description de l'image et favorise la reconstruction des caractères à partir des graphèmes.

Un modèle particulier existe pour la prise en compte des espaces inter-mots. Son architecture est présentée sur la figure 19. Deux chemins sont possibles afin de prendre en compte le cas de sous-segmentation de l'espace dans la phrase. Il est à noter que l'autre transition ne peut émettre que deux symboles parmi les cinq associés aux points de segmentation : ceux correspondant à un petit et un grand espace (voir annexe 1.3).

Dans [37], El-Yacoubi a montré que la prise en compte des cas de sur-segmentation des caractères en trois graphèmes permet une amélioration des performances. Il mentionne également que ce phénomène n'est pas très fréquent. De manière à effectuer une estimation fiable des paramètres associés au troisième segment, il propose d'utiliser le concept d'états liés. Pour cela les quatre transitions entre les états 5, 6, 7 et 8 sont liées pour l'ensemble des modèles sauf "m, M, w et W" pour lesquels le nombre d'exemples dans la base de données est suffisant.

Dans la section 1.1.4 nous avons décrit une technique associée aux modèles de Markov cachés : l'interpolation de modèles. Les paramètres d'interpolation introduits permettent de choisir localement et automatiquement le meilleur modèle. Nous avons utilisé cette technique de manière à interpoler deux modèles dont la seule différence est l'ensemble de primitives utilisé. Le but était de choisir automatiquement la meilleure représentation de l'information contenue dans les graphèmes.

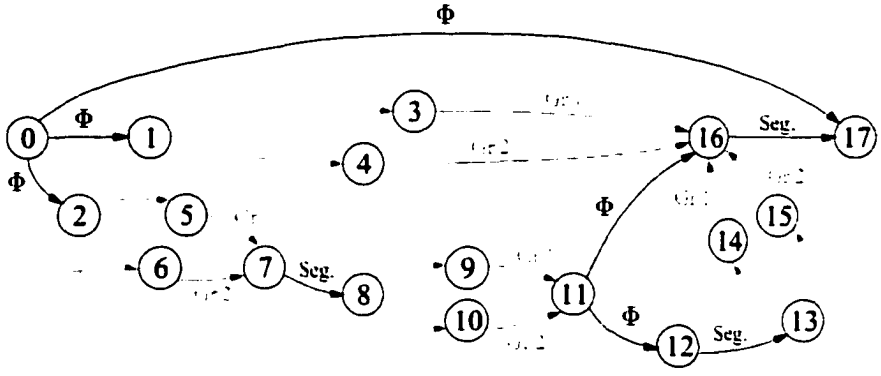


FIGURE 20 Architecture de l'interpolation du modèle caractère 3 segments.

Nous présentons sur la figure 20 le modèle résultant de l'application de l'interpolation. Seules les transitions associées aux graphèmes sont interpolées, comme présenté sur la figure 6. Lors de notre application de l'interpolation, seule la caractérisation de l'information présente dans les segments d'écriture diffère entre les modèles interpolés. De ce fait les paramètres associés aux points de segmentation sont identiques aux différents modèles ; ils ne nécessitent alors pas d'interpolation.

2.3.5 L'apprentissage des modèles

L'apprentissage du système est réalisé par l'intermédiaire de l'algorithme de Baum-Welch décrit dans la section 1.1.3.3. Pour sa mise en œuvre nous utilisons deux ensembles de séquences d'observations O_{App} et O_{Val}. Le premier est utilisé pour la ré-estimation des paramètres et le second pour tester le modèle obtenu à chaque itération.

La première étape de l'apprentissage est l'initialisation aléatoire de l'ensemble des paramètres. La seconde est la ré-estimation des paramètres à l'aide de l'algorithme de Baum-Welch. Afin de ne pas tomber rapidement dans un maximum local de la probabilité d'observation, une technique similaire au recuit simulé est appliquée. Elle consiste à altérer légèrement l'ensemble des probabilités après chaque itération, en fonction d'un facteur de température T. Ce dernier décroît régulièrement au cours de l'apprentissage de manière à diminuer progressivement cette altération. La phase de validation est alors effectuée. Elle a pour but d'assurer une bonne généralisation du modèle. Afin d'évaluer ce phénomène, la vraisemblance du corpus de validation $\mathbf{O}_{Val}$ par le modèle obtenu à l'itération en cours Λ^t est évaluée :

$$\mathcal{L}(\mathbf{O}_{Val}, \Lambda^t) = \frac{1}{Q} \sum_{i=1}^{Q} \log \Pr(\mathcal{O}^i | \Lambda^t) \tag{2.5}$$

Les probabilités des Q séquences de ce corpus sont obtenues à l'aide de la procédure Forward (voir section 1.1.3.1). Les étapes de ré-estimation, altération et validation sont alors répétées. La fin de l'apprentissage est obtenue lorsque l'on a simultanément :

$$T = 0 \quad \text{et} \quad \sigma_i \leq \epsilon \tag{2.6}$$

où σ_i est la différence relative entre la vraisemblance du corpus de validation entre l'itération i et l'itération $i-1$:

$$\sigma_i = \frac{\mathcal{L}(\mathbf{O}_{Val}, \Lambda^t) - \mathcal{L}(\mathbf{O}_{Val}, \Lambda^{t-1})}{\mathcal{L}(\mathbf{O}_{Val}, \Lambda^t) + \mathcal{L}(\mathbf{O}_{Val}, \Lambda^{t-1})} \tag{2.7}$$

A chaque itération de l'apprentissage cette quantité est évaluée et les différents paramètres sont mémorisés tant que la probabilité d'observation augmente. La valeur du seuil ϵ est typiquement de l'ordre de 10^{-2}.

Lors de l'interpolation de deux modèles, la même procédure d'apprentissage est utilisée, cependant seuls les paramètres de l'interpolation λ sont ré-estimés à chaque itération.

2.3.6 La stratégie de reconnaissance

L'étape de reconnaissance consiste à trouver le nom de commune qui a produit de manière la plus probable la séquence de primitives extraites de l'image traitée. Plus formellement, nous cherchons le nom de commune w, c'est-à-dire la séquence de caractères, appartenant au vocabulaire $\mathfrak{L}$, qui maximise la probabilité *a posteriori* que le modèle nom de commune associé ait produit la séquence de primitives inconnue $\mathcal{O}$:

$$\Pr(\hat{w}|\mathcal{O}) = \max_{w \in \mathfrak{L}} \Pr(w|\mathcal{O}) \tag{2.8}$$

L'utilisation de la règle de Bayes permet d'obtenir l'équation fondamentale de la reconnaissance de forme :

$$\Pr(w|\mathcal{O}) = \frac{\Pr(\mathcal{O}|w)\ \Pr(w)}{\Pr(\mathcal{O})} \tag{2.9}$$

Le terme $\Pr(\mathcal{O})$, la probabilité *a priori* de la séquence d'observation, est indépendant du mot testé w. Il n'influence pas la maximisation de l'équation 2.8 et peut donc être négligé. La probabilité *a priori* du mot : $\Pr(w)$, ne dépend pas de la forme à reconnaître. Elle est directement liée au vocabulaire de l'application. L'ensemble des probabilités associées à ce vocabulaire est appelé *modèle de langage*. Ce dernier permet d'incorporer une information contextuelle pouvant faciliter la reconnaissance. Cependant, dans le cas de notre application, utilisant un vocabulaire de grande taille, ce modèle est difficile à mettre en œuvre, principalement à cause du manque de données d'apprentissage. De ce fait il est plus judicieux de considérer l'ensemble des entrées de notre lexique équiprobables et ainsi négliger ce terme. Finalement l'optimisation de l'équation 2.8 se fait uniquement sur le terme $\Pr(\mathcal{O}|w)$.

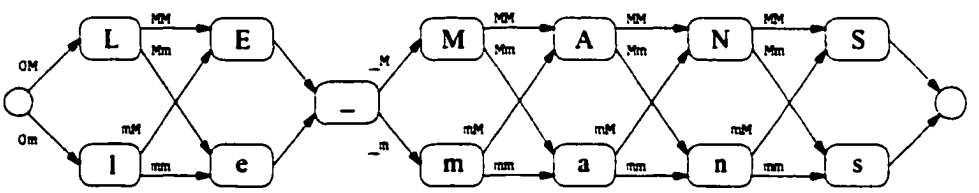


FIGURE 21 Architecture du modèle global de reconnaissance pour le nom de commune LE_MANS.

La mise en œuvre de cette procédure se fait par l'intermédiaire de l'algorithme de Viterbi (voir section 1.1.3.2). Il permet, pour chaque mot w_i du lexique $\mathfrak{L}$, d'effectuer l'alignement avec la séquence d'observations et de calculer la probabilité $\Pr(\mathcal{O}|w_i)$.

Afin de prendre en compte les différents styles d'écriture utilisés par les scripteurs, un modèle global de reconnaissance est construit. Ce dernier est composé des modèles caractères minuscules et majuscules en parallèle, comme présenté sur la figure 21. Un certain nombre de transitions permet de modéliser le changement de case en cours de réalisation de la phrase. Premièrement deux transitions partent de l'état initial notées OM et Om. Elles modélisent le fait que la première lettre est majuscule (M) ou minuscule (m). Par la suite, quatre transitions MM, Mm, mM et mm caractérisent les différentes possibilités de changement. Les probabilités associées à ces six transitions sont obtenues en estimant la fréquence des évènements correspondant, sur notre corpus de données d'apprentissage.

Sur la figure 21, un exemple comportant un modèle espace est représenté, ainsi que deux transitions permettant de poursuivre l'alignement avec un caractère majuscule ou minuscule. Cependant, pour la modélisation du système de base, l'évaluation de ces évènements n'a pas été effectuée. En fait ces transitions sont équiprobables de manière à ne pas favoriser un style plus qu'un autre après un espace.

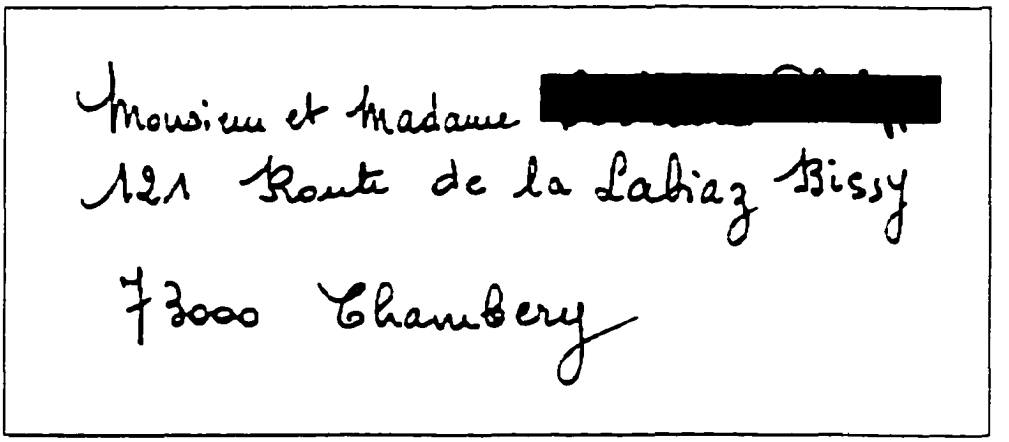


FIGURE 22 Exemple d'enveloppe de courrier français.

2.3.7 Le protocole de test du système

Lors de l'évaluation des performances du système, un protocole spécifique, guidé par l'application visée est utilisé. Il sera décrit dans cette sous-section.

Nous présentons sur la figure 22, un exemple d'enveloppe de courrier français. La disposition classique des différents champs (destinataire, ligne de distribution, code postal et nom de ville) y est respectée. L'information contenue par le champs "nom de ville" est redondante vis-à-vis de celle apportée par le code postal. En effet ce dernier permet de coder par l'intermédiaire de cinq chiffres l'ensemble des communes françaises, ainsi que certaines subdivisions. Cela signifie que l'identification du code postal induit la connaissance du nom de commune.

Après la localisation des différents champs sur l'enveloppe, la reconnaissance du code postal est effectuée. En effet il est plus facile de reconnaître une chaîne de cinq chiffres qu'une chaîne de caractères de longueur inconnue. La lecture du nom de commune sert en fait à valider le code postal trouvé. Ce dernier est donc utilisé pour guider la reconnaissance du nom de commune. Plus concrètement, un indice de confiance est attribué à chaque chiffre du code postal. Si la valeur de l'un d'eux est inférieure à un seuil fixé, un lexique

d'au maximum dix noms de ville est généré, en effectuant la recherche des codes postaux valides parmi les dix possibles. Lorsque l'indice de confiance de deux chiffres est faible, le lexique généré peut contenir jusqu'à 100 noms de ville, etc.

Afin de simuler les conditions réelles d'utilisation, l'évaluation du système est réalisée en effectuant plusieurs tests. La différence entre eux est le nombre de candidats potentiels testés. Dans un premier temps et de manière à conserver le protocole expérimental instauré par le concepteur du système standard, leur nombre est de 10, 100 et 1 000 noms de ville. De manière à mieux évaluer l'impact des améliorations apportées, nous avons étendu le nombre de tests à cinq en utilisant 5 000 et 10 000 candidats.

Pour chaque exemple contenu dans le corpus de test, l'alignement entre la séquence de primitives extraites et l'intitulé exact est effectué. Par la suite 9, 99, 999, 4 999 et 9 999 autres intitulés sont tirés aléatoirement dans le lexique global utilisé, contenant environ 37 000 entrées. L'intitulé reconnu est celui ayant la plus forte probabilité.

2.3.8 Les données utilisées

L'évaluation du système de reconnaissance a été effectué à l'aide de la base de données du SRTP principalement. Elle est composée de 408 fichiers divisés en deux groupes : *chaîne* et *nonchaine*, contenant respectivement 193 et 215 fichiers. Chaque fichier contient 100 images binaires d'enveloppes réelles du courrier français, scannées à 8 points par millimètre. La localisation et l'étiquetage des différents blocs présents sur les enveloppes (code postal, nom de commune, blocs de distribution,...) ont été effectués de manière manuelle.

2.3.8.1 Les corpus de données SRTP

Afin de mettre en œuvre l'apprentissage et le test du système, trois corpus sont nécessaires : apprentissage, validation et test. Le corpus d'apprentissage contient les images

extraites des fichiers *B100200000.pix* à *B100229900.pix*, celui de validation celles des fichiers *B100230000.pix* à *B100234500.pix*. Finalement le corpus de test contient les images extraites des fichiers *B100100000.pix* à *B100105400.pix*. Ces indications sont données à titre indicatif, de manière à ce que d'autres expérimentateurs puissent reproduire nos expériences et également comparer leurs résultats aux nôtres. Du fait de problèmes liés à l'extraction des informations de localisation, l'ensemble des images de certains fichiers n'est pas accessible. Le nombre exact d'échantillons contenu dans chaque corpus est présenté dans le tableau I. Il est à noter que l'ensemble des images extraites de ces différents fichiers a été conservé, aucun nettoyage des corpus n'a été réalisé.

TABLEAU I

Nombre d'échantillons des corpus extraits de la base SRTP.

Corpus	*Apprentissage*		*Validation*		*Test*	
BÂTON	4 839	*40,2%*	1 174	*33,8%*	1 684	*36%*
Cursif	6 104	*50,8%*	1 831	*52,7%*	2 440	*52,2%*
MiXTe	1 079	*9%*	470	*13,5%*	550	*11,8%*
Total	**12 022**		**3 475**		**4 674**	

Durant notre travail, nous avons évalué l'influence du style d'écriture sur les performances du système de reconnaissance. Nous considérons trois types d'écriture :

- *bâton* : toutes les lettres composant le mot ou la phrase sont majuscules,
- *cursif* : toutes les lettres sont minuscules, exception faite des premières lettres de chaque mot, qui peuvent être majuscules,
- *mixte* : tous les autres cas.

Nous présentons dans le tableau I le nombre d'échantillons de chaque corpus, décomposé suivant le type d'écriture.

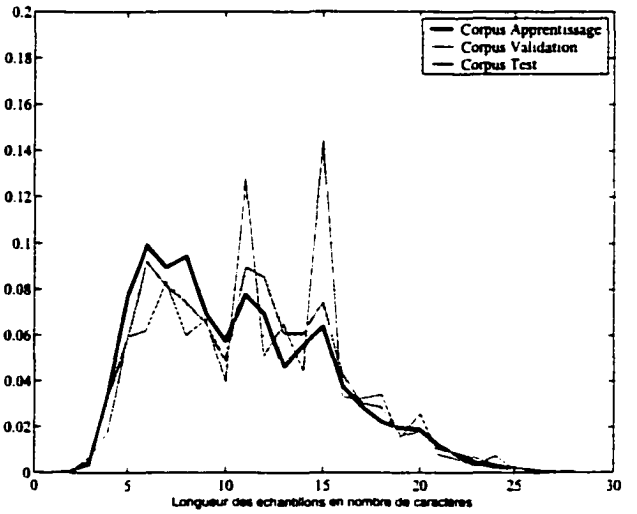


FIGURE 23 Distribution des échantillons de la base SRTP en fonction de leur longueur.

2.3.8.2 Caractérisation de la base SRTP

Afin de caractériser les différents corpus de données extraits de la base SRTP, nous avons évalué la distribution des échantillons en fonction de leur longueur en caractères [58]. Elle est présentée sur la figure 23. La longueur moyenne des échantillons de chaque corpus a également été calculée. Une dernière donnée a été estimée : le nombre d'intitulé différents, de manière à caractériser la diversité au sein de chaque corpus. Ces informations sont présentées dans le tableau II.

TABLEAU II

Caractérisation des corpus de données de la base SRTP.

Corpus	*Apprentissage*	*Validation*	*Test*
Longueur moyenne	10,69	11,64	11,14
Nombre d'intitulés différents	4 814 *40%*	1 392 *40%*	2 540 *54,3%*

L'analyse des distributions et des valeurs moyennes montrent que les trois corpus sont similaires. Celui de validation a une valeur moyenne un peu plus élevé que les autres,

deux pics peuvent également être remarqués pour des longueurs d'échantillons de 11 et 15 caractères. En fait, ces derniers sont uniquement plus importants pour le corpus de validation que pour les autres corpus. Globalement cette remarque signifie que la base de données contient un grand nombre d'exemples de longueur 11 et 15 caractères. Sur le tableau II, le pourcentage d'intitulés différents dans chaque corpus a été spécifié. Pour les corpus d'apprentissage et de validation, nous pouvons conclure que chaque intitulé apparaît en moyenne deux fois et demie.

2.3.9 Le lexique global

Comme mentionné dans la première partie de ce chapitre, un vocabulaire doit être défini pour tout système de reconnaissance de l'écriture. Dans notre cas il s'agit de celui des noms de communes françaises. L'ensemble de ces derniers est regroupé dans un lexique que nous qualifions de global. Il contient également certaines variantes d'écriture comme par exemple la contraction du mot `Saint` en `St`.

Le lexique global utilisé contient exactement 37 098 entrées. La longueur moyenne des noms de commune le composant est de 12 caractères. Cette valeur est du même ordre, bien que légèrement supérieure, que celles des différents corpus de données. La conclusion découlant de cette remarque est que nos corpus de données sont bien représentatifs de notre problème de reconnaissance.

2.4 Évaluation du système standard

Le but de notre projet étant d'améliorer le système de reconnaissance de base du SRTP, nous avons effectué une analyse approfondie de ces performances afin de mettre en évidence ces points faibles et ainsi orienter nos travaux. Elle a été le sujet d'un rapport technique [57] et également d'une publication [58]. Dans la suite de cette section, nous allons présenter les différentes évaluations réalisées ainsi que l'orientation choisie de notre projet.

2.4.1 Comparaison des performances sur deux bases de données

Lors de la prise en main du système, nous avons évalué ses performances sur deux bases de données. En effet, en plus de la base SRTP (voir section 2.3.8), nous avons utilisé une partie (les sous-bases **BD** et **BS**) de la base de données fournie par le *Center of Excellence on Document Analysis and Recognition* (CEDAR) [71]. Elle est composée d'images scannées de courrier réel américain (300 ppi, 256 niveaux de gris). L'ensemble des échantillons est classé en deux catégories : apprentissage ou test. Ceux qualifiés d'apprentissage ont été divisés en deux sous-ensembles : le corpus d'apprentissage et celui de validation. La base de données construite contient alors 3 108 échantillons d'apprentissage, 529 de validation et 377 de test. Les images ont été binarisées à l'aide de l'algorithme d'Otsu [125]. Un point important à mentionner est qu'un nom de ville est décomposé en mots. Par exemple, l'intitulé "*New York City*" est décomposé en trois images indépendantes. Ne disposant pas de l'information permettant de reconstruire l'image originale, un tel exemple correspond alors à trois échantillons. Une conséquence directe de ce phénomène est que la taille moyenne des exemples de cette base (6,7 caractères) est bien inférieure à celle de la base SRTP (11,2 caractères).

Pour chaque exemple de test de la base CEDAR, trois lexiques sont fournis. Ils ont été construits dynamiquement, en fonction de la confiance accordée à la reconnaissance du code postal, de la même manière que notre protocole expérimental (voir section 2.3.7). Les trois lexiques sont donc de tailles différentes. Afin de les caractériser, le nombre d'entrées moyen a été évalué. Le plus petit lexique contient en moyenne 13,1 entrées et les deux autres respectivement 102,8 et 886,2 entrées. Nous pouvons constater que ces valeurs sont du même ordre que celles proposées dans notre protocole expérimental.

Afin de préserver la continuité avec les travaux précédents [37, 38], nous avons utilisé un lexique global différent de celui décrit dans la sous-section 2.3.9. Il contient uniquement les intitulés présents dans l'ensemble de la base de données, soit 6 815 entrées.

Deux systèmes de reconnaissance ont été construits en effectuant un apprentissage à l'aide de chaque base de données. L'évaluation de leurs performances a été réalisée en utilisant le lexique décrit ci-dessus pour le système SRTP et les lexiques fournis pour le système CEDAR. Les taux de reconnaissance obtenus sont présentés dans le tableau III.

TABLEAU III

Taux de reconnaissance obtenus sur les bases de données SRTP et CEDAR.

Taille du lexique utilisé	10	100	1 000
Système SRTP	98,9%	95,3%	86,9%
Système CEDAR	88,9%	75,8%	56%

L'analyse des valeurs montre une grande différence de performance entre les deux systèmes. Le système utilisé étant développé sur la base de données SRTP, certaines étapes des pré-traitements ne sont pas satisfaisantes pour les données CEDAR, comme par exemple la suppression de lignes horizontales soulignant le nom de ville. Un autre facteur important est le faible nombre d'échantillons de la base CEDAR. La différence de longueur des échantillons entre les deux bases influence également les performances. Afin de mettre en évidence ce dernier point, nous avons représenté sur la figure 24 les taux d'erreur pour les deux systèmes, en fonction de la longueur des échantillons des bases de test. Globalement nous pouvons remarquer que plus la taille des échantillons est grande plus le taux d'erreur est faible.

2.4.2 Évaluation de l'influence de différents paramètres

Afin de valider ces hypothèses, nous avons réalisé différentes séries d'expériences. Premièrement l'influence du nombre d'échantillons utilisés au cours de l'apprentissage a été étudié pour le système SRTP. Différents systèmes ont été construits en utilisant un nombre croissant d'exemples (de 1 500 au maximum). Afin d'étudier l'influence de la longueur des échantillons, une nouvelle base de données a été construite en décomposant les noms

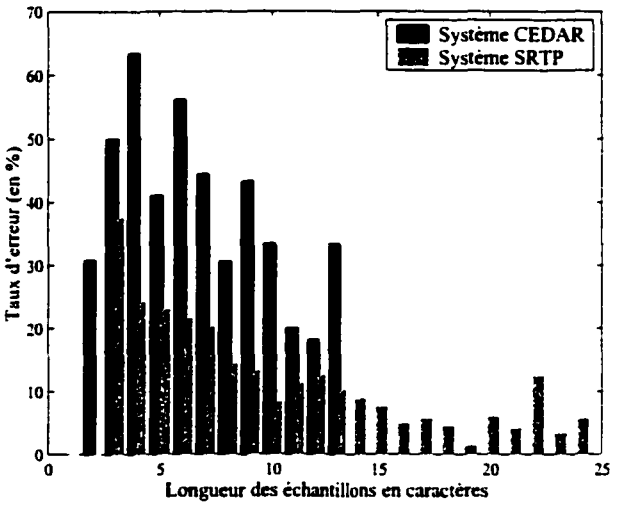


FIGURE 24 Taux d'erreur des systèmes CEDAR et SRTP en fonction de la longueur des échantillons.

de ville de la base SRTP en mots. Cette dernière contient 18 343 mots simples d'une longueur moyenne de 5,8 caractères. La distribution des exemples est similaire à celle de la base CEDAR, mis à part un pic au niveau des mots de deux caractères, dû à la présence d'articles ou prépositions : `le`, `la` ,`de`, `et`,..., dans le vocabulaire des noms de communes françaises. Afin de la différencier de la base de données classique, nous la qualifierons de "Mots Simples". L'influence d'un dernier paramètre a été étudié : la longueur moyenne des entrées du lexique utilisé pour évaluer un échantillon de test. Pour cela, un nouveau lexique a été construit de la même manière que la base de données, c'est-à-dire en décomposant les noms de ville du lexique SRTP en mots simples. Ce dernier contient 7 636 entrées d'une longueur moyenne de 7,1 caractères.

Sur la figure 25 sont présentés les taux de reconnaissance obtenus pour les trois séries d'expériences réalisées, lors de l'utilisation d'un lexique de taille 1 000. La première, marquée de triangles, correspond à l'utilisation de la base de données et du lexique classique. Elle permet d'évaluer l'influence du nombre d'échantillons d'apprentissage. Pour la seconde série, c'est la base de données SRTP Mots Simples qui a été utilisée et le lexique

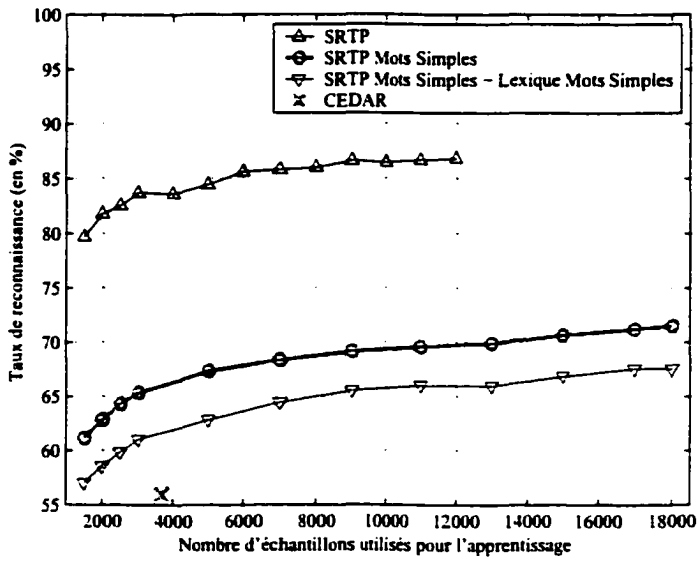


FIGURE 25 Influence du nombre d'échantillons d'apprentissage et de la stratégie Mot / Phrase sur les systèmes de reconnaissance – Expériences réalisées avec un lexique de taille 1 000.

classique. La courbe associée est marquée par des cercles sur la figure 25. La troisième série, représentée à l'aide de triangles inversés, correspond à l'utilisation de la base et du lexique SRTP Mots Simples. Finalement la performance obtenue avec le système CEDAR (voir tableau III) est représentée par une croix. À l'annexe 3.4, les mêmes données sont présentées mais lors de l'utilisation de lexique de tailles 10, 100 et 1 000.

Une première constatation est que l'augmentation du nombre d'échantillons d'apprentissage conduit bien sûr à une amélioration des performances, quelle que soit la série d'expériences. Cependant, la réduction du nombre d'échantillons pour le système SRTP conduit toujours à une performance bien meilleure que celle obtenue par le système CEDAR.

La comparaison des deux premières séries d'expériences permet d'évaluer directement l'influence de la longueur moyenne des échantillons à reconnaître sur les performances globales du système. Comme nous pouvons le constater elle est importante puisque les taux de reconnaissance sont de plus de quinze points inférieurs lors de l'utilisation de

la base de mots simples. Cette constatation confirme que pour le système standard les exemples longs sont plus faciles à reconnaître que les courts. En effet la séquence de primitives extraites est de longueur proportionnelle à celle de l'exemple à tester. L'explication est que pour une séquence longue il est plus probable que la reconnaissance s'appuie sur des points d'ancrage forts, comme des primitives très discriminantes ou encore la présence d'espaces inter-mots.

L'influence de la longueur des entrées du lexique global peut être évaluée en comparant les deux dernières courbes. Ce paramètre influence également de manière non négligeable les performances globales du système, puisque la différence entre les deux courbes est de l'ordre de 5%. Ce paramètre intervient au niveau de la dispersion en terme de longueur des entrées tirées aléatoirement lors du test d'un échantillon. Si elle est faible, c'est-à-dire que les entrées du lexique utilisé sont de longueur similaire, les probabilités associées à chacune d'elles seront du même ordre et la reconnaissance sera plus difficile. Par contre plus la dispersion est grande plus la reconnaissance sera facilitée.

L'analyse réalisée a permis de montrer et quantifier l'influence de la taille en caractères des exemples ainsi que la dispersion des entrées du lexique utilisé, toujours en caractères. La conclusion de cette analyse est que pour la reconnaissance d'échantillons courts il faut avoir une caractérisation très discriminante de l'information présente dans l'image, c'est-à-dire des ensembles de primitives les plus discriminants possibles.

2.4.3 Estimation directe des paramètres

L'apprentissage du système de reconnaissance se fait de manière automatique par l'intermédiaire de l'algorithme de Baum-Welch. Afin de le mettre en œuvre il faut présenter au système l'intitulé exact, c'est-à-dire la chaîne de caractères, correspondant à l'échantillon. De ce fait le système utilise un apprentissage dit supervisé. Suite aux pré-traitements, l'image est segmentée en un nombre de graphèmes plus ou moins important. L'algorithme d'apprentissage effectue alors automatiquement l'alignement entre les graphèmes et les

caractères, en fonction de la modélisation choisie (dans notre cas un modèle caractère peut absorber entre 0 et 3 graphèmes), afin de calculer les probabilités associées aux paramètres du système.

Il est possible d'obtenir les valeurs de ces paramètres de manière différente. La connaissance de l'alignement exact entre la chaîne de caractères et les couples graphème/primitives pour l'ensemble des échantillons d'apprentissage permettrait d'effectuer une estimation directe des paramètres de nos modèles, par comptage d'occurrences. L'application de l'algorithme de Viterbi (voir section 1.1.3.2) permet d'obtenir l'alignement conduisant à la plus forte probabilité. Cependant nous ne pouvons pas être sûrs que l'attribution des graphèmes aux différents caractères est correcte. En fait le seul moyen garantissant l'exactitude de l'alignement est qu'un opérateur humain effectue une vérification.

Une interface graphique [32] a été construite dans ce but. Elle permet de visualiser l'image segmentée en graphème ainsi que l'alignement caractère/graphème obtenu suite à l'application de l'algorithme de Viterbi. Sa fonction principale est de permettre la modification de cet alignement afin d'attribuer le nombre réel de graphèmes à chaque caractère.

Cet outil a été utilisé afin de corriger l'alignement obtenu par le système standard SRTP sur notre corpus d'apprentissage (les 10 000 premiers échantillons, soit plus de 105 000 caractères). Un même opérateur a réalisé l'ensemble de cette vérification. Cela a permis d'effectuer les corrections d'alignement de manière constante. L'alignement de seulement 20% des échantillons a été modifié. Cela signifie que dans une grande majorité des cas, le système effectue correctement le regroupement de graphèmes en caractères et donc valide le choix de notre modélisation.

2.4.3.1 Analyse de la segmentation des caractères

La vérification humaine des alignements a permis également de récolter différentes informations. Nous présentons dans le tableau IV les statistiques concernant la segmentation des caractères en graphèmes.

TABLEAU IV

Statistique sur la segmentation des caractères en graphèmes pour le corpus d'apprentissage de la base SRTP.

Nombre de segments par caractère	0	1	2	3	4	5
Nombre de caractères concernés	3 286	74 647	26 028	1 646	64	3
Pourcentage du corpus	3,11%	70,64%	24,63%	1,56%	0,06%	0,003%

Nous pouvons remarquer que pour plus de 95% des caractères un ou deux segments sont utilisés. Cela traduit un comportement normal de notre algorithme de segmentation (voir section 2.3.2) qui s'appuie sur la détection des minima du contour supérieur. Les cas de sur-segmentation en trois graphèmes ne sont pas très fréquents. Comme nous l'avons mentionné auparavant, seulement quelques caractères sont touchés par ce phénomène. Les cas de sur-segmentation en quatre et cinq graphèmes sont problématiques puisqu'ils ne sont pas pris en compte par notre modélisation. De ce fait le système doit effectuer un décalage dans l'alignement et il attribuera obligatoirement certains (au moins un) graphèmes à des caractères qui ne leur correspondent pas. Un échantillon contenant un tel cas risque fort de ne pas être reconnu. Nous présentons sur la figure 26 des lettres segmentées en quatre graphèmes. Les causes principales de ce phénomène sont l'utilisation de lettre majuscule stylisée en début de mots, la fragmentation de l'image lors de la phase de binarisation et la grande sensibilité de l'algorithme de segmentation. Cependant ce phénomène de sur-segmentation n'intervient que dans très peu de cas. Il n'est donc pas nécessaire de prendre

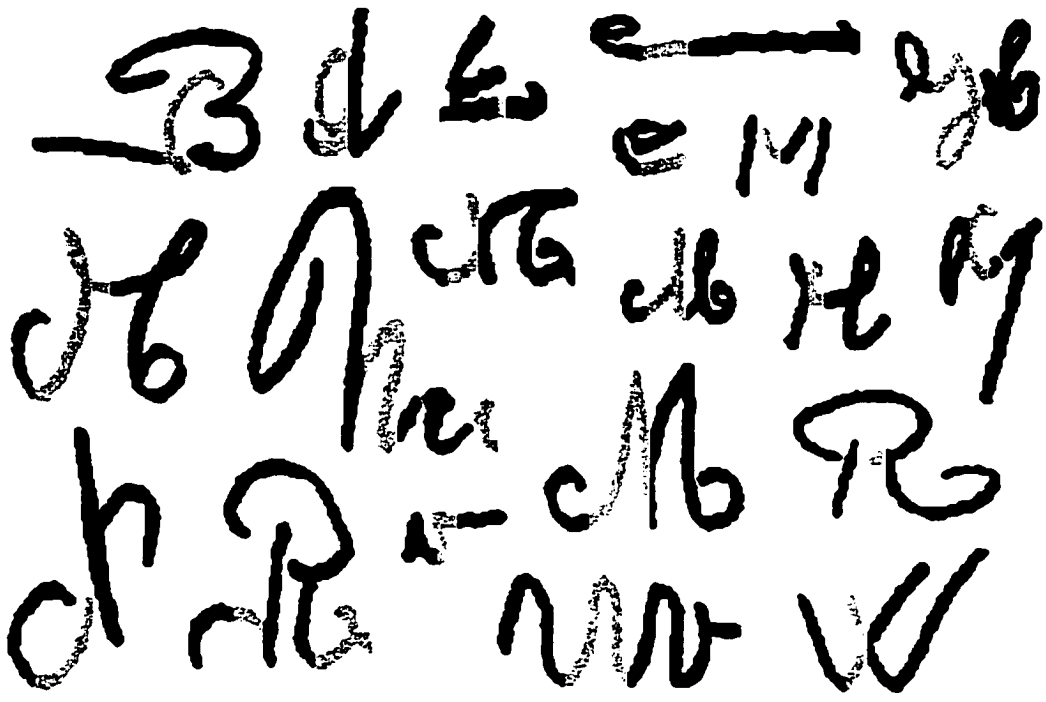


FIGURE 26 Différents exemples de lettre sur-segmentée en quatre graphèmes.

en compte ces cas directement dans la modélisation comme pour la sur-segmentation en trois graphèmes.

Comme nous pouvons le remarquer dans le tableau IV, la sous-segmentation de caractères est deux fois plus importante que la sur-segmentation en trois graphèmes. De plus, une analyse des confusions du système de reconnaissance a montré que la sous-segmentation est responsable d'environ 7% des erreurs. La modélisation utilisée permet de prendre en compte ce phénomène. Cependant, pour certains échantillons, il est possible que le système introduise un décalage dans l'alignement entre les graphèmes et les caractères et donc une certaine confusion lors de la reconnaissance.

Sur la figure 27 sont présentés un certain nombre d'exemples de sous-segmentation. Il est certain que l'algorithme de segmentation a une part importante de responsabilité dans

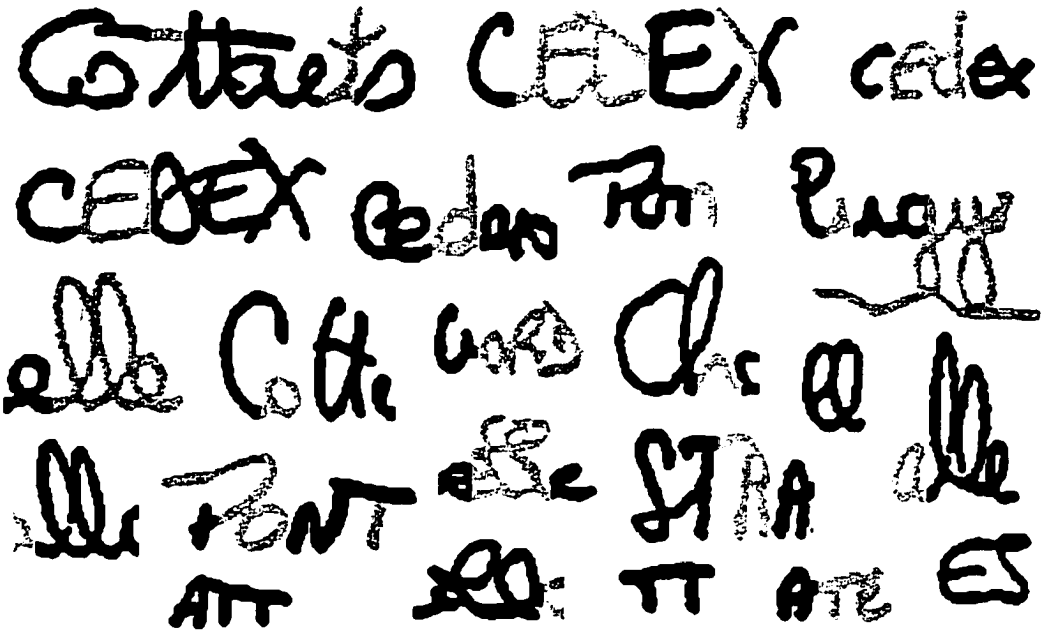
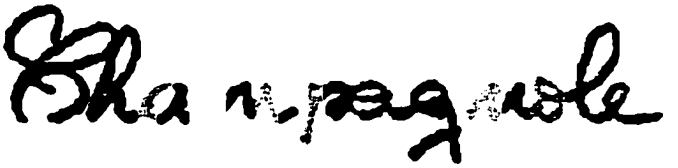


FIGURE 27 Différents exemples de caractères sous-segmentés.

l'apparition de ce phénomène. Comme nous l'avons déjà mentionné dans la section 2.3.2, il n'existe pas d'algorithme de segmentation parfait. Son développement doit s'accommoder d'un certain nombre de compromis. En l'occurrence, nous interdisons la segmentation d'un mot à l'emplacement d'une boucle, car cette dernière est une primitive de base importante pour la reconnaissance. Si deux caractères d'un mot sont liés à deux endroits au moins, une boucle est formée. Le principale responsable de ce phénomène est le scripteur lorsqu'il ne lève pas son outil, qu'il écrit de manière trop compacte ou rapidement. La première étape de nos pré-traitements a également une influence. En effet avant toute transformation, le contour externe de l'image lui est ajoutée. Le but de cette étape est de retrouver la connexité de certaine masse de pixels perdue lors de la binarisation. Cependant elle peut conduire à lier deux caractères si la distance les séparant n'est pas supérieure à deux pixels.



Intitulé : Champagnole

FIGURE 28 Échantillon contenant un cas de sous-segmentation (Ch) et un cas de sur-segmentation (m).

Le phénomène de sous-segmentation, malgré qu'il soit pris en compte par la modélisation reste problématique. Il est possible d'envisager l'utilisation de modèles particuliers pour les cas les plus fréquents comme ll, ou, tt, TT, ... Une autre possibilité afin de pallier le problème de sous-segmentation est bien sûr l'amélioration de l'algorithme de segmentation.

Lors de la vérification de l'alignement des échantillons, d'autres données ont été récoltées. Dans l'annexe 2, nous présentons le nombre d'occurrences de chaque caractère modélisé par le système, rencontrés dans les échantillons vérifiés. De plus les phénomènes de sous- et sur-segmentation ont été quantifiés pour chacun d'eux. Les valeurs obtenues permettent de mieux évaluer l'influence de l'algorithme de segmentation.

2.4.3.2 Performances du système obtenu par évaluation directe des paramètres

Une fois que nous disposons de l'alignement exact entre les caractères et les graphèmes, un système de reconnaissance peut être construit. En effet la connaissance de l'alignement exact revient à connaître le chemin emprunté dans le graphe associé à l'échantillon. Cela permet d'effectuer le comptage du franchissement des différentes transitions, en fonction du modèle caractère (voir figure 18) et du couple de primitives perceptuelle/bâton. Les valeurs obtenues permettent l'estimation des probabilités de transitions. Pour cela une seule propriété doit être respectée : la somme des probabilités sortant d'un nœud doit être égale à 1. L'estimation des différents paramètres a_{ijk} et $a_{ij\Phi}$ de chaque modèle caractère

(voir section 1.1.2) revient à utiliser les relations énoncées par les équations 1.30 et 1.31. Plus précisément elles peuvent être ré-écrites de la manière suivante :

$$a_{ijk} = \frac{\text{nombre de franchissements de la transition } s_i \rightarrow s_j \text{ en émettant le symbole } v_k}{\text{nombre de visites de l'état } s_i}$$

$$a'_{ij\Phi} = \frac{\text{nombre de franchissements de la transition } s_i \rightarrow s_j \text{ en émettant le symbole } \Phi}{\text{nombre de visites de l'état } s_i}$$

Comme nous pouvons le constater dans l'annexe 2, certains caractères ne sont rencontrés que peu de fois et d'autres jamais. Cela signifie qu'une partie des paramètres du système seront nuls. Afin de pallier le problème, une valeur minimale est donnée aux probabilités ne pouvant pas être estimées par comptage. Elle a été fixée à 10^{-5} qui est inférieure à la plus petite valeur obtenue lors de l'estimation. Une étape de normalisation des paramètres est alors réalisée de manière à conserver la somme des probabilités égale à 1. Après cette phase nous disposons d'un système de reconnaissance opérationnel.

Afin de comparer les performances obtenues entre le système standard résultant de l'étape d'apprentissage (voir tableau III) et celui construit par évaluation directe, le même corpus de test et le même lexique ont été utilisés. L'influence du nombre d'échantillons d'apprentissage sur les performances a également été évaluée. Nous présentons sur la figure 29 les taux de reconnaissance obtenus lors de l'utilisation de lexiques de taille 10, 100 et 1 000, pour des systèmes utilisant entre 1 500 et 10 000 échantillons d'apprentissage. Les taux de reconnaissance obtenus par le système SRTP ont été reportés dans un but de comparaison. Le corpus d'apprentissage n'étant pas étiqueté en entier, les courbes associées au système SRTP ont des points supplémentaires.

L'analyse des différentes courbes de la figure 29 montre clairement que l'apprentissage automatique, c'est-à-dire en utilisant l'algorithme de Baum-Welch, conduit à de meilleures performances lorsque le nombre d'échantillons d'apprentissage est faible (inférieur à 5 000). Cependant la différence de performances entre les deux techniques s'atténue avec l'aug-

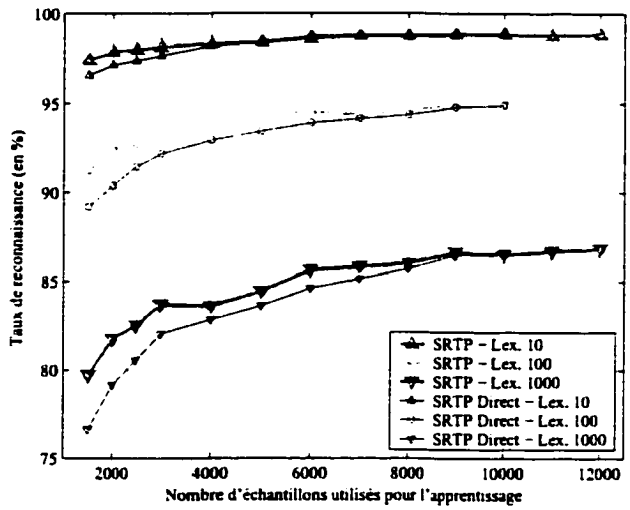


FIGURE 29 Comparaison des stratégies "Apprentissage Automatique" / "Évaluation Directe des paramètres" – Expériences réalisées avec des lexiques de taille 10, 100 et 1 000.

mentation du nombre d'échantillons d'apprentissage, jusqu'à être quasi-nulle. La supériorité de l'apprentissage automatique vient principalement du fait qu'il effectue une estimation de l'ensemble des paramètres de nos modèles. En effet au départ une initialisation aléatoire attribue une valeur à tous les paramètres. Ces valeurs sont ensuite utilisées de manière à ré-estimer l'ensemble des probabilités. Par contre lors de l'estimation directe, certains paramètres ne peuvent pas être calculés, une valeur minimale leur est alors attribuée. Cette différence se traduit par une probabilité en sortie du modèle de reconnaissance plus faible pour le système à estimation directe. Une analyse statistique a permis de constater que pour 90% des échantillons du corpus de test, la probabilité en sortie du modèle est plus forte pour le système standard.

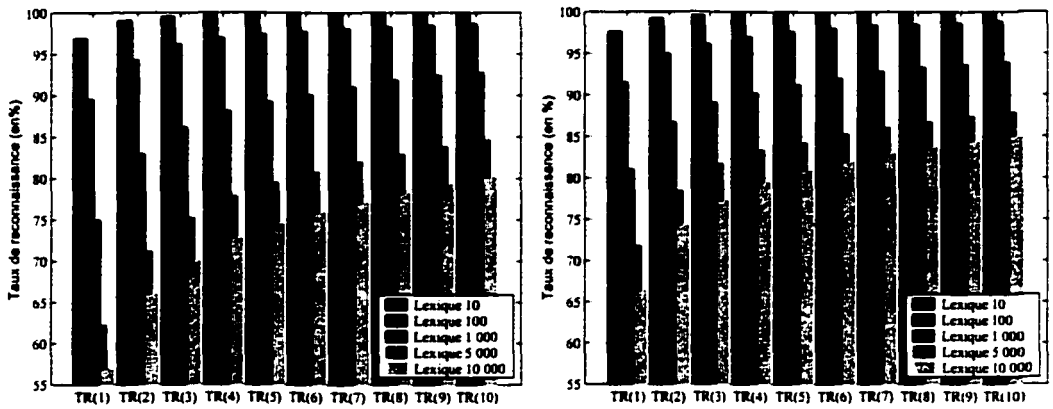
En conclusion nous pouvons dire qu'il est plus intéressant d'effectuer un apprentissage automatique lorsque la quantité d'échantillons d'apprentissage est faible. Il serait intéressant d'effectuer l'étiquetage du corpus d'apprentissage de manière à compléter cette étude et constater le comportement des courbes associées au système à estimation directe.

2.4.4 Performances sur la base de données standard

Afin de mieux évaluer l'influence des modifications que nous allons apporter au système, il est intéressant d'utiliser des lexiques de plus grande taille. En effet en réalisant l'alignement d'un échantillon avec un plus grand nombre d'entrées du lexique, le système est confronté à un nombre de difficultés plus important. Les probabilités que le système rencontre des alignements entre primitives/graphèmes et caractères non représentés dans le corpus d'apprentissage sont plus importantes. Les taux de reconnaissance associées à l'utilisation d'un lexique de grande taille seront donc plus faibles, comme nous avons déjà pu le remarquer avec les expériences utilisant 10, 100 et 1 000 entrées. Cependant ils seront également plus sensibles aux améliorations que nous pouvons apporter.

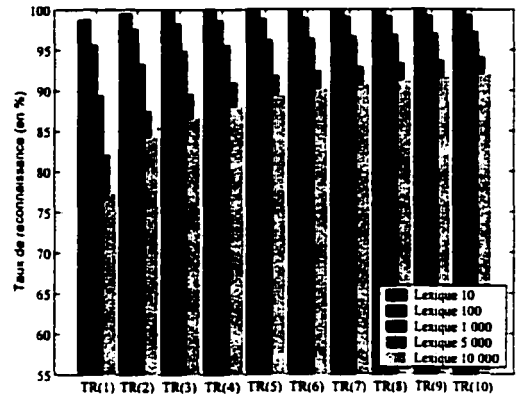
L'évaluation des performances du système standard a donc été réalisée en utilisant 10, 100, 1 000, 5 000 et 10 000 entrées. Le lexique global utilisé lors de ces tests est celui présenté dans la section 2.3.9. Nous avons dans un premier temps évalué les performances des deux ensembles de primitives individuellement puis leur prise en compte conjointe. Ces évaluations ont été réalisées en fonction des différents types d'écriture : bâton, cursif et mixte. Dans le cas où un exemple est mal reconnu, nous avons également évalué sa position dans la liste de solutions fournies par le système. Les taux de reconnaissance obtenus sont détaillés dans l'annexe 3.

Sur la figure 30 sont présentés les taux de reconnaissance obtenus pour les différents systèmes construits. La notation "Système Perceptuel" fait référence au système utilisant l'ensemble de primitives perceptuelles, par opposition au "Système Bâton" qui utilise le second ensemble de primitives. Finalement nous utiliserons la notation "Système Standard" pour faire référence à l'utilisation conjointe des deux ensembles de primitives. La notation TR(x) signifie que l'intitulé exact de l'exemple testé fait partie des x premières solutions proposées par le système de reconnaissance. De manière à compléter l'analyse, nous pré-



(a) Système Perceptuel

(b) Système Bâton

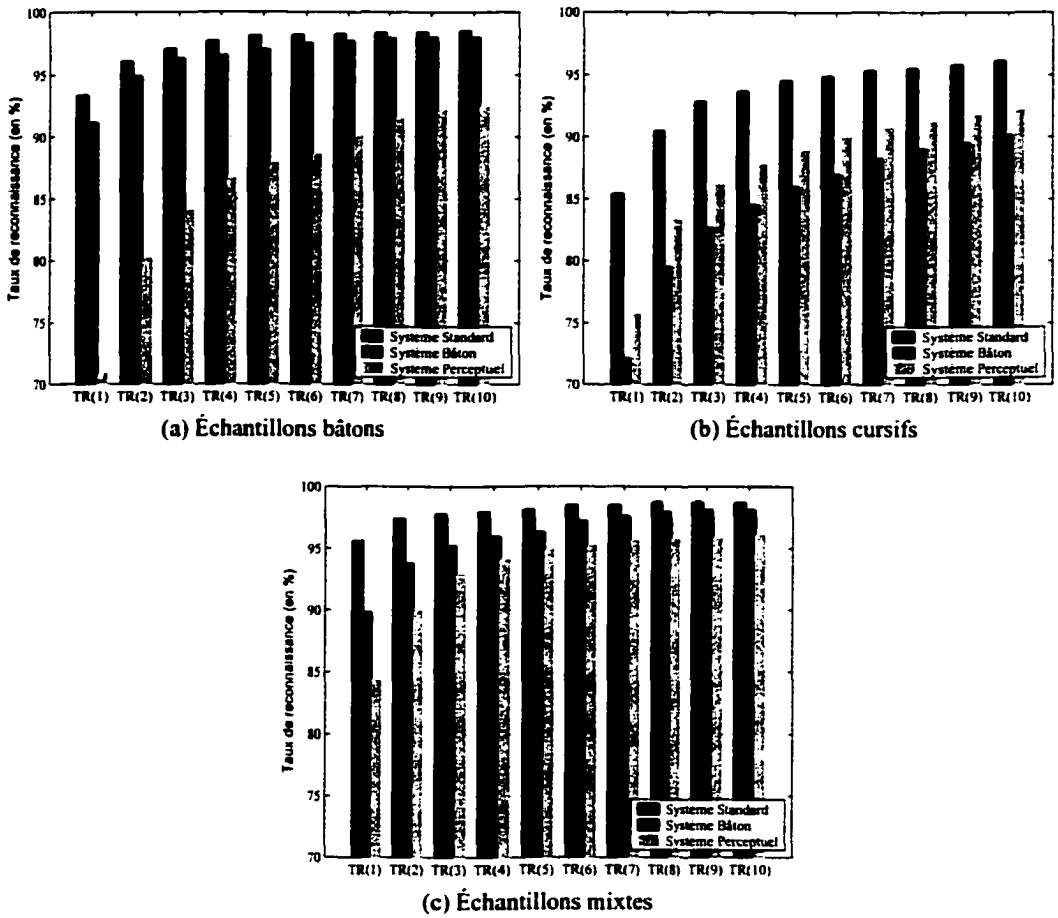


(c) Système Standard

FIGURE 30 Évaluation des performances des systèmes de reconnaissance Bâtons, Perceptuel et Standard – Expériences réalisées avec des lexiques de taille 10, 100, 1 000, 5 000 et 10 000.

sentons sur la figure 31 les taux de reconnaissance lors de l'utilisation d'un lexique 1 000 en fonction des différents types d'écriture pris en compte.

A l'aide de la première série de graphiques, nous pouvons constater la différence de performances entre les systèmes Perceptuel et Bâton. Ce dernier obtient globalement des taux de reconnaissance plus élevés. Plus la taille du lexique utilisé est grande plus cette différence est flagrante. Ce phénomène s'explique par le fait que le Système Bâton est très performant pour les échantillons bâtons comme nous pouvons le constater sur la figure



(a) Échantillons bâtons

(b) Échantillons cursifs

(c) Échantillons mixtes

FIGURE 31 Performances des systèmes de reconnaissance Bâton, Perceptuel et Standard en fonction du type d'écriture des échantillons – Expériences réalisées avec un lexique de taille 1 000.

31(a). Par contre, le Système Perceptuel, bien qu'il soit plus performant que le Bâton sur les échantillons cursifs, n'obtient pas des taux de reconnaissance aussi élevés (voir figure 31(b)). De plus le nombre d'échantillons cursifs dans notre base de données étant plus important que celui de bâtons, la différence de performances entre les deux systèmes est amplifiée.

L'analyse des différents graphiques permet également de constater l'intérêt de la technique de prise en compte conjointe des ensembles de primitives. En effet cette dernière permet d'obtenir des taux de reconnaissance supérieurs à ceux obtenus par les systèmes Perceptuel et Bâton, quel que soit le type d'écriture des échantillons. Cette amélioration est amplifiée lors de l'utilisation de lexiques de grande taille. Cette technique de prise en compte de l'information implique une augmentation importante du nombre de primitives et donc du nombre de paramètres de nos modèles. Cela permet de mieux appréhender la grande variabilité de l'écriture.

Une analyse des graphiques de la figure 31 permet de mieux comprendre l'amélioration des performances apportée par la prise en compte conjointe des deux ensembles de primitives. Pour les échantillons bâtons, nous pouvons constater que l'ensemble bâton est très performant et que l'apport des primitives perceptuelles ne permet d'augmenter que très faiblement les taux de reconnaissance. Par contre, l'apport des primitives bâtons à l'ensemble perceptuel permet une nette amélioration des performances pour les exemples cursifs. Concernant les échantillons mixtes, il est plus difficile de tirer des conclusions et ce pour deux raisons. Premièrement ils sont en nombre réduit et deuxièmement nous ne connaissons pas exactement la composition de ces exemples en termes de caractères majuscules et minuscules. Cependant nous pouvons remarquer que globalement ce sont pour ces exemples que les taux de reconnaissance du Système Standard sont les plus élevés. Malgré la nette amélioration apportée par la prise en compte conjointe des ensembles de primitives, les taux de reconnaissance des échantillons cursifs sont les plus faibles. De part sa nature, ce type d'écriture est et reste le plus difficile à reconnaître.

2.4.5 Évaluation du pouvoir discriminant des primitives

L'évaluation du système de reconnaissance a été effectuée uniquement sur la base de ces taux de reconnaissance, c'est-à-dire une évaluation globale. Un système de reconnaissance est en fait la combinaison d'un certain nombre de modules. Il serait donc intéressant d'ef-

fectuer une évaluation objective de chacun d'eux. El-Yacoubi propose une telle évaluation pour les modules de pré-traitements et d'extraction de primitives dans [38].

Lors de l'analyse des performances du système sur deux bases de données, nous avons conclu sur l'importance du pouvoir discriminant individuel des primitives. Il est donc utile d'évaluer objectivement les deux ensembles utilisés par le système standard. Pour cela nous proposons d'utiliser un indicateur entropique introduit dans la section 1.2.3 : la perplexité conditionnelle des classes étant donnée une primitive $PP(C|F = f_j)$. Cette dernière est obtenue à partir de l'entropie conditionnelle définie par l'équation 1.65 de la manière suivante :

$$PP(C|F = f_j) = 2^{H(C|F=f_j)} \tag{2.10}$$

Cet indicateur permet d'obtenir une évaluation du pouvoir discriminant de chaque primitive individuellement. L'ensemble de primitives peut être évalué globalement par l'intermédiaire des valeurs obtenues et des fréquences d'occurrence comme exposé dans l'équation 1.66. La pertinence de l'utilisation de cet indicateur a été discutée dans la section 1.2.4.

Afin d'évaluer nos ensembles de primitives à l'aide de cet indicateur, nous devons disposer des probabilités $\Pr(\omega_i|f_j)$ et $\Pr(f_j)$. Pour cela l'algorithme de Viterbi est utilisé. Comme nous l'avons vu dans la section 1.1.3.2 ce dernier permet d'obtenir l'alignement entre une séquence de primitives et une de caractères conduisant à la plus forte probabilité. Cette procédure est appliquée pour chaque échantillon de notre corpus de validation, en utilisant l'intitulé exact correspondant. Durant cette phase, un comptage d'occurrences des différents couples primitive/classe est réalisé. Après normalisation des valeurs de ces compteurs, nous obtenons les probabilités conjointes $\Pr(\omega_i, f_j)$, dont certaines sont bien sûr nulles. Ces dernières permettent d'estimer les probabilités marginales $\Pr(\omega_i)$ et $\Pr(f_j)$

de la manière suivante :

$$\Pr(\omega_i) = \sum_{j=1}^{N_F} \Pr(\omega_i, f_j) \quad \text{pour} \quad i = 1, 2, \ldots, N_C \tag{2.11}$$

$$\Pr(f_j) = \sum_{i=1}^{N_C} \Pr(\omega_i, f_j) \quad \text{pour} \quad j = 1, 2, \ldots, N_F \tag{2.12}$$

Les probabilités conditionnelles sont alors obtenues par l'intermédiaire de l'équation :

$$\Pr(\omega_i | f_j) = \frac{\Pr(\omega_i, f_j)}{\Pr(f_j)} \quad \text{pour} \quad i = 1, 2, \ldots, N_C \text{ et } j = 1, 2, \ldots, N_F \tag{2.13}$$

Une fois ces quantités obtenues, nous pouvons estimer l'entropie conditionnelle de chaque primitive, qui permet alors de calculer la perplexité associée. Ces valeurs permettent également l'évaluation globale de l'ensemble de primitives.

Nous présentons respectivement dans les tableaux V et VI l'évaluation des primitives perceptuelles et bâtons. La seconde colonne correspond à la fréquence d'occurrence des différentes primitives. Nous avons indiqué dans la troisième colonne de ces tableaux le nombre de classes de notre modélisation (209 au total) que chaque primitive a caractérisé au moins une fois. Les deux colonnes suivantes présentent les valeurs d'entropie et de perplexité conditionnelle des primitives : $H(C|F = f_j)$ et $PP(C|F = f_j)$. La dernière ligne des tableaux correspond à l'évaluation globale de l'ensemble de primitives.

L'analyse des valeurs des deux tableaux montrent que certaines primitives sont bien plus discriminantes que d'autres. En particulier, concernant l'ensemble perceptuel nous pouvons remarquer que la primitive la moins discriminante "-" est la plus fréquente. En effet cette dernière couvre plus de 50% de la distribution. D'après sa définition (voir annexe 1), elle correspond à l'absence des caractéristiques de base. Sa fréquence d'apparition élevée s'explique en partie par le fait que pour l'écriture bâton les dépassements n'existent pas. Cela signifie qu'une grande partie des graphèmes associés à ce type d'écriture est caracté-

TABLEAU V

Évaluation du pouvoir discriminant de l'ensemble de primitives perceptuelles – Présentation des primitives par ordre décroissant de perplexité conditionnelle.

Primitives	Fréquence (en %)	Nb. classes touchées	Entropie	Perplexité
-	51,90	164	5,93	61,14
h	5,75	127	5,63	49,51
H	11,81	138	5,47	44,34
t	0,38	62	5,38	41,59
B	1,33	71	5,03	32,72
f	0,39	51	4,93	30,57
y	0,25	43	4,83	28,48
T	0,49	48	4,56	23,62
b	1,09	57	4,52	22,99
d	1,28	65	4,39	20,93
D	0,80	55	4,33	20,18
S	0,85	48	4,20	18,42
L	3,05	70	4,17	17,99
F	0,22	27	4,15	17,72
P	0,19	26	3,91	15,06
O	6,71	73	3,91	15,01
o	11,02	115	3,70	13,01
K	1,20	46	3,35	10,19
Q	0,33	28	3,29	9,81
0	0,27	23	3,27	9,63
q	0,03	9	2,85	7,22
p	0,05	11	2,85	7,20
Z	0,44	17	2,78	6,88
z	0,05	9	2,50	5,65
G	0,10	9	2,23	4,69
g	0,01	4	1,79	3,46
I	0,01	2	0,92	1,89
Global			**5,26**	**38,28**

risée par la primitive "-". De plus l'absence de dépassements et de boucles est également rencontrée pour un grand nombre de graphèmes de l'écriture cursive, ceux provenant des lettres : c, i, m, n, r, s, u, v, w et x. Une confirmation du très faible pouvoir discriminant de cette primitive est l'écart important en terme de valeur de perplexité avec la seconde primitive. Pour l'ensemble des couples de primitives consécutives du tableau V un tel écart n'est plus rencontré.

TABLEAU VI

Évaluation du pouvoir discriminant de l'ensemble de primitives bâtons – Présentation des primitives par ordre décroissant de perplexité conditionnelle.

Primitives	Fréquence (en %)	Nb. classes touchées	Entropie	Perplexité
-	20,18	154	5,88	58,95
F	1,92	109	5,78	55,01
|	18,18	147	5,68	51,13
>	2,32	112	5,30	39,30
r	9,84	131	5,04	32,83
C	10,15	141	4,97	31,28
W	1,37	77	4,91	30,05
P	1,56	79	4,73	26,47
O	14,22	127	4,72	26,39
B	9,26	124	4,72	26,26
Q	1,11	59	4,65	25,08
m	0,22	34	4,46	22,05
A	5,21	113	4,46	21,96
E	4,45	89	3,20	9,21
Global			**5,14**	**35,22**

Au sujet de l'ensemble bâton, nous pouvons remarquer que la distribution des primitives est plus uniforme que pour l'ensemble perceptuel. La primitive la plus fréquente est également la moins discriminante. Une autre remarque est qu'une seule primitive de cet ensemble à une valeur de perplexité inférieure à 20, alors que pour l'ensemble perceptuel, la majorité des primitives a une valeur inférieure à ce seuil. Il est difficile de comparer directement ces valeurs car ce n'est pas la même distribution des primitives qui est évaluée. Cependant la différence entre les valeurs étant importante, nous pouvons énoncer le fait qu'une grande partie des primitives cursives sont plus discriminantes que la plupart des primitives bâton.

L'évaluation globale de la perplexité montre que l'ensemble bâton est plus discriminant que l'ensemble perceptuel. Cette constatation est en contradiction avec la remarque que nous venons de faire. De par sa définition l'entropie conditionnelle globale (voir équation 1.66) est une somme des valeurs individuelles pondérées par les fréquences d'occurrence des primitives. De ce fait la primitive "-" de l'ensemble perceptuel pèse beaucoup dans

l'évaluation globale de l'ensemble de primitive : $0,52 \times 5,93 = 3,08$, soit $58,5\%$ de la valeur d'entropie globale de cet ensemble. Une interprétation peut être donnée aux valeurs de perplexité globale. En effet nous pouvons dire que l'utilisation des deux ensembles de primitives permet de réduire la perplexité de la tâche de reconnaissance de 209 (nombre de classes de notre modélisation) à 38,28 et 35,22. L'objectif étant d'obtenir une valeur de 1, qui traduit un problème sans difficulté. Le pouvoir discriminant global des primitives issues du produit cartésien de nos deux ensembles a également été évalué :

$$PP(C|F_{P\times B}) = 20,70$$

Cette valeur de perplexité est bien inférieure à celles obtenues par les ensembles individuellement (38,28 pour l'ensemble perceptuel et 35,22 pour l'ensemble bâton). Ce résultat prouve qu'il existe une certaine complémentarité dans l'information apportée par les deux ensembles de primitives.

En conclusion nous pouvons dire que certaines primitives de nos ensembles sont discriminantes alors que d'autres ne le sont que très peu. C'est particulièrement le cas pour la primitive "-" de l'ensemble perceptuel. Nous avons également énoncé qu'une grande partie des primitives perceptuelles sont plus discriminantes que celles de l'ensemble bâton. L'utilisation conjointe des deux ensembles dans le Système Standard montre la complémentarité des deux ensembles.

2.5 Conclusions sur le système et directions de nos travaux

Dans les deux sections précédentes nous avons décrit les différents modules du système de reconnaissance de l'écriture manuscrite du SRTP et présenté l'analyse approfondie de ses performances. Cette dernière a été réalisée de manière à dégager une direction principale pour l'amélioration des performances du système. Cette section nous permettra d'exposer nos conclusions ainsi que l'orientation de nos travaux.

2.5.1 Conclusions sur les différents modules du système

Le premier module du système de reconnaissance est celui des pré-traitements. Il permet de modifier l'image afin de supprimer au maximum les variabilités intrinsèques à l'écriture. Une analyse des confusions du Système Standard a montré que ce module est responsable d'environ 10% des erreurs. Les causes imputées à ce module sont la déformation prononcée du tracé ou sa fragmentation, ainsi que la jonction de caractères. Son principal défaut vient de la détection et surtout du filtrage heuristique d'extremum. Les méthodes mises en œuvre sont globalement satisfaisantes mais peuvent être améliorées. Cependant, cette action nécessiterait un fort investissement en temps pour régler les problèmes associés à un nombre restreint d'échantillons. Cela se traduirait également par une faible amélioration des performances et une augmentation de la complexité de ce module, qui peut se traduire par un temps d'exécution plus long. En conclusion nous préférons conserver ce module dans l'état actuel et trouver une direction de travail plus rentable en terme d'amélioration.

L'image est ensuite segmentée en graphèmes, dans le but d'isoler les caractères du mot à reconnaître. L'algorithme mis en œuvre effectue cette tâche correctement pour plus de 70% des caractères (voir tableau IV), le reste des caractères étant majoritairement sur-segmentés. Ce phénomène est en grande partie pris en compte par notre modélisation. Nous avons pu constater l'adéquation du couple segmentation/modélisation associé au système, avec les données à traiter. Il est certain que la segmentation peut être améliorée, comme nous pouvons le constater en regardant les exemples présentés sur les figures 27 et 26. Le problème le plus important à régler serait les cas de sous-segmentation. Cependant une telle action se traduirait par l'ajustement ou l'ajout de certaines heuristiques, qui risque d'entraîner une augmentation du temps de traitement. Du point de vue de l'application visée, le temps accordé au traitement d'une enveloppe doit rester le plus court possible. Concernant la modélisation il est possible d'envisager la prise en compte de modèle bi-caractère pour les cas les plus fréquents de sous-segmentation comme `ll`, `tt`, `ou`,

or,.... La contrepartie de cette amélioration possible est l'augmentation du nombre de paramètres du système. De manière globale, nous avons constaté que le couple segmentation/modélisation est satisfaisant. De ce fait nous allons concentrer nos efforts sur une autre partie du système.

La première évaluation du système a permis de mettre en évidence l'importance du pouvoir discriminant individuel des primitives mises en œuvre. En effet nous avons montré que la longueur des échantillons à reconnaître a une influence significative sur les taux de reconnaissance. Plus ils sont longs, plus leur reconnaissance est facilitée. La raison est que dans les exemples longs, les probabilités de présence de points d'ancrage forts pour l'alignement sont plus élevées. En d'autres termes, la présence de primitives très discriminantes au sein d'un ensemble est importante. L'évaluation objective du pouvoir discriminant a permis de confirmer que certaines primitives utilisées ne sont pas très discriminantes. Cependant nous pouvons mentionner que d'autres semblent suffisamment discriminantes. Une possibilité d'amélioration est donc l'apport de primitives plus discriminantes dans le système. De plus sur la figure 31 nous pouvons constater que pour les exemples cursifs aucun des deux ensembles utilisés n'est réellement performant. Le développement de primitives mieux adaptées à la reconnaissance des exemples cursifs doit permettre l'amélioration des performances du système de reconnaissance. Nous proposons donc d'orienter nos travaux vers l'amélioration de la représentation de l'information présente dans l'image.

Un dernier point à signaler concernant l'étude des performances du système standard vient de l'analyse des graphiques de la figure 30. En effet nous pouvons constater que lorsque le système ne donne pas la bonne solution, l'intitulé exact est tout de même souvent classé dans les dix premières réponses. En effet la valeur TR(10) associée au Système Standard, pour un lexique 10 000, est de plus de 92%. Une amélioration possible est alors d'ajouter un module dit de post-traitement, effectuant une vérification de ces solutions à l'aide de données complémentaires afin de valider ou modifier la décision finale. Cette possibilité

d'amélioration d'un système de reconnaissance fait l'objet de nombreux travaux, comme le mentionne Koerich [91]. De plus ce dernier a développé un tel module pour le système de reconnaissance utilisé,conduisant à une amélioration significative des performances.

2.5.2 Orientation de nos travaux

Comme nous venons de le mentionner, nous proposons d'orienter notre projet de recherche vers l'amélioration de la représentation de l'information présente dans les images d'écriture. Cette phase passe par le développement de nouveaux ensembles de primitives, qui pourront être plus discriminants que ceux déjà utilisés. Une revue de la littérature associée montre que de nombreux travaux ont déjà été réalisés dans cette direction. Loncaric [109] et Trier [150] présentent un panorama des facteurs de forme, le second s'intéressant particulièrement au domaine de la reconnaissance d'écriture.

Une nouvelle représentation de l'information étant extraite, il faut l'intégrer dans le système de reconnaissance. Le système standard utilise deux sources d'information, caractérisées par des ensembles de primitives. Il les intègre en les prenant en compte conjointement. Il est possible d'utiliser cette technique pour intégrer une troisième source. Cependant cette technique conduit à une augmentation exponentielle du nombre de paramètres. Il faudrait augmenter le nombre d'échantillons proportionnellement afin de disposer d'une estimation fiable. Il est alors plus judicieux de recourir à une autre technique. Il est possible d'utiliser une technique de combinaison de classificateurs [2, 48, 86, 152, 156]. Elle consiste à construire autant de classificateurs que de sources d'information disponibles, puis de choisir la stratégie de combinaison de manière à rencontrer les objectifs visés. Par opposition à la technique précédente, chaque classificateur aura un nombre restreint de paramètres. Elle ne nécessitera donc pas une augmentation de la taille de la base de données. Cependant le choix de la technique de combinaison reste heuristique. Une autre approche consiste à définir un grand nombre de caractéristiques possibles puis d'effectuer la sélec-

tion des plus pertinentes pour la tâche désirée. Nous proposons d'utiliser une technique se rapprochant de cette catégorie.

Dans le but d'amélioration du système de reconnaissance du SRTP, nous proposons de développer de nouveaux espaces de représentation, permettant entre autres de mieux caractériser l'écriture cursive. Cependant, les ensembles de primitives utilisés actuellement permettent d'obtenir de bonnes performances. De ce fait il ne semble pas judicieux de les supprimer simplement. Dans cette optique, nous proposons de développer un algorithme permettant l'amélioration de la représentation de l'information utilisée par un système en remplaçant les primitives les moins discriminantes par de nouvelles. Ce dernier sélectionnera d'une certaine manière l'information la plus pertinente permettant d'augmenter les taux de reconnaissance du système à un niveau local.

2.6 Résumé

Ce chapitre a permis de présenter notre domaine d'intérêt : la reconnaissance de l'écriture manuscrite. Après avoir passé en revue un certain nombre de techniques utilisées pour la mise en œuvre d'un système de reconnaissance de l'écriture, nous avons présenté celui sujet de notre étude. Dans un premier temps les différents modules le composant ont été décrits : les pré-traitements, la segmentation, l'extraction de caractéristiques et la modélisation. Sa description s'est poursuivie par l'exposé de la mise en œuvre de son apprentissage et de l'évaluation de sa performance, ainsi que celui des données utilisées lors de ces étapes. Afin de mettre en évidence ses faiblesses, une analyse approfondie de ses performances a été effectuée. L'analyse des résultats obtenus au cours des différentes expériences réalisées nous a menés à conclure que la représentation de l'information fournie au système de reconnaissance est faible et donc qu'elle peut être améliorée.

Nous proposons donc d'orienter nos travaux de recherche dans cette direction, c'est-à-dire l'amélioration de la représentation de l'information au sein du système de reconnaissance. Pour cela nous allons développer de nouveaux espaces de représentation puis utiliser un

algorithme de sélection de caractéristiques permettant de choisir la représentation conduisant au meilleur taux de reconnaissance.

CHAPITRE 3

L'EXTRACTION DE CARACTÉRISTIQUES ADAPTÉES À L'ÉCRITURE MANUSCRITE

En reconnaissance de l'écriture manuscrite, les données à traiter sont des images. La mise en œuvre d'une étape de segmentation permet de la diviser en différentes imagettes de taille moins importantes. Cependant une imagette reste une matrice de pixels. Cette représentation de l'information n'est pas la plus adaptée pour la plupart des systèmes de reconnaissance. Une étape d'extraction de caractéristiques est réalisée de manière à extraire l'information la plus discriminante pour la tâche de reconnaissance et également pour réduire le volume d'informations qui sera fourni au système.

Dans la suite de ce rapport, les termes *caractéristique* et *primitive* seront employés souvent. Afin de supprimer toutes ambiguïtés, nous allons expliciter ces deux termes. Par caractéristique, nous entendons une *mesure quantitative ou qualitative* effectuée sur l'image. Il s'agit par exemple d'une distance, d'une configuration de concavité, de la taille d'un dépassement, ou encore de la présence d'une boucle dans une zone donnée. À partir de plusieurs caractéristiques, nous pouvons construire un *espace de représentation*, dans lequel chacune correspondra à un axe. Une donnée, dans notre cas précis une imagette, correspondra à un point unique dans cet espace. Notre système de reconnaissance utilisant une modélisation discrète, la représentation de l'information doit se faire sous la forme d'une *séquence de primitives*. Dans ce but, l'espace de représentation est discrétisé de manière à obtenir un nombre fini de partitions. Chaque partition est alors caractérisée par un symbole appelé *primitive*. Un *ensemble de primitives* est alors l'ensemble des primitives permettant de représenter symboliquement un espace de représentation.

Ce chapitre nous permettra d'introduire le domaine de l'extraction de caractéristiques. Dans un premier temps un exposé des techniques d'extraction d'information généralement

utilisées pour la reconnaissance de l'écriture manuscrite sera effectué. Nous poursuivrons en présentant les différents espaces de représentation que nous avons mis en œuvre pour l'amélioration de notre système. La section suivante permettra de présenter les expériences réalisées à l'aide de ces nouveaux espaces de représentation. Nous terminerons bien sûr par une conclusion et une discussion concernant de possibles améliorations.

3.1 Les caractéristiques extraites de l'écriture

Que ce soit pour les approches globales ou analytiques de reconnaissance, il existe un besoin quant à la réduction de la quantité d'informations présente dans l'image. En effet, pour la prise de décision, un système de reconnaissance n'a besoin que de l'information pertinente pour différencier un objet d'un autre. Dans ce but une étape d'extraction de caractéristiques est réalisée. C'est une phase critique lors de la construction d'un système de reconnaissance. L'une des raisons pour laquelle cette étape pose un problème est qu'une grande majorité des techniques d'extraction s'accompagne d'une perte d'information irrémédiable. De ce fait, il faut effectuer un compromis entre quantité et qualité de l'information.

Pour un problème de classification donné, la principale qualité recherchée pour un ensemble de caractéristiques est sa faculté de rassembler les objets appartenant à une même classe dans une même partition de l'espace de représentation, tout en éloignant autant que possible les autres. Cette qualité est communément appelée *pouvoir discriminant* de l'ensemble de caractéristiques

L'extraction de caractéristiques en reconnaissance de l'écriture est confrontée au grand problème de la variabilité intra-classe. En effet, d'un point de vue visuel, un caractère peut prendre différentes formes, en fonction de sa position dans le mot. Cependant, les plus grandes variations sont introduites par le scripteur. L'écriture étant propre à chaque individu, le tracé résultant de l'écriture d'un même mot par deux personnes peut être bien différent. De plus, pour un même scripteur, un certain nombre de contraintes influencent

la réalisation du tracé de son écriture. Nous pouvons citer entre autre l'outil, le support et même l'humeur de l'individu.

Dans la littérature, il existe un grand nombre de travaux concernant l'extraction de caractéristiques pour la reconnaissance de l'écriture. Afin de les présenter il est préférable de les regrouper en catégories. Nous allons suivre celle proposée par Gaillat et Berthod [49] et reprise par Heutte [66]. Les différentes techniques d'extraction sont classées en fonction de la notion physique que le concepteur a voulu mettre en évidence.

3.1.1 Comparaison globale

Les techniques de cette catégorie utilisent la caractéristique la plus simple de l'image, c'est-à-dire l'état ou l'intensité de chaque pixel. La notion mise en œuvre est la ressemblance visuelle de la forme traitée avec un certain nombre de modèles. Ce sont des méthodes utilisées classiquement en traitement d'images. Il n'y a pas réellement d'extraction, puisque l'ensemble des pixels de l'image sert à constituer le vecteur caractéristique. Cependant une étape de normalisation en taille des images est nécessaire, afin de pouvoir effectuer la comparaison avec les modèles. Cette dernière implique bien sûr une perte d'information. La prise de décision s'effectue alors en évaluant une mesure de similarité entre l'image traitée et les différents modèles. Un grand nombre de possibilités est présent dans la littérature quant à la mise en œuvre de cette mesure. Son choix, laissé au concepteur, ainsi que l'étape de normalisation influenceront fortement les performances des approche de cette catégorie.

De par leur nature, ces caractéristiques ne permettent pas de décrire les propriétés, que ce soient géométriques ou structurelles, des caractères. De ce fait leur utilisation se cantonne à la reconnaissance de caractères isolés et plus particulièrement à celle des chiffres. Dans certaines conditions elles peuvent donner de bons résultats comme le montre Gader *et al.* dans [45].

3.1.2 Transformations et développements en séries

Les techniques présentées de la section précédente sont basées sur l'apparence visuelle de la forme à reconnaître. Cette approche est logique puisque de premier abord l'homme n'utilise que cette information pour effectuer la reconnaissance d'objets. Cependant le système de perception visuel humain est bien plus complexe [61] et effectue un grand nombre de traitements lui permettant d'identifier et d'associer différentes caractéristiques. Il est donc intéressant pour le domaine de la reconnaissance de formes d'essayer d'extraire, à partir des images, des informations "non visibles". De telles techniques sont regroupées dans cette catégorie. Elles utilisent une transformation globale de manière à changer d'espace de représentation et ainsi faciliter l'extraction de caractéristiques pertinentes.

La plupart des techniques présentées dans cette catégorie ne sont pas propres à la reconnaissance de l'écriture. Au contraire, elles sont largement utilisées dans le domaine général de la reconnaissance de formes. La transformée de Fourier est certainement une des plus utilisée en reconnaissance de formes et de caractères [41, 112]. Les caractéristiques extraites sont en fait les descripteurs de Fourier basés sur les coefficients complexes des séries de Fourier. Elles sont invariantes aux rotations et aux changements d'échelle. Pour la reconnaissance de caractères, la transformée pourrait être calculée directement sur l'image du caractère, c'est-à-dire une application bidimensionnelle. Cependant on lui préfère généralement une transformation mono-dimensionnelle appliquée sur le contour du caractère. Afin de satisfaire la contrainte de périodicité du signal, le contour analysé doit être fermé. La propriété d'invariance aux rotations implique des problèmes de reconnaissance de certains caractères comme 6 et 9. Il faut donc rajouter d'autres caractéristiques au vecteur de manière à régler ce problème. Par exemple Mahmoud [112] utilise comme complément un encodage du contour des caractères.

Une autre transformation globale, assez proche de celle de Fourier, est celle en ondelette [113, 143]. Le principal intérêt des ondelettes est qu'elles permettent d'obtenir une

information fréquentielle localisée concernant un signal ou une fonction de base choisi. Ce type de données est particulièrement intéressant pour la classification. Malgré certains avantages cette technique est peu utilisée en reconnaissance de formes. La raison est que les caractéristiques extraites ne sont pas invariantes à la translation. En d'autres termes, un léger décalage du caractère donnera des coefficients d'ondelettes totalement différents. Chen et Bui [20] proposent un ensemble de caractéristiques invariantes à la rotation, à la translation et au changement d'échelle en couplant la transformation de Fourier et celle en ondelette. L'approche consiste à appliquer une première transformation qui permet de décrire la forme analysée en coordonnées polaire (r, θ), où l'origine est le centre de masse de la forme. Ensuite la transformée de Fourier est appliquée sur l'axe des angles θ, puis la transformée en ondelettes sur celui des rayons r.

Concernant les caractères, Andrews compare les transformations de Karhunen-Loéve (KL), Fourier, Hadamard (ou Walsh) et Haar dans [5]. Sa principale conclusion est que la transformation KL est trop gourmande en temps de calcul et qu'il est préférable d'utiliser les transformation de Fourier ou Hadamard. Avec la puissance actuelle des ordinateurs la transformée KL est de nouveau utilisée [99].

Une autre grande famille appartenant à cette catégorie est celle des moments invariants. L'invariance recherchée est celle à la rotation, à la translation et au changement d'échelle. Une revue de l'invariance dans le domaine de la reconnaissance de formes est présentée dans [155]. Les caractéristiques extraites par ces techniques sont considérées comme le résultat d'une transformation globale appliquée uniquement aux pixels de la forme analysée. De part leur nature, elles sont très peu sensibles aux variations locales de la forme, ce qui explique leur grande utilisation. Il existe plusieurs formulations des moments invariants, comme celles de Hu [67] et celle de Li [104]. Cependant les plus utilisées actuellement sont celles dérivées des polynômes complexes de Zernike [8, 79, 154]. La raison est que ces dernières ont des performances supérieures en termes d'invariance. Belkasim [12] effectue une revue de différents moments invariants et conclu sur la supériorité des moments

de Zernike normalisés. En reconnaissance de l'écriture, Kundu [98] utilise dans son système un ensemble de moments en collaboration avec d'autres caractéristiques. Heutte *et al.* [65] incluent dans leur vecteur caractéristique structurel et statistique, pour la reconnaissance de formes, les moments de Hu, de manière à prendre en compte la forme globale des caractères.

Les mots ou les caractères peuvent être considérés comme une combinaison de segments de différentes tailles, orientations et positions. Buse [17] propose d'utiliser les propriétés des filtres de Gabor afin d'extraire ces informations. La technique consiste à appliquer sur des images en niveaux de gris, une banque de n filtres de Gabor, répondant pour différentes directions. Les images résultantes sont ensuite binarisées permettant d'obtenir les segments présents dans l'image en fonction de leur orientation. Leur analyse permet d'obtenir un certain nombre de caractéristiques en évaluant par exemple leur surface, la position de leur centre de gravité,...

Dans [143], les auteurs proposent une approche basée sur les ondelettes et la compare avec d'autres techniques. Ils concluent sur la supériorité, en terme de pouvoir discriminant, des caractéristiques extraites par l'intermédiaire de leur technique, par rapport à celles obtenues à partir des moments de Zernike et de Li.

3.1.3 Concavités, convexités et occlusions

Les caractéristiques extraites de cette famille s'efforce de mettre en évidence les concavités, convexités et les occlusions présentes dans la forme analysée. Elles permettent de mettre en évidence les propriétés topologiques et géométriques de la forme. De part leur nature elles peuvent être utilisées pour caractériser une forme tant d'un point de vue local que global. Concernant la reconnaissance de caractères, une propriété intéressante de ces caractéristiques est leur faible sensibilité aux distorsions et donc aux variations de style. Une autre qualité, appréciée pour des applications industrielles, est que leur extraction

n'est pas coûteuse en termes de temps de calcul, en particulier par rapport à celles de la catégorie précédente.

Deux techniques sont principalement utilisées pour l'extraction de ces caractéristiques. La première utilise une image binaire et procède par étiquetage du fond de cette image, habituellement les pixels blancs. La technique consiste à attribuer à chacun de ces pixels une étiquette caractérisant sa vision de la concavité : entièrement fermée ou ouverte dans une ou plusieurs directions. Une seconde étape permet d'effectuer un comptage des différentes configurations rencontrées et ainsi d'obtenir un vecteur caractéristique. Cette technique est utilisée entre autres par Gader *et al.* [45], Impedovo *et al.* [73], Favata *et al.* [40] et Farouz [39].

La seconde technique est appliquée sur l'image du contour des formes à analyser. Un suivi du contour extérieur des caractères permet de détecter les parties concaves et convexes de la forme. La présence de contours intérieurs correspondra alors à celle d'occlusions dans la forme analysée. Cette technique est utilisé par Dzuba *et al.* afin de déterminer la présence d'arcs sur les contours intérieurs et extérieurs à proximité d'extremums [33]. D'autres systèmes utilisent également cette approche [16, 63]. En reconnaissance de l'écriture, la détection des boucles est souvent réalisée du fait de l'importance de ces caractéristiques de base. C'est généralement la technique présentée dans ce paragraphe qui est utilisée pour leur détection.

3.1.4 Allongements horizontaux et verticaux

Les caractéristiques de cette catégorie permettent de mettre en évidence les propriétés structurelles des caractères. En effet les directions des allongements de pixels d'un caractère permettent d'en exprimer la structure de son tracé. La technique d'obtention de ces caractéristiques consiste à effectuer une projection des pixels du caractère sur un axe perpendiculaire à la direction de recherche des allongements. La détection des maxima locaux sur l'histogramme résultant permet d'obtenir la position et la valeur des allonge-

ments. L'application de cette technique se fait généralement sur une image binaire des caractères. Un problème associé à cette approche est qu'elle induit la détection d'un grand nombre de maxima, ne correspondant pas forcément à des allongements. De ce fait ces caractéristiques sont principalement utilisées pour la reconnaissance de caractères imprimés multi-fontes et de caractères chinois. En effet la régularité associée aux différentes fontes de l'écriture imprimée permet d'obtenir les allongements réels.

Le succès de l'application de ces caractéristiques aux caractères chinois vient du fait que ce type d'écriture est principalement composées de traits horizontaux verticaux et obliques. Les allongements pour cette direction sont alors obtenus en effectuant des projections obliques à $\pm 45°$.

Chim *et al.* utilise ces caractéristiques, complétées par une détection de segments de lignes, pour son système de reconnaissance de chiffres [24]. Une approche un peu différente est proposée par Park [128]. Il propose d'effectuer différentes projections, non pas uniquement sur un axe mais également sur un point, le centre de l'image en l'occurrence. Heutte *et al.* utilise également ce type de caractéristiques dans leur vecteur caractéristique structurel et statistique pour la reconnaissance de caractères manuscrits [65]. Il propose une technique basée sur l'analyse de l'histogramme cumulée des projections horizontales et verticales de manière à détecter correctement les maxima.

3.1.5 Particularités locales

Cette famille de caractéristiques, comme la précédente, permet de mettre en évidence la structure des caractères. Les particularités locales détectées sont les fins de trait, les jonctions en T ou Y, ainsi que les intersections (X). De telles informations permettent de mettre en œuvre une description précise de la structure des caractères. La principale technique permettant d'obtenir ces caractéristiques consiste à appliquer une série de masques, un par particularité recherchée, sur l'image. Pour l'écriture imprimée, ces derniers sont appliqués directement sur l'image des caractères et à des endroits particuliers, choisis de manière

à favoriser la discrimination entre les différents caractères. Concernant l'écriture manuscrite, une étape de squelettisation du tracé est réalisée au préalable. De plus, l'écriture manuscrite étant bien plus variable que l'imprimé, l'application de masques ne se fera pas uniquement à un endroit précis de l'image mais plutôt sur une zone, voire l'image entière.

Leur faculté de décrire la structure des caractères rendent ces techniques attractives pour la reconnaissance d'écriture. Elles ne sont généralement pas utilisées seules. Han [63] les utilise en association avec des primitives géométriques. Il propose d'effectuer une décomposition du squelette et de détecter les particularités suivantes : fins de trait, jonctions, croisements, boucles et simples courbes. Les particularités locales sont également utilisées en collaboration avec d'autres caractéristiques dans les systèmes de reconnaissance de l'écriture décrit dans [65, 98, 140].

3.1.6 Intersections avec des droites

Les propriétés projectives d'un caractère peuvent être mises en évidence en effectuant le comptage d'intersections entre une ou plusieurs droites et le caractère. Différents paramètres sont alors ajustables : le nombre de droites, leurs orientations et la longueur des segments de droite. En effet une droite couvrant toute la surface du caractère peut être utilisée. Cependant la recherche de particularités conduit à positionner seulement des segments de droite à des endroits particuliers choisis par le concepteur. En plus du nombre d'intersections, leur localisation est également informante.

Le système de lecture de chèques présenté par Knerr *et al.* utilisent ces caractéristiques [88]. Pour cela le comptage du nombre d'intersections entre le caractère et trois droites horizontales et trois verticales est réalisé. Heutte *et al.* n'utilisent pour leur part que deux droites horizontales et une verticale pour la reconnaissance de caractères [65].

Dans cette catégorie, nous pouvons classer une autre méthode basée sur les intersections. Il s'agit des lieux caractéristiques ou "characteristic Loci" [89] qui consiste à étiqueter

chaque pixel du fond de l'image, en fonction du nombre d'intersections entre les quatre demi-droites horizontales et verticales issues de ce point et le caractère. Dans l'application présentée la valeur de chaque compteur ne peut être que 0, 1 ou 2, correspondant respectivement à aucune, une et plus de deux intersections. Cette contrainte n'est cependant pas obligatoire.

3.1.7 Mesures physiques ou géométriques

Ce type de caractéristiques est généralement une mesure de distance entre les points de l'image, appartenant ou non à l'écriture. De telles mesures permettent de prendre en compte les propriétés métriques des caractères. L'estimation de la distance entre les pixels du caractère les plus éloignés suivant les axes horizontal ou vertical par exemple, permet d'obtenir la largeur et la hauteur totale du caractère. Ces données peuvent ensuite être combinées de manière à obtenir le rapport hauteur/largeur par exemple. Le calcul de distance peut également être effectué à des endroits précis de l'image des caractères, de manière à mettre en évidence une particularité propre à l'un ou plusieurs. Une autre possibilité consiste à mesurer la distance entre le bord de l'image et le premier pixel appartenant au caractère. Cette technique permet de prendre en compte le profil du caractère. Elle est couramment utilisée en reconnaissance de caractères.

La prise en compte de distances nécessite obligatoirement une normalisation, soit de la distance elle même et dans ce cas après avoir effectué la mesure, soit de la taille de l'image avant l'évaluation de la distance.

Kundu *et al.* [98] utilisent dans leur système de reconnaissance de mots le rapport hauteur/largeur dans leur vecteur. Kim utilise également cette caractéristique [81]. Oh dans [122] propose deux techniques d'extraction de caractéristiques basées sur des mesures de distances. La première "*Distance Transformation* (DT) *Feature*" consiste à estimer pour chaque pixel du fond de l'image, la distance le séparant du plus proche pixel appartenant au caractère. Elle est appliquée sur une image normalisée de taille 16×16, les 256 va-

leurs résultantes sont alors présentées directement à un réseau de neurones. La seconde approche nommée "*Directionnal Distance Distribution* (DDD) *Feature*" est basée sur le même concept. Cependant les mesures de distance sont réalisées en fonction des huit directions et pour les pixels blancs et noirs de l'image. Chaque pixel est alors caractérisé par huit mesures de distance. Afin de réduire le nombre de dimensions de son vecteur, l'auteur propose de calculer la moyenne des distances pour des cellules de 4×4 pixels. Le vecteur final contient 256 composantes. L'auteur compare alors les performances en reconnaissance de ces caractéristiques à d'autres et conclu sur la supériorité des primitives DDD.

3.1.8 Mesures de densité

La détection de traits caractéristiques peut également être effectuée en analysant la densité de pixels appartenant au caractère, dans l'image entière ou dans certaines parties uniquement. L'application la plus courante de cette technique consiste à effectuer un découpage de l'image en un nombre de zones et d'effectuer l'estimation de la densité de pixels dans chacune d'elles. Cette technique de découpage est connue sous le terme anglais de "*zoning*". Elle est basée sur des concepts de la perception humaine. Une étude de l'utilisation de cette technique pour la reconnaissance de caractères manuscrits a été effectuée par Suen *et al.* dans [148]. Une difficulté liée à cette approche est le choix du nombre de zones utilisées pour le *zoning*. En effet l'utilisation d'un grand nombre de zones permet de mieux représenter les caractères. Cependant cette option va s'accompagner d'une augmentation du temps de calcul associé à l'extraction et à la reconnaissance.

Ce type de caractéristiques est rarement utilisée seule dans un vecteur caractéristique. Elles viennent plus souvent compléter une représentation, comme par exemple dans le système présenté par Oliveira *et al.* dans [123]. Dans cette approche la technique de *zoning* est utilisée. Pour chaque zone, l'information de densité vient compléter un codage des concavités et du contour.

Comme nous venons de le voir, la technique de *zoning* n'est pas seulement associée aux mesures de densité. Son utilisation permet d'une certaine manière de localiser l'extraction de caractéristiques. Son utilisation permet donc d'incorporer un contexte local directement lors de l'extraction de caractéristiques. Cette faculté est une raison de la grande utilisation de cette technique. Kimura [83] l'utilise sur le contour des caractères. Le nombre de segments du contour est alors compté dans chaque zone et ce pour différentes orientations : 0°, 45°, 90° ou 135°. Le vecteur caractéristique contient alors quatre composantes par zone.

3.1.9 Conclusion

Nous venons de présenter un survol des techniques d'extraction de caractéristiques pour la reconnaissance de caractères. Lors de la conception d'un système de reconnaissance, cette étape est importante car la représentation de l'information résultante conditionnera tout le reste du processus. De ce fait, la conception de ce module reste une tâche difficile, se traduisant généralement par un compromis entre la qualité de la représentation de l'information et la quantité d'information, qui va conditionner le temps de calcul. Il existe bien sûr certaines contraintes liées à l'application visée. Nous pouvons par exemple penser à une contrainte de temps réel pour des systèmes industriels, qui impose un temps d'extraction maximum.

Comme nous l'avons vu dans les différents exemples cités lors de cette revue de littérature, les composantes du vecteur caractéristique sont obtenues à partir de plusieurs techniques d'extraction. En effet la complémentarité des différentes représentations possibles est exploitée de manière à augmenter le pouvoir discriminant du vecteur caractéristique.

Une dernière remarque concernant l'extraction de caractéristiques est liée à une spécificité de notre domaine d'application. En effet l'écriture est sujette à un grand nombre de variations, qui se traduit principalement par une grande dispersion intra-classe. De ce fait, il peut être judicieux d'extraire des caractéristiques peu sensibles aux variations locales. D'un autre coté, certains caractères peuvent être différenciés seulement en analysant une

particularité locale comme par exemple les lettres cursives a et o. Cette remarque conforte la thèse qu'un vecteur caractéristique composite doit permettre de prendre en compte la forme globale comme les spécificités locales des caractères.

3.2 Les espaces de représentation développés

Dans cette section nous allons présenter les différents espaces de représentation que nous avons mis en œuvre. Leur choix a été guidé par notre but, à savoir l'amélioration de la représentation de l'information utilisée par notre système de reconnaissance. Du fait de l'application industrielle visée : le tri automatique du courrier, le temps d'extraction de ces caractéristiques ne doit pas être trop long.

Comme nous l'avons mentionné lors de l'analyse des performances du système standard, l'écriture cursive est la plus difficile à traiter. Cette remarque a guidé nos choix concernant les différents espaces de représentation développés. En effet une amélioration de la représentation de l'écriture cursive doit directement impliquer une augmentation des performances du système de reconnaissance.

Il est pertinent de mentionner à ce niveau que l'extraction de caractéristiques doit être pensée au niveau des graphèmes et non pas uniquement au niveau des caractères entiers. En effet, notre algorithme de segmentation a été conçu de manière à favoriser la sur-segmentation des caractères. Environ 27% des caractères de notre base de données sont composés de deux graphèmes ou plus (voir tableau IV).

3.2.1 L'espace de représentation des concavités (CCV)

3.2.1.1 Justifications

Le premier espace de représentation que nous avons développé est basé sur l'analyse des concavités de la forme des graphèmes. Ce choix a été guidé par une constatation : environ 52% des graphèmes sont caractérisés par la primitive perceptuelle "-" (voir tableau V).

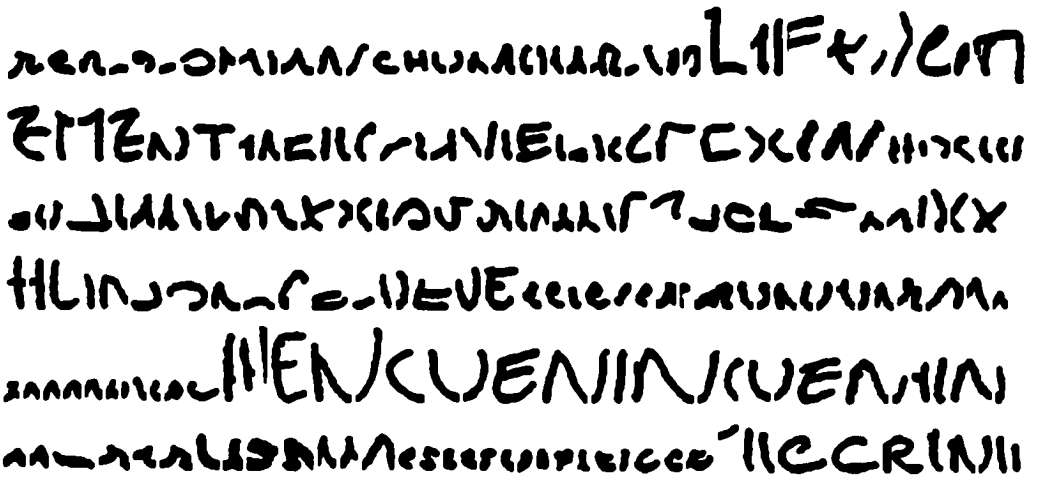


FIGURE 32 Une série d'exemples de graphèmes caractérisés par la primitive perceptuelle "-".

Cette dernière encode l'absence des caractéristiques de base que sont les dépassements hauts, bas et les boucles. Nous avons effectué une analyse visuelle des graphèmes caractérisés par cette primitive. Un échantillon de ces graphèmes est présenté sur la figure 32. Elle permet de constater la grande disparité des formes caractérisées par cette primitive.

L'ensemble perceptuel a été construit afin de caractériser plus particulièrement l'écriture cursive. La primitive "-" devrait donc servir à représenter les graphèmes provenant de caractères ne contenant ni boucles, ni dépassements, c'est-à-dire c, i, m, n, r, s, u, v, w et x. Nous pouvons constater la présence de tels graphèmes sur la figure 32. Cependant ils peuvent également provenir des caractères a, e et o, lorsque les boucles ne sont pas fermées ou inexistantes, ainsi que de caractères contenant des dépassements lorsque ceux-ci sont segmentés en plusieurs graphèmes (comme par exemple la partie gauche d'un y ou la partie droite d'un b). De plus, une grande partie des graphèmes provenant de l'écriture bâton est caractérisée par cette primitive du fait que les dépassements sont inexistants pour ce style d'écriture.

La prise en compte des dépassements et des boucles dans les graphèmes réalisée par l'intermédiaire des primitives perceptuelles permet d'intégrer une information concernant la structure du graphème. Cette dernière est pertinente pour la reconnaissance. Cependant elle ne caractérise pas l'aspect de la forme. L'analyse des concavités permettra d'apporter une telle information. De par la nature de l'information extraite, cette technique devrait permettre de différencier les parties gauche et droite des lettres comme n et u. Les boucles mal fermées auront une signature facilement identifiable, ce qui devrait faciliter leur prise en compte. De par leur aspect différent (voir figure 32), les graphèmes de l'écriture bâton produiront également des vecteurs caractéristiques bien particuliers. Un des points forts de cette approche est sa faible sensibilité aux variations locales de la forme, propriété intéressante pour la reconnaissance de l'écriture manuscrite.

3.2.1.2 Le vecteur caractéristique CCV

Nous avons choisi de mettre en œuvre l'extraction de concavités par la technique d'étiquetage de pixels. Cette approche utilise l'image binaire de la forme à analyser. Elle consiste à attribuer une étiquette à chaque pixel du fond de l'image, les blancs en l'occurrence, en fonction de sa vision des concavités. La technique consiste à effectuer un lancer de rayon dans les huit directions pour chaque pixel blanc de l'image et de récupérer le point de terminaison de ces huit sondes. Deux alternatives sont possibles :

1. la sonde se termine sur un pixel de l'écriture,
2. la sonde se termine sur le cadre de la zone d'analyse.

Plus le nombre de sondes se terminant sur l'écriture est important, plus la concavité est fermée. Nous présentons sur la figure 33 plusieurs exemples de configurations de concavités rencontrées. Il en existe $2^8 = 256$ distinctes. La construction du vecteur caractéristique consiste alors à compter le nombre de pixels étiqueté par chacune de ces configurations, puis à normaliser les valeurs de manière à obtenir une distribution. Cette étape garantie l'homogénéité du vecteur, indépendamment de la taille de l'image analysée.

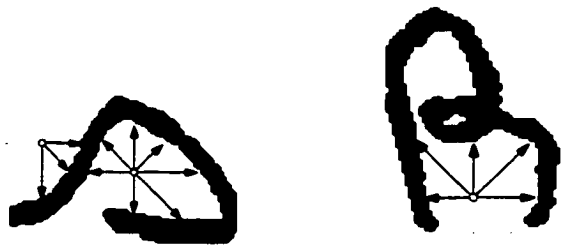


FIGURE 33 Illustration de l'extraction des concavités.

(a) Concavité fermée (b) Sortie Sud-Est

FIGURE 34 Illustration de l'extraction des concavités – Recherche d'une sortie lorsqu'une concavité fermée est rencontrée.

Dans [66] l'auteur propose de rajouter quatre configurations lorsqu'une concavité fermée est détectée, c'est-à-dire lorsque les huit sondes ont atteint un pixel d'écriture. Dans ce cas on cherche à sortir de cette concavité en utilisant deux directions successives, comme par exemple Nord puis Est. Ce concept est illustré sur la figure 34. Ces configurations supplémentaires permettront de mieux caractériser les concavités fermées et ainsi de supprimer certaines confusions.

Le vecteur caractéristique associé à cette technique d'extraction peut contenir $2^8+4 = 260$ composantes. Cependant il est certain que certaines configurations ne seront jamais ou très peu rencontrées, en particulier celles discontinues. Ce sont les configurations pour lesquelles le nombre de sondes se terminant sur l'écriture peut être divisé en deux regroupements continus. Le fait de ne pas les prendre en compte permet de réduire considérablement le nombre de dimensions de l'espace de représentation.

3.2.2 L'espace de représentation des distributions de distances directionnelles (DDD)

3.2.2.1 Justifications

Notre but étant d'améliorer la représentation de l'information extraite de l'écriture, il est judicieux d'utiliser des caractéristiques provenant de différentes familles. Cette approche permettra d'obtenir une certaine complémentarité entre les différents espace de représentation mis en œuvre. Dans cet ordre d'idées, nous avons décidé de mettre en évidence les propriétés physiques des formes par l'intermédiaire d'un espace basé sur des mesures de distances. Comme nous l'avons déjà mentionné, notre réflexion concernant le choix de nouveaux espaces de représentation est principalement orientée par des constatations effectuées sur l'écriture cursive. Une analyse des formes présentées sur figure 32 permet de constater que la prise en compte de distributions de distances doit faciliter la discrimination entre les formes. En effet, les distributions de distances horizontales entre le graphème d'un c et celui de la partie gauche d'un u seront différentes et donc faciliteront leur discrimination.

Nous avons décidé de mettre en œuvre la technique proposée par Oh [122], pour la richesse de la représentation fournie. En effet, elle permet de prendre en compte les distributions de distances dans les huit directions de Freeman que ce soit pour les pixels blancs ou les noirs. Cette technique permet de prendre en compte d'une certaine manière huit profils différents de la forme analysée.

Oh a développé cette extraction de caractéristiques pour la reconnaissance de caractères isolés et pour une utilisation avec des réseaux de neurones. Les résultats obtenus par cette technique sont supérieurs à d'autres approches dans le contexte expérimental proposé par l'auteur. L'application de cette technique sur des graphèmes qui ont une forme plus simple que des caractères entiers devrait permettre d'obtenir de bons résultats.

3.2.2.2 Le vecteur caractéristique DDD

L'extraction d'information par cette technique est réalisée sur une image binaire. Contrairement à l'approche utilisée pour l'extraction de concavités, tous les pixels, blancs ou noirs, sont analysés. Pour chaque pixel, la distance qui le sépare du prochain pixel de couleur opposée est mesurée et ce dans les huit directions de Freeman (identique aux huit sondes utilisées pour l'extraction de concavités). La métrique utilisée est la distance de Manhattan, c'est-à-dire que la distance entre un pixel est ses huit voisins est de 1. De plus, Oh propose d'utiliser une technique de "*tiling*" (voir [122]) afin d'augmenter le pouvoir discriminant de ces primitives. Elle consiste à apposer tout autour de l'image analysée, une copie de cette dernière de manière à ce que les sondes envoyées à partir d'un point se prolongent dans la même image. L'utilisation de différents angles de rotation appliqués aux copies de l'image conduisent à différentes possibilités pour le "*tiling*". Une fois l'ensemble des distances mesurées, la zone d'analyse est divisée en un certain nombre de cellules de taille identique. L'auteur divise la hauteur et la largeur de la zone par 4 obtenant ainsi 16 cellules. La moyenne des distances est alors calculée pour chaque zone, pour les huit directions et pour les pixels blancs et noirs séparément. Chaque zone est alors caractérisée par 16 valeurs réelles et le vecteur caractéristique résultant contient donc $16 \times 16 = 256$ valeurs. L'auteur normalise alors ces valeurs entre 0,0 et 1,0 afin de satisfaire aux contraintes d'entrée de son système. Une représentation graphique de cette extraction est présentée à la figure 35.

L'application de cette technique que nous désirons mettre en œuvre est différente de celle proposée par Oh. En effet, nos données sont des graphèmes et non pas des caractères entiers. L'étape de "*tiling*" que propose l'auteur, permettant de mieux caractériser la forme, n'a alors pas le même intérêt. Cette étape n'a donc pas été mise en œuvre. De ce fait les mesures de distance se terminent sur le bord de la zone d'analyse.

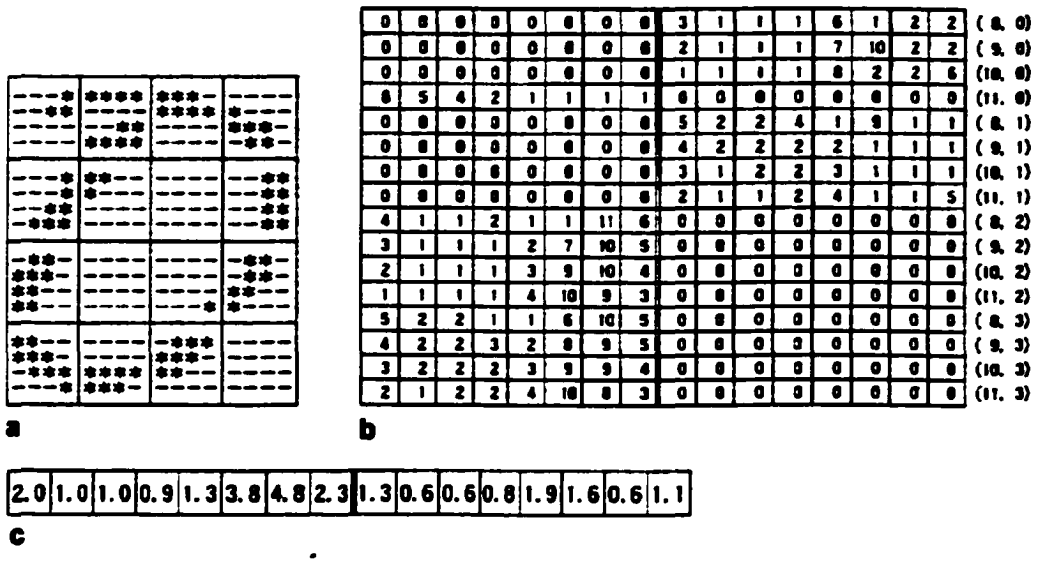


FIGURE 35 Extraction des caractéristiques DDD d'après [122] – **a** Image normalisée et divisée en 16 zones. – **b** Codage des 16 distances de l'ensemble des pixels (Blancs/Noirs) de la troisième zone. – **c** Vecteur caractéristique associé à la troisième zone.

Une seconde différence entre notre mise en œuvre et celle proposée par Oh se situe au niveau de l'image de départ. En effet dans [122] une normalisation en taille est effectuée au début du processus, suivie par une nouvelle binarisation. Cette étape se traduit obligatoirement par une perte d'information, qui peut être acceptable lorsqu'elle est appliquée sur des caractères entiers. En effet ces derniers garderont tout de même leurs caractéristiques. Par contre la perte résultante de l'application sur des graphèmes, de formes beaucoup plus variables, risque d'être trop grande. Cette étape n'a donc pas été réalisée. Par contre une normalisation est tout de même nécessaire afin de conserver l'homogénéité des représentations entre des graphèmes de tailles différentes. Pour cela nous avons normalisé les distances en fonction de la taille des graphèmes. Les mesures suivant l'axe horizontal ont été divisées par la largeur du graphème et celles suivant l'axe vertical par la hauteur. Quant à celles effectuées suivant les axes à $\pm 45°$, elles ont été divisées par la valeur minimum

choisie entre la hauteur et la largeur du graphème. Le choix de cette valeur s'explique par la métrique utilisée : la distance de Manhattan.

Afin de réduire la dimension de l'espace de représentation, la zone d'analyse n'a été divisée qu'en quatre zones au lieu de 16. Le vecteur caractéristique ne contient alors que $16 \times 4 = 64$ composantes. Pour une raison de compatibilité avec son système, Oh effectue une normalisation des valeurs du vecteur caractéristique à la fin du processus. N'ayant pas les mêmes contraintes, cette étape n'est pas réalisée.

3.2.3 L'espace de représentation des histogrammes de direction (HD)

3.2.3.1 Justifications

Afin de compléter la représentation de l'information contenue dans les graphèmes, il nous a paru judicieux d'effectuer une analyse du contour des formes. En effet ce dernier est très informant et représente directement l'aspect de la forme. Des caractéristiques extraites à partir du contour seront de nature différente de celles basées sur les concavités ou les distributions de distances directionnelles, qui sont elles extraites à partir de regroupements de pixels. Cela garantit une certaine complémentarité des représentations.

Il y a plusieurs possibilités pour l'encodage du contour d'une forme. Une approche utilisée est celle des descripteurs de Fourier. Comme nous l'avons mentionné dans la section 3.1.2 leur propriété d'invariance conduit à leur adjoindre d'autres caractéristiques afin de pouvoir différentier certains caractères.

Nous avons opté pour une autre approche, celle proposée par Kimura [83]. Elle est basée sur une estimation de l'histogramme des directions effectuée lors d'un suivi de contour. Sa mise en œuvre est simple et peu coûteuse en temps de calcul, particulièrement comparée à l'extraction des descripteurs de Fourier. Cette remarque est la principale raison du choix de cette approche.

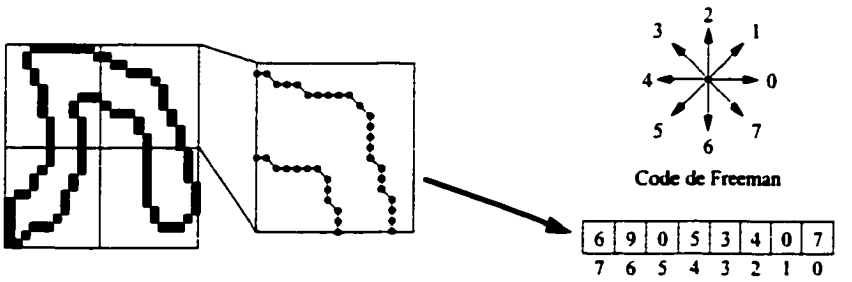


FIGURE 36 Illustration de l'extraction des histogrammes de direction sur le contour d'un graphème.

3.2.3.2 Le vecteur caractéristique HD

Nous avons décidé de mettre en œuvre la technique des histogrammes de directions afin de coder les contours des graphèmes. L'extraction de ces caractéristiques est donc réalisée sur l'image des contours. Cette dernière est disponible dans notre système après la segmentation de l'écriture en graphèmes. La première étape de l'extraction consiste à découper l'image en un certain nombre de zones (technique du "*zoning*"). Un suivi de contour est alors réalisé de manière à obtenir les directions entre pixels successifs. Ces dernières sont codées par l'intermédiaire du code de Freeman (voir figure 36). Les occurrences de ces directions sont alors dénombrées pour chaque zone individuellement. Nous présentons sur figure 36 un exemple de cette technique d'extraction. De même que pour les autres caractéristiques mises en œuvre, une étape de normalisation est nécessaire de manière à pouvoir comparer les vecteurs de graphèmes de taille différente. Pour ces caractéristiques, la normalisation s'effectue en divisant chaque compteur de direction d'une zone par le nombre de pixels de contours présents dans cette zone. Le vecteur caractéristique associé à cet espace de représentation contient alors huit composantes par zone.

Le seul paramètre associé à cette technique d'extraction est le nombre de zones du découpage de l'image du graphème. Afin de conserver une certaine cohérence avec les primitives DDD, nous avons choisi d'utiliser quatre zones. Ce nombre de zones peut sembler faible, cependant nous désirons utiliser la technique de *zoning* en amont de cette phase d'extrac-

tion. En particulier nous pensons utiliser le découpage en zones d'écriture de manière à effectuer l'analyse indépendamment pour la zone des dépassements hauts, la zone médiane et la zone des dépassements bas. Cela signifie que lors des différentes expériences que nous allons réaliser le nombre de zones global sera plus important.

3.3 La création des ensembles de primitives

Comme nous l'avons vu dans le second chapitre de cette thèse, le processus de reconnaissance s'effectue en plusieurs étapes successives. Notre système utilisant une modélisation markovienne discrète, l'image de l'écriture doit être représentée par une suite de symboles ou primitives. Ces dernières doivent être obtenues à partir de l'espace de représentation évalué. Nous allons présenter dans cette section les étapes permettant la construction d'un ensemble de primitives à partir d'un espace de représentation donné.

3.3.1 La quantification vectorielle

Les différents espaces de représentation que nous avons décidé de mettre en œuvre sont caractérisés par un vecteur à composantes réelles. Une étape de quantification vectorielle est donc nécessaire afin de diviser cet espace en classes ou régions. L'algorithme LBG décrit dans la section 1.3.3 a été développé dans ce but. Son choix a été guidé par sa simplicité d'implémentation, sa rapidité d'exécution et le fait qu'il ne requiert pas d'initialisation. En effet une telle étape est souvent nécessaire, comme pour l'algorithme des *k-means* décrit dans la section 1.3.2. Elle consiste à choisir aléatoirement un nombre d'échantillons de la base d'apprentissage équivalent au nombre de régions désirées. Une telle étape a forcément une influence non-négligeable sur le résultat final de la quantification vectorielle. Dans le cas de l'algorithme LBG, le centre de gravité de l'ensemble des échantillons est utilisé comme point de départ du processus.

La principale limitation de l'algorithme LBG est le nombre final de régions qui doit être une puissance de 2. Pour un problème donné, il est parfois possible de connaître le nombre

idéal de régions. Ce n'est pas le cas pour notre application. La seule information dont nous disposons est le nombre de classes mises en œuvre par notre modélisation : 209. Idéalement nous pouvons considérer qu'une classe doit être caractérisée par une primitive. Dans la pratique il est certain qu'il n'est pas possible de caractériser l'ensemble des formes associées à une classe par une seule primitive, cela en raison des différentes sources de variation propres à l'écriture. De plus il est possible que parmi les 209 classes certaines soient liées. En effet il est probable que les formes associées à la lettre i soient très proches de celles associées au premier graphème de la lettre u. Ces remarques confirment le fait que nous ne pouvons pas connaître *a priori* le nombre de régions idéal.

Du point de vue de la mise en œuvre de l'algorithme LBG (voir section 1.3.3) certains paramètres doivent être choisis. Le premier est la métrique employée pour la classification des échantillons. La distance euclidienne a été utilisée à cet effet. Le second est le seuil de distorsion permettant de d'arrêter l'étape de classification. Il a été fixé expérimentalement à 10^4 de manière à obtenir un compromis précision/rapidité satisfaisant. Une étape particulière à l'algorithme LBG est celle de la division des régions. Au début du processus il n'y a qu'une seule région caractérisée par son centre de gravité. Nous estimons à cet instant l'écart type associé à chaque composante du vecteur caractéristique. Ces valeurs sont ensuite utilisées lors de la division. Cette étape consiste à remplacer chaque centre de gravité associé à une région par deux nouveaux, obtenus en ajoutant et retranchant les valeurs d'écart type à chaque composante. Afin de prendre en compte l'augmentation du nombre de régions, les valeurs du vecteur d'écart type sont divisées par deux à chaque itération, après le calcul des nouveaux centres. Cette technique n'est certainement pas optimale. Cependant expérimentalement elle est satisfaisante. De plus elle ne nécessite pas de calculs particuliers, comme la recherche des bornes de la régions et donc favorise la rapidité de la quantification vectorielle.

Lors de l'évaluation d'un espace de représentation, sa division est effectué jusqu'à l'obtention de 1 024 centres. Durant les étapes intermédiaires de la quantification vectorielle,

nous mémorisons les centres de gravité obtenus. En associant un symbole par centre, nous pouvons construire un ensemble de primitives à chaque étape. Afin d'évaluer un espace de représentation, nous construisons cinq ensembles de primitives contenant respectivement : 64, 128, 256, 512 et 1 024 symboles.

3.3.2 Augmentation du pouvoir discriminant des primitives

Comme nous l'avons déjà mentionné, la qualité de la représentation de l'information, c'est-à-dire l'ensemble de primitives à ce moment, influencera directement les performances du système de reconnaissance. Cette qualité peut être quantifiée par l'intermédiaire du pouvoir discriminant des primitives (voir section 1.2.4). Il permet de mesurer la faculté qu'une primitive a de différencier les formes parmi les classes de la modélisation mises en œuvre. Il est donc intéressant d'intégrer, lors de la conception d'un ensemble de primitives, cette information concernant les classes. Ceci est possible en utilisant l'algorithme LDA présenté à la section 1.4. Ce dernier permet d'obtenir une transformation linéaire de l'espace de représentation des données garantissant une séparation des échantillons en fonction d'une variable qualitative, la classe de l'échantillon en l'occurrence.

Afin de mettre en œuvre l'algorithme LDA, nous devons disposer de l'étiquette "classe" pour chaque échantillon. Cette information n'est pas directement disponible dans notre base de données. Nous proposons alors d'exploiter une possibilité offerte par la modélisation markovienne pour extraire cette information. En effet l'étape de *Backtracking* de l'algorithme de Viterbi (voir section 1.1.3.2) permet d'obtenir l'alignement entre les graphèmes et les classes de la modélisation conduisant à la plus forte probabilité d'observation. Cette étape peut alors être utilisée de manière à étiqueter automatiquement les graphèmes. Cependant un système de reconnaissance fonctionnel est nécessaire pour effectuer cette tâche. Pour obtenir un ensemble de N_F primitives à partir d'un espace de représentation, nous proposons la stratégie suivante :

1. **Quantification vectorielle :** effectuer une quantification vectorielle afin d'obtenir le nombre de régions désirées N_F.
2. **Apprentissage (1ère phase) :** à partir de l'ensemble de N_F primitives obtenu, construire un système de reconnaissance.
3. **Étiquetage des graphèmes :** utiliser l'étape de *Backtracking* de l'algorithme de Viterbi pour étiqueter les graphèmes.
4. **Calcul des paramètres LDA :** appliquer l'algorithme LDA sur l'ensemble des échantillons de la base d'apprentissage afin d'obtenir la matrice de transformation W.
5. **Projection des échantillons :** utiliser la matrice W afin de projeter l'ensemble des échantillons d'apprentissage dans le nouvel espace.
6. **Quantification vectorielle :** effectuer une quantification vectorielle dans ce nouvel espace afin d'obtenir le nombre de régions désirées N_F.
7. **Apprentissage (2ème phase) :** à partir du nouvel ensemble de N_F primitives, construire le système de reconnaissance final.

Dans cette application de l'algorithme LDA, un premier système de reconnaissance est utilisé afin d'obtenir la classe associée à chaque échantillon d'apprentissage. Cette technique ne garantit pas que l'étiquette attribuée à chaque graphème est exacte. Cependant c'est une alternative acceptable à l'étiquetage manuel de toute la base, qui serait le seul moyen de l'obtenir. Il est à noter qu'il n'y a pas de possibilité pour vérifier la validité de l'étiquette. Les taux de reconnaissance obtenus par le premier système de reconnaissance ne peuvent pas être utilisés dans ce but. En effet la reconnaissance s'effectue globalement sur le nom de ville et non pas au niveau caractère. De ce fait, il est possible que pour un exemple donné, le système utilise l'alignement graphème/classe correct, mais que la probabilité au niveau nom de ville soit plus faible que celle associée à une autre entrée du lexique. Nous assumons que dans la grande majorité des cas le premier système retourne l'étiquette cor-

recte. Une vérification manuelle, effectuée sur un nombre limité d'échantillons, a permis de valider cette hypothèse.

3.3.3 Prise en compte des zones d'écriture

Dans le but d'améliorer la qualité des ensembles de primitives, nous proposons d'utiliser la technique de *zoning*. Cette dernière consiste à découper l'image en différentes zones et d'extraire pour chacune d'elles un vecteur caractéristique. La représentation de l'information résultante est plus précise et plus riche. De plus cette technique permet d'intégrer directement lors de l'extraction une certaine information contextuelle locale. Cette dernière doit permettre d'obtenir des ensembles de primitives plus discriminants.

Nous proposons d'effectuer le découpage des graphèmes en fonction des zones d'écriture définies à la fin des pré-traitements : zone des dépassements hauts, zone médiane et zone des dépassements bas. Cette approche permet de prendre en compte l'information contextuelle associée à la présence de dépassements.

Les différentes zones d'écriture sont obtenues à l'aide d'heuristiques, au cours de la phase de pré-traitements de l'image de l'écriture. Leur localisation n'est pas parfaite, en partie à cause de la grande variabilité de l'écriture. Une autre cause est la non-optimalité de l'algorithme utilisé pour leur détection. Idéalement, pour des exemples de type bâton, seule la zone médiane devrait être présente. Dans la pratique il en est différemment et la présence des zones de dépassement est souvent remarquée pour ce type d'exemples.

Sur la figure 37, nous présentons un échantillon bâton ainsi que la hauteur des trois zones d'écriture. Les indications h_{Sup}, h_{Med} et h_{Inf} sont relatives respectivement à la zone supérieure (ou zone des dépassements hauts), la zone médiane et la zone inférieure (ou zone des dépassements bas).

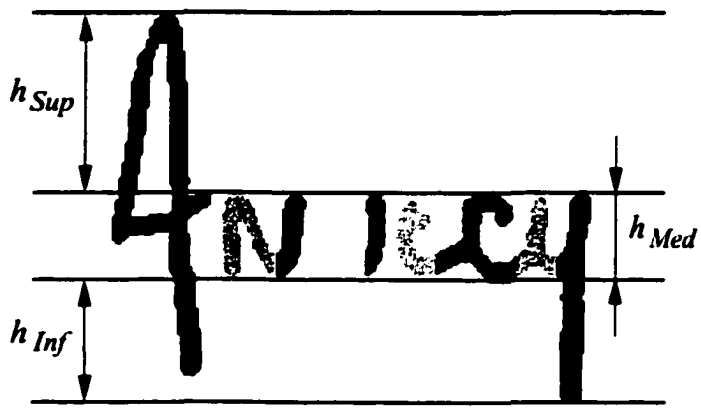
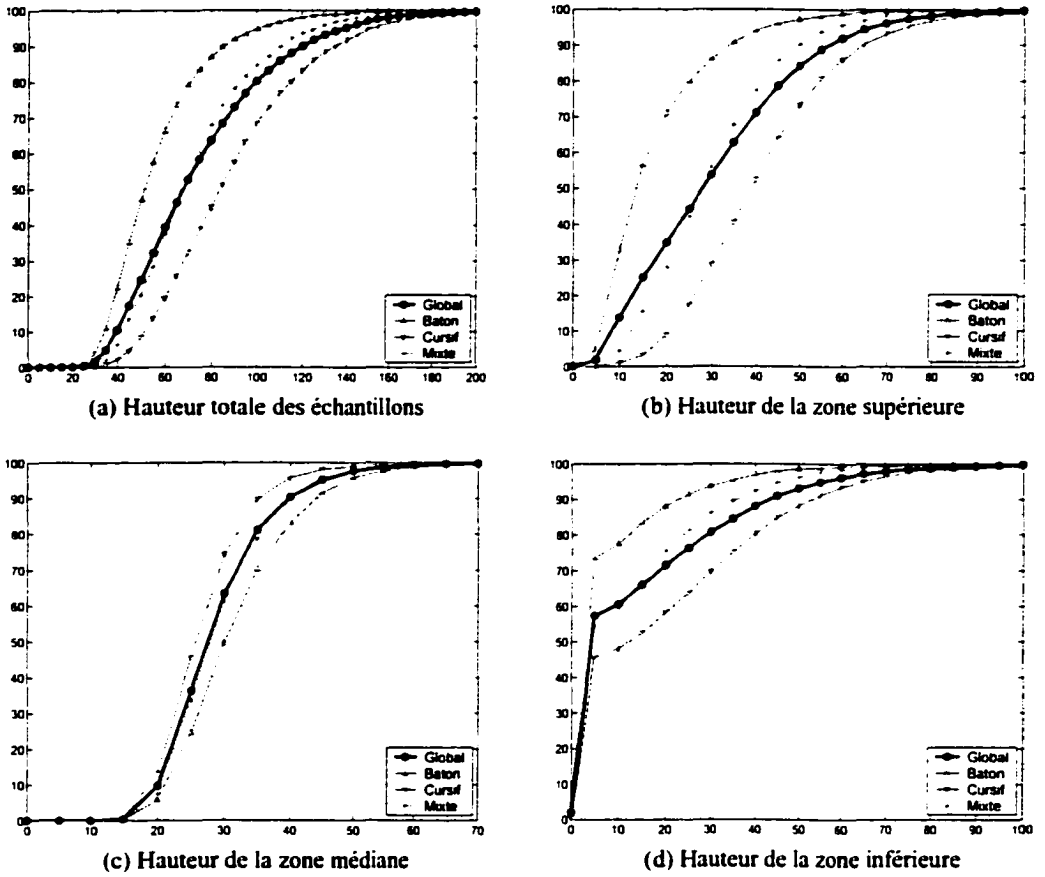


FIGURE 37 Définition de la hauteur des différentes zones d'écriture.

3.3.3.1 Analyse des zones d'écriture

Afin d'évaluer le comportement de la définition des zones d'écriture pour notre base de données, nous avons réalisé une analyse statistique de leur hauteur. Nous présentons sur la figure 38 les distributions cumulées de différentes hauteurs mesurées en pixels sur les échantillons de notre corpus d'apprentissage. L'analyse a été réalisée globalement ainsi qu'en fonction du type d'écriture, c'est-à-dire bâton, cursif et mixte. Pour les différents graphiques l'axe des abscisses correspond à la mesure associée en nombre de pixels.

Il est certain qu'une analyse des hauteurs en pixels est sensible à la résolution choisie lors de l'acquisition des images. Notre base de données étant homogène, c'est-à-dire que l'ensemble des images a été scanné avec la même résolution, l'analyse peut être effectuée directement. Afin de compléter cette dernière, nous présentons les valeurs moyennes en pixels de la hauteur des exemples et des différentes zones d'écriture dans le tableau VII.



(a) Hauteur totale des échantillons

(b) Hauteur de la zone supérieure

(c) Hauteur de la zone médiane

(d) Hauteur de la zone inférieure

FIGURE 38 Distributions cumulées des hauteurs en pixels des différentes zones d'écriture.

TABLEAU VII

Hauteur moyenne en pixels des différentes zones d'écriture – Présentation en fonction du type d'écriture des échantillons.

	Global	BÂTON	Cursif	MiXte
Hauteur totale	**75,3**	57,2	89,7	73,3
Zone supérieure	**31,3** (41,6%)	17,8 (31,1%)	41,6 (46,4%)	30,6 (41,7%)
Zone médiane	**29,4** (39,0%)	32,2 (56,3%)	27,3 (30,4%)	29,9 (40,8%)
Zone inférieure	**14,6** (19,4%)	7,2 (12,6%)	20,8 (23,2%)	12,8 (17,5%)

3.3.3.1.1 La hauteur totales des échantillons

La première conclusion suite à l'analyse du la figure 38(a) est que la hauteur des exemples n'est pas uniforme et qu'une nette différence peut être remarquée entre les exemples bâtons et cursifs. Ces derniers ont globalement une hauteur plus importante (90% des exemples bâtons ont une hauteur inférieure à 85 pixels alors que cette mesure atteint 135 pixels pour les cursifs). Ce phénomène est normal puisque l'écriture bâton ne possède pas de dépassement et le scripteur essaie généralement et naturellement de placer l'ensemble des lettres entre deux droites parallèles. Par contre dans le cas de l'écriture cursive, la présence de dépassements haut et bas conduit le scripteur à augmenter la hauteur globale de son écriture. Une dernière remarque concernant la hauteur totale des échantillons est que les exemples mixtes ont un comportement similaire à celui de l'ensemble des exemples.

3.3.3.1.2 La zone médiane

L'analyse de la hauteur de la zone médiane figure 38(c) montre qu'il existe une certaine stabilité pour l'ensemble des échantillons. En effet, 90% de la distribution est comprise entre 17 et 44 pixels. L'analyse des valeurs moyennes présentées à la 4è ligne du tableau VII permet de confirmer ce phénomène. D'une certaine manière cela traduit également l'homogénéité de notre base de données. Par rapport à la hauteur totale des exemples nous pouvons constater que la tendance s'est inversée, c'est-à-dire que ce sont maintenant les exemples bâtons qui ont une hauteur plus importante que les exemples cursifs. Ce phénomène est normal, puisque dans le cas des exemples bâtons seule la zone médiane devrait exister. De ce fait la hauteur totale des exemples de ce type se reporte principalement sur la zone médiane.

3.3.3.1.3 La zone supérieure

La figure 38(b) permet d'analyser le comportement de la zone des dépassements hauts. Nous pouvons constater une nette différence entre les exemples bâtons et cursifs. Les

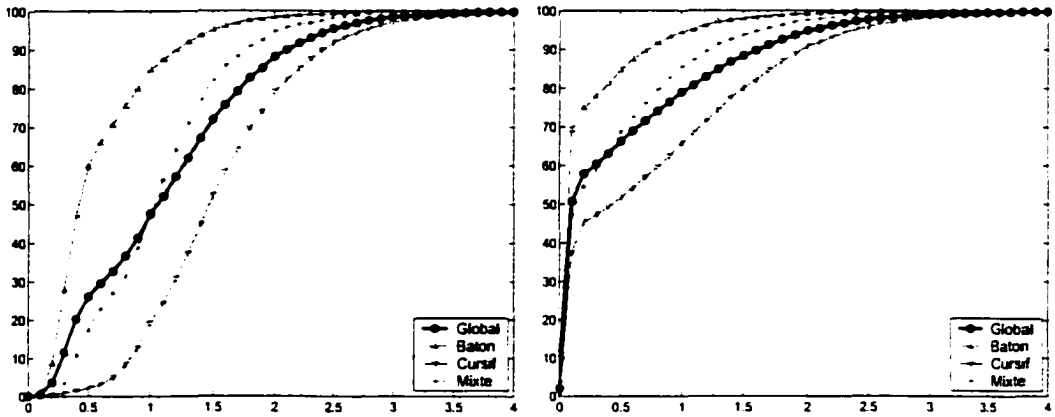
valeurs moyennes du tableau VII permettent de la quantifier. Globalement nous pouvons dire que la zone des dépassements hauts est deux fois plus grande pour les exemples cursifs que pour les exemples bâtons. Ce phénomène était attendu puisque les exemples bâtons ne devrait pas comporter de dépassements.

D'un autre point de vue, nous pouvons remarquer que seulement 5% des exemples bâtons ont un dépassement inférieur à 5 pixels. Cette remarque signifie que pour la très grande majorité des exemples bâtons, la présence d'une zone supérieure est détectée. Ce phénomène est normal car notre algorithme essaie toujours de détecter trois zones d'écriture. Cependant une heuristique permet de ne pas la prendre en compte si sa taille est inférieure à un tiers de celle de la zone médiane. Du point de vue des exemples cursifs, nous pouvons conclure sur la présence d'une zone de dépassements hauts significative pour plus de 90% des échantillons.

3.3.3.1.4 La zone inférieure

La zone des dépassements bas (figure 38(d)) n'a pas le même comportement que celui remarqué pour les dépassements hauts. En effet cette zone est inexistante pour une grande partie des échantillons. Cette zone n'est prise en compte lors de la définition des primitives perceptuelles que si elle a une taille supérieure à la moitié de la zone médiane.

Les heuristiques permettant de prendre en compte les dépassements ont été fixées en considérant l'apprentissage de l'écriture cursive au cours primaire. En effet la taille d'un dépassement bas est défini comme étant égale à celle du corps du mot. Dans le cas des dépassements hauts elle peut être égale (pour les lettres `t` et `d`) ou deux fois plus grande (pour `b`, `f`, `h`, `k` et `l`). Cette remarque permet également de justifier la différence de comportement remarquée entre les deux zones de dépassements.



(a) Rapport $\rho_H = \frac{h_{Sup}}{h_{Med}}$ (b) Rapport $\rho_B = \frac{h_{Inf}}{h_{Med}}$

FIGURE 39 Distributions cumulées des rapports de hauteur entre les zones de dépassements et la zone médiane.

3.3.3.2 Rapport entre les différentes zones

Afin de compléter notre analyse sur le comportement des zones de dépassements, nous avons calculé pour l'ensemble de notre corpus d'apprentissage deux rapports de hauteur (évalués en pixels) :

1. celui de la zone supérieure sur la zone médiane : ρ_H,
2. celui de la zone inférieure sur la zone médiane : ρ_B.

Sur la figure 37 nous avons indiqué les différentes mesures des zones d'écriture, elles sont estimées en pixels. Les deux rapports peuvent alors être définis de la manière suivante :

$$\rho_H = \frac{h_{Sup}}{h_{Med}} \tag{3.1}$$

$$\rho_B = \frac{h_{Inf}}{h_{Med}} \tag{3.2}$$

Les valeurs de ces rapports ont été collectées en fonction du type d'écriture. Nous présentons sur la figure 39 leurs distributions cumulées.

Une première constatation est que le comportement des deux rapports est différent. Ceci est normal étant données nos constatations sur la hauteur des zones de dépassements. L'analyse du graphique relatif à ρ_H montre que seulement 15% des exemples bâtons ont une zone supérieure plus importante que la zone médiane alors que pour les exemples cursifs cette quantité s'élève à 80%. Nous avons mentionné l'existence d'heuristiques permettant de prendre en compte ou non les zones de dépassements. Pour la zone supérieure, cette heuristique correspond à une valeur de $\rho_H > 1/3$. Nous pouvons alors quantifier sur le graphe que 60% des exemples bâtons ont un dépassement haut considéré comme significatif alors que cette quantité est de l'ordre de 99% pour les exemples cursifs.

L'analyse du graphique associé à ρ_B montre que seulement 5% des exemples bâtons ont une zone de dépassements bas plus grande que la zone médiane. Cette quantité augmente jusqu'à 15% pour les échantillons mixtes et 35% pour les cursifs. L'heuristique associée à cette zone correspond à : $\rho_B > 1/2$. Pour cette valeur, nous pouvons constater que 15% des exemples bâtons, 52% des exemples cursifs et environ 30% des exemples mixtes ont des dépassements bas significatifs.

3.3.3.3 Conclusions sur l'analyse de zones d'écriture

La première conclusion de ces analyses est que globalement les zones de dépassements ont des comportements différents. Deux raisons peuvent être identifiées :

1. de par son apprentissage de l'écriture, l'homme a tendance à effectuer des dépassements hauts plus grands que les dépassements bas. Cette consigne est certainement basée sur le fait que la ligne de base inférieure est plus stable que celle délimitant le corps des minuscules. Afin de mieux marquer un dépassement haut il est alors préférable de l'accentuer.
2. Les pré-traitements mis en œuvre, en particulier l'étape de normalisation de la ligne de base, influencent la taille des zones de dépassements. Ce processus permet de définir la ligne de base et de positionner l'écriture sur cette dernière. La transforma-

tion effectuée induit une modification de l'histogramme vertical de l'écriture qui est ensuite utilisé pour la définition des zones d'écriture.

Les zones de dépassements pour les exemples bâtons devraient être inexistantes. Cependant notre analyse a montré que ce n'était pas le cas. Nous pouvons également identifier deux causes à ce phénomène. La première est directement liée au scripteur qui n'écrit pas l'ensemble des caractères avec la même hauteur ou accentue la première lettre des mots (comme pour les fontes PETITES MAJUSCULES) ou encore donne un style à certaine lettre en allongeant la partie finale vers le haut ou le bas. La seconde source est directement liée aux étapes de pré-traitements. En effet la normalisation de la ligne de base inclut une transformation de l'image et peut induire une augmentation ou une réduction de la hauteur de certaines lettres.

Nous présentons sur la figure 40 un assortiment d'échantillons bâtons comportant des dépassements hauts ou bas significatifs, permettant d'illustrer ces deux points.

Les différentes analyses effectuées montrent bien la différence entre les exemples bâtons et cursifs. Nous avons constaté que les pré-traitements conditionnent la définition des zones d'écriture. En particulier pour les exemples bâtons, la stratégie adoptée n'est certainement pas optimale. Cependant il est certain que si l'extraction de primitives est réalisée sur le rectangle englobant des graphèmes, c'est-à-dire sans tenir compte des zones d'écriture, leur influence est nulle.

3.3.3.4 Différentes stratégies de *zoning* pour l'extraction de primitives

Parmi les trois zones d'écriture, la zone médiane est sans conteste celle qui contient le plus d'information. Les zones de dépassements nous informent sur leurs présences et accessoirement sur leurs formes. Par contre la zone médiane contient l'information sur le tracé principal des graphèmes ainsi que celle portant sur leurs connexions ou ligatures. Ce concept a été présenté par Simon [144] : en résumé, le corps du mot est considéré comme la partie régulière alors que les dépassements sont des singularités. Il est donc intéres-

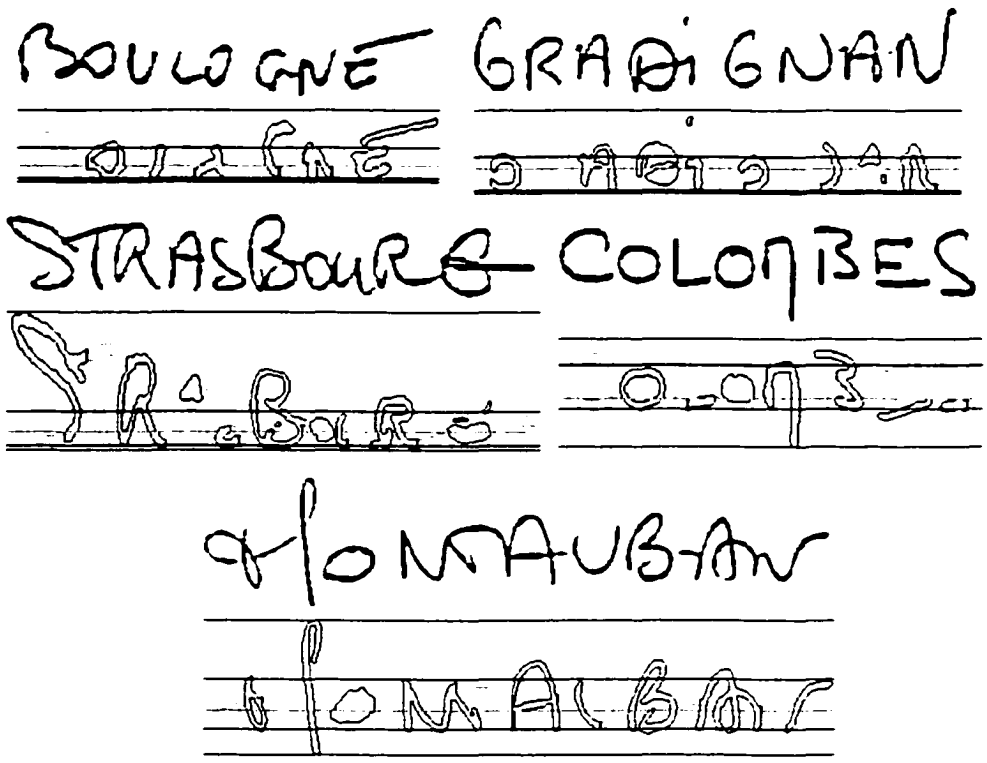


FIGURE 40 Plusieurs échantillons de type bâton comportant des dépassements significatifs (avant et après pré-traitements).

sant de rechercher une information discriminante dans la zone médiane. Dans cet ordre d'idée, nous proposons de tester différentes stratégies de *zoning*, basées sur la division de la zone médiane. Nous présentons sur la figure 41 les différentes configurations qui seront utilisées.

La première option proposée revient à prendre en compte simplement les trois zones d'écriture. Les trois suivantes consiste à diviser la zone médiane de différentes manières, en deux ou quatre zones. L'utilisation de la stratégie de *zoning* s'accompagnera obligatoirement d'une augmentation de la dimension des espaces de représentation, puisque l'extraction d'un vecteur caractéristique par zone sera réalisée.

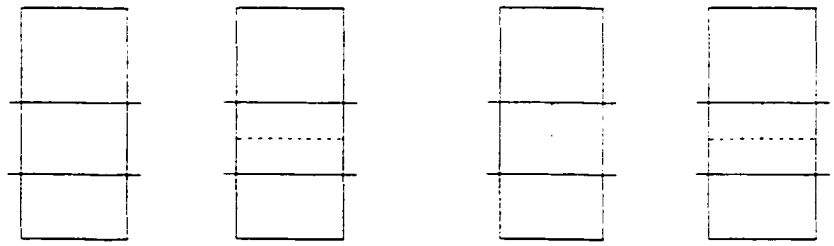


FIGURE 41 Présentation de quatre stratégies de *zoning* proposées pour l'extraction des primitives.

3.3.4 Pondération des zones d'écriture

Dans le cas de l'application de la technique de *zoning* à la reconnaissance de l'écriture manuscrite, il semble naturel de considérer les zones d'écriture. Notre réflexion est principalement basée sur les exemples cursifs qui sont plus difficiles à reconnaître. Pour ce type d'exemple, c'est la seule possibilité permettant de caractériser une information similaire dans une zone particulière. Cette technique permet de s'affranchir des problèmes de normalisation des hauteurs. Les primitives obtenues seront bien sûr conditionnées par l'algorithme de détection des zones d'écriture. Diverses constatations confirment que cette étape n'est pas parfaite, en particulier pour les exemples bâtons. En effet, nous avons constaté que pour une quantité non négligeable d'échantillons des zones de dépassements sont prises en compte. Elles ne correspondent pas à de réels dépassements puisque ce type d'écriture n'en possède pas, mais plutôt à des lettres stylisées ou encore des différences de hauteur entre les lettres d'un même mot. Dans ce dernier cas deux causes sont possibles, soit le scripteur a une écriture non régulière, soit les différentes transformations de l'image effectuées au cours des pré-traitements ont induit une déformation de certains caractères.

La prise en compte des zones d'écriture doit permettre de mieux considérer l'information contenue dans les graphèmes associés à l'écriture cursive. Cependant, dans le cas des exemples bâtons, elle risque d'introduire une certaine confusion. De ce fait nous proposons

de définir une technique de pondération ayant pour but la réduction de l'influence des deux zones de dépassements sur les échantillons de type bâton.

3.3.4.1 Définition de la pondération des zones

Le type d'écriture d'un nouvel exemple présenté à notre système ne peut pas être connu *a priori*. La technique de pondération doit alors traiter l'ensemble des échantillons indépendamment de cette information. Pour cela nous devons prendre en compte différentes remarques afin de définir une stratégie correcte. La première est que la zone médiane est toujours informante quelque soit le type d'exemples. De ce fait la technique de pondération ne doit pas réduire l'influence des caractéristiques extraites de cette zone. La seconde est que la taille des dépassements est généralement plus importante et plus informante pour les exemples cursifs que pour les bâtons (voir figure 38(b)). De plus nous avons montré que pour les exemples bâtons il est souvent non justifié de prendre en compte la zone supérieure.

Ces différentes constatations ont guidé la définition de notre stratégie de pondération. Premièrement seules les caractéristiques extraites des zones de dépassements seront pondérées. Pour cela, nous proposons d'utiliser comme facteur de pondération, la hauteur relative des zones de dépassements par rapport à la hauteur de la zone médiane. Cela signifie que pour l'ensemble des caractéristiques extraites des zones supérieures et inférieures, nous allons associer respectivement les facteurs de pondération ρ_H et ρ_B, définis par les équations 3.1 et 3.2. Comme nous désirons réduire l'influence des caractéristiques associées aux dépassements, la valeur de ces paramètres doit être majorée. En effet, sans cette borne, un effet inverse à celui désiré pourrait survenir. Pour éviter ce phénomène, les deux facteurs de pondération possèdent une borne supérieure égale à la valeur 1.

Nous proposons de prendre en compte la pondération lors du calcul de la distance entre échantillons et centres de gravité. Soit un échantillon $\mathbf{x}_i = [x_1^i, x_2^i, \ldots, x_N^i]^t$ dans un espace à N dimensions et un centre de gravité $\mathbf{z}_k = [z_1^k, z_2^k, \ldots, z_N^k]^t$ résultant de la quanti-

fication vectorielle. Les composantes de ce dernier sont obtenues de la manière suivante :

$$\mathbf{z}_j^k = \frac{1}{N_{Vk}} \sum_{i=1}^{N_{Vk}} x_j^i \tag{3.3}$$

où N_{Vk} est le nombre d'échantillons rattachés au centre de gravité $\mathbf{z}_k$. Soit n_H, n_M et n_B, le nombre de dimensions de l'espace de représentation relatif aux zones supérieure, médiane et inférieure respectivement. Nous pouvons donc écrire la relation : $N = n_H + n_M + n_B$. Considérons le vecteur de pondération $\mathbf{p}_i = [p_1^i, p_2^i, \ldots, p_N^i]^t$ associé à l'échantillon $\mathbf{x}_i$. Ses composantes sont obtenues de la manière suivante :

$$\begin{cases} p_k^i = \min(1, \rho_H^i) & \text{si} \quad 0 < k \leq n_H \\ p_k^i = 1 & \text{si} \quad d_H < k \leq n_H + n_M \\ p_k^i = \min(1, \rho_B^i) & \text{si} \quad n_H + n_M < k \leq N \end{cases} \tag{3.4}$$

Lors de notre quantification vectorielle, nous utilisons comme métrique la distance euclidienne d_E définie entre l'échantillon x_i et le centre z_k par :

$$d_E(\mathbf{x}_i, \mathbf{z}_k) = \sqrt{\frac{1}{N} \sum_{j=1}^{N} \left(x_j^i - z_j^k\right)^2} \tag{3.5}$$

La prise en compte de la pondération conduit à ré-écrire cette équation de la manière suivante :

$$d_{Ep}(\mathbf{x}_i, \mathbf{z}_j) = \sqrt{\frac{1}{P_i} \sum_{j=1}^{N} p_j^i \left(x_j^i - z_j^k\right)^2} \tag{3.6}$$

avec :

$$P_i = \sum_{j=1}^{N} p_j^i = (\rho_H^i \times n_H) + n_M + (\rho_B^i \times n_B) \leq N \tag{3.7}$$

L'utilisation de cette distance euclidienne pondérée d_{Ep} permettra de réduire l'influence des zones de dépassements lorsque leur taille est inférieure à celle de la zone médiane. Cependant il est certain que son utilisation aura des effets de bord, en raison de la non optimalité de la détection des zones d'écriture.

Cette approche a l'avantage de ne pas modifier les valeurs du vecteur caractéristique associé au graphème. Par contre, une de ces limitations est qu'elle n'est plus utilisable si l'espace de représentation subi une transformation, comme dans le cas de l'application de l'algorithme LDA. En effet dans le nouvel espace l'association composante du vecteur/zone d'écriture est perdue, puisque chaque nouvelle composante est une combinaison linéaire de celles du vecteur original. Nous avons donc envisagé la possibilité de pondérer directement les valeurs des composantes des vecteurs caractéristiques. Cela revient à considérer un nouveau vecteur caractéristique dans un espace de même dimension : $\mathbf{y}_i = [y_1^i, y_2^i, \ldots, y_N^i]^t$, pour lequel chaque composante est obtenue de la manière suivante :

$$y_j^i = p_j^i \times x_j^i \tag{3.8}$$

où le vecteur des pondérations $\mathbf{p}_i$ est inchangé (voir équation 3.4). Cette approche conduit à modifier directement les valeurs des composantes des différents vecteurs. Suite à cette pondération, nous pouvons envisager d'effectuer une normalisation des composantes du vecteur. Cependant une telle action conduit à modifier les valeurs associées à la zone médiane. Cela revient à modifier complètement la nature de l'espace de représentation. Une telle normalisation doit donc être évitée.

La prise en compte de la pondération de cette manière induit une modification importante au niveau de la quantification vectorielle puisque l'estimation des coordonnées des différents centres de gravité est modifiée. En effet, l'équation 3.3 devient :

$$\mathbf{z}_j^k = \frac{1}{N_{Vk}} \sum_{i=1}^{N_{Vk}} y_j^i = \frac{1}{N_{Vk}} \sum_{i=1}^{N_{Vk}} p_j^i \times x_j^i \tag{3.9}$$

Le calcul des centres de gravité inclut alors directement la pondération des zones d'écriture. De ce fait, lors de la recherche du centre de gravité le plus proche d'un échantillon, la distance euclidienne doit être utilisée. L'équation 3.5 peut s'écrire sous la forme :

$$d_E(\mathbf{y}_i, \mathbf{z}_k) = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(p_j^i\, x_j^i - z_j^k\right)^2} \tag{3.10}$$

$$= \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(p_j^i\, x_j^i - \frac{1}{N_{Vk}}\sum_{l=1}^{N_{Vk}} p_j^l\, x_j^l\right)^2} \tag{3.11}$$

Cette mesure est différente de la distance pondérée de l'équation 3.6. En effet cette dernière peut s'écrire de la manière suivante :

$$d_{Ep}(\mathbf{x}_i, \mathbf{z}_k) = \sqrt{\frac{1}{P_i}\sum_{j=1}^{N}\left(\sqrt{p_j^i}\, x_j^i - \sqrt{p_j^i}\, z_j^k\right)^2} \tag{3.12}$$

$$= \sqrt{\frac{1}{P_i}\sum_{j=1}^{N}\left(\sqrt{p_j^k}\, x_j^i - \sqrt{p_j^i}\,\frac{1}{N_{Vk}}\sum_{l=1}^{N_{Vk}} x_j^l\right)^2} \tag{3.13}$$

Le fait de prendre en compte la pondération au niveau des caractéristiques induit une modification des données de départ pour la construction du premier ensemble de primitives. Cela signifie que toutes les étapes suivantes seront affectées. En d'autres termes les deux stratégies de pondération conduiront à la construction de systèmes différents. La comparaison des équations 3.11 et 3.13 montrent qu'il est difficile d'obtenir une relation entre les deux stratégies. De plus il est impossible de prédire laquelle conduira aux meilleurs résultats.

L'évaluation des différents espaces de représentation mis en œuvre est présentée dans le chapitre 5. En effet, tous les résultats expérimentaux obtenus au cours de notre travail ont été regroupés dans ce chapitre de notre thèse.

3.4 Résumé

Dans ce chapitre nous avons traité de l'extraction de caractéristiques pour la reconnaissance de l'écriture manuscrite. La revue bibliographique de ce domaine a permis de constater la grande variété de techniques permettant d'extraire l'information de l'image de l'écriture. Le but de cette étape est de réduire la quantité d'information et d'essayer d'extraire la plus pertinente pour la tâche de reconnaissance. Elle est donc très importante dans un système puisque la représentation de l'information choisie conditionne tout le reste du processus de reconnaissance.

Dans le cas de la reconnaissance de l'écriture manuscrite, il ne faut pas perdre de vue lors du choix de la ou des techniques d'extraction à utiliser, que ce signal est soumis à plusieurs sources de variation. De ce fait il est conseillé de développer des primitives le plus insensibles possible à de faibles variations de forme (variation locale de forme). Dans cette optique, l'extraction de la structure des caractères semble la plus conseillée. Cependant l'information structurelle n'est pas toujours suffisante pour différentier certains caractères. Il est alors intéressant d'aller chercher une information plus locale de manière à compléter la représentation.

Différentes contraintes liées à l'application visée peuvent aider le concepteur d'un système dans le choix de la ou des techniques d'extraction de caractéristiques à mettre en œuvre, mais ce n'est pas toujours le cas. Les différentes options ne permettent pas toutes d'extraire le même type d'information de l'image. De plus il n'est pas facile de faire un pronostic quant aux résultats obtenus avec telle ou telle technique d'extraction. Il est alors naturel d'adopter l'approche qui consiste à extraire plusieurs ensembles de primitives, à partir d'espaces de représentation différents, puis d'effectuer une sélection des primitives les

plus pertinentes pour la tâche de reconnaissance visée. Cette technique est connu sous le nom de *sélection de caractéristiques* ou *feature selection* en anglais. Elle est le sujet du chapitre suivant.

CHAPITRE 4

LA SÉLECTION DE CARACTÉRISTIQUES

Tout système de reconnaissance ou de classification, traitant des données réelles comme des images ou encore des signaux acoustiques, doit passer par une phase d'extraction de primitives ou de caractéristiques. Elle permet de convertir les données réelles (mesures physiques, analyse statistique, réponse à un stimulus,...) dans un format propre à leur utilisation. Du fait de la nature numérique des systèmes actuels, cette représentation de l'information est caractérisée par une série de valeurs numériques qui traduisent, soit la présence ou l'absence (cas booléen) ou la valeur associée (cas réel) à la caractéristique concernée. La dimension de l'espace de représentation du phénomène analysé comporte alors autant de dimensions que de mesures effectuées sur les données de départ.

Dans le cas de l'analyse de signaux complexes le nombre de dimensions peut rapidement devenir élevé, de l'ordre de plusieurs dizaines. Parmi l'ensemble de mesures effectuées sur le phénomène originel, toutes ne sont pas aussi pertinentes. Il est possible que certaines correspondent à du bruit ou qu'elles soient peu informantes, corrélées ou même inutiles au système pour l'accomplissement de sa tâche. D'après Jain [75], la performance d'un système de classification dépend fortement des relations entre le nombre d'échantillons utilisés, le nombre de caractéristiques considérées et la complexité du système. En effet, pour obtenir un système performant il est nécessaire d'avoir un nombre d'échantillons suffisamment grand et représentatif des différents phénomènes à modéliser, de manière à estimer correctement ses paramètres. De plus théoriquement une relation exponentielle lie le nombre d'échantillons au nombre de caractéristiques utilisées par le système : pour l'ajout d'une nouvelle caractéristique, le nombre d'échantillons doit être augmenté de manière exponentielle. Ces différentes remarques montrent clairement qu'il est nécessaire, lors de la construction d'un système, de limiter le nombre de caractéristiques prise en compte, de manière à optimiser ses performances. De ce fait la sélection de caractéristiques est un do-

maine de recherche actif depuis plusieurs décennies. De nombreux travaux et publications traitent de ces techniques qui sont appliquées dans un grand nombre de domaines.

Le but de notre projet est d'améliorer les performances d'un système de reconnaissance de l'écriture manuscrite. Nous avons conclu de l'analyse de ses performances que son principal point faible est la faiblesse de la représentation de l'information utilisée. De ce fait il est pertinent d'intégrer de nouvelles sources d'information. Dans ce but nous avons proposé de développer plusieurs espaces de représentation de manière à extraire des primitives de type différent permettant d'apporter une information complémentaire. Le système utilise actuellement deux ensembles de primitives. L'intégration d'informations à partir de ces deux sources se fait en les combinant directement au niveau de la création de l'ensemble de primitives. L'intégration d'une ou plusieurs nouvelles sources conduira à une augmentation exponentielle du nombre de paramètres du système. D'un autre coté, nous ne pouvons pas augmenter le nombre d'échantillons en conséquence. Il est alors impossible d'utiliser la même technique afin d'intégrer de nouvelles sources d'information. Dans ce but nous avons développé un nouvel algorithme permettant de sélectionner et d'intégrer différentes sources d'information dans le système. Ce dernier s'apparente au domaine de la sélection de caractéristiques.

Dans ce chapitre, nous allons présenter le domaine de la sélection de caractéristiques en donnant quelques définitions. La seconde section permettra de passer en revue les différentes approches possibles. Nous poursuivrons alors par une présentation de notre algorithme.

4.1 Le domaine de la sélection de caractéristiques

La sélection de caractéristiques est une technique permettant de choisir les caractéristiques, variables ou mesures les plus intéressantes, pertinentes ou informantes, à un système donné, pour la réalisation de la tâche pour laquelle il a été conçu. Cette phase est généralement un module important d'un système complexe. Les domaines d'application

des techniques de sélection de caractéristiques sont variés tels que la modélisation, la classification, l'apprentissage automatique (*Machine Learning*) et l'analyse exploratoire de données (*Data Mining*). Dans ce mémoire nous nous intéressons plus particulièrement à la sélection de variables pour la classification.

Dans [75], les auteurs proposent une revue des techniques statistiques utilisées en reconnaissance de formes. Une différence est faite entre les techniques d'extraction (*Feature Extraction*) et de sélection (*Feature Selection*) de variables. Les premières permettent de créer de nouveaux ensembles de caractéristiques, en utilisant une transformation ou une combinaison d'un espace de départ et en effectuant une réduction du nombre de dimensions. Des techniques bien connues se rattachant à cette catégorie sont : l'analyse en composantes principales (ACP) ou *Karhunen-Loève expansion* [44], l'analyse en composantes indépendantes [72] (plus appropriée que l'ACP pour les distributions non-gaussiennes), l'analyse discriminantes linéaire (LDA) [44]. Ce ne sont pas ces techniques qui nous intéressent, elles ne seront donc pas plus détaillées. Les techniques de sélections correspondent aux algorithmes permettant de sélectionner un sous-ensemble de caractéristiques parmi un ensemble de départ, en utilisant divers critères et différentes méthodes. Ce sont ces techniques qui nous intéressent et qui seront étudiées dans la suite de cette section.

4.1.1 Définition de la sélection de caractéristiques

La définition de la sélection de caractéristiques proposée par Pudil *et al.* dans [132] est la suivante : étant donnée une fonction permettant de mesurer la qualité d'un sous-ensemble de primitives, la sélection de variables est réduite au problème de la recherche du sous-ensemble optimal par rapport à cette mesure.

Dans [74, 75], l'énoncé de la sélection de caractéristiques proposée est la suivante : étant donné un ensemble de dimension n, il faut sélectionner le sous-ensemble de dimension m tel que $m < n$, conduisant au taux d'erreur le plus faible. Liu dans [108] ajoute que l'objectif est de réduire le taux d'erreur ou tout autre critère de sélection raisonnable.

Dans [95] les auteurs énoncent une liste de trois objectifs pour réaliser une sélection de primitives pour la classification :

- réduire la tâche d'extraction de primitives,
- améliorer la précision du module de classification,
- améliorer la fiabilité de l'estimation de la performance.

Finalement, Dash [27] propose de regrouper les techniques de sélection de caractéristiques en fonction de l'objectif visé. Il identifie alors quatre classes distinctes :

1. "*Idealized*" : trouver le sous-ensemble de taille minimale qui est nécessaire et suffisant pour atteindre l'objectif fixé.
2. "*Classic*" : sélectionner le sous-ensemble de m variables à partir de l'ensemble en contenant n, $m < n$, tel que la fonction critère choisie soit optimisée pour tous les sous-ensembles de taille m.
3. "*Improving prediction accuracy*" : choisir un sous-ensemble de caractéristiques afin d'améliorer la précision de la prédiction ou diminuer la taille de la structure sans diminution significative de la précision de prédiction du classificateur, construit en utilisant seulement les variables sélectionnées.
4. "*Approximating original class distribution*" : sélectionner un sous-ensemble de variables tel que la distribution des classes résultante soit aussi proche que possible de la distribution des classes étant donné l'ensemble des variables complet.

Ces deux dernières approches sont reprises par Koller et Sahami dans [92]. La définition de la sélection de caractéristiques diffère suivant l'auteur. L'avantage de cette dernière est qu'elle est générale et permet ainsi de regrouper l'ensemble des algorithmes.

Afin de présenter les différentes techniques de sélection des caractéristiques, nous allons adopter une approche semblable à celle de Leray dans sa thèse [101]. Il propose de considérer trois composantes pour la mise en œuvre d'un algorithme de sélection de caractéristiques :

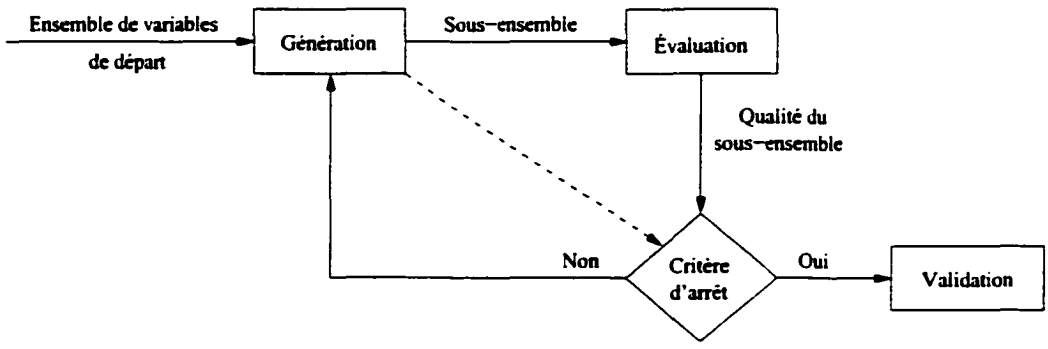


FIGURE 42 Représentation graphique du processus de sélection de caractéristiques d'après [27].

1. un critère d'évaluation des variables, permettant de comparer les différents sous-ensembles et de choisir le meilleur ;
2. une procédure de recherche, permettant d'explorer les différentes combinaisons de caractéristiques ;
3. un critère d'arrêt, permettant de stopper la procédure de recherche ou de choisir le sous-ensemble de caractéristiques à sélectionner.

Dash et Liu proposent dans [27] une quatrième composante : une procédure de validation permettant de vérifier que le sous-ensemble sélectionné est valide. Le schéma de la figure 42 représente cette approche. Les techniques permettant de mettre en œuvre ces différents modules seront détaillées dans la suite de cette section.

4.1.2 Évaluation des caractéristiques

Différents auteurs énoncent que le meilleur critère d'évaluation est le taux de reconnaissance du classificateur, mais que cette mesure est rarement fiable du fait qu'elle est obtenue à partir d'un nombre fini et limité d'échantillons. Il est également important de mentionner qu'un sous-ensemble de caractéristiques est optimal uniquement par rapport au critère d'évaluation. De ce fait, le choix de ce critère est très important. Il existe différentes possibilités quant à cet indicateur.

4.1.2.1 Mesures de distance

Dans la littérature elles sont également appelées mesure de séparabilité, de divergence ou de discrimination. Dans le cas d'un problème à deux classes, une variable X est préférée à Y, si X introduit une plus grande différence dans la distribution de probabilité conditionnelle des classes que Y. Un exemple de ce critère est la distance euclidienne.

4.1.2.2 Mesures d'information

Les indicateurs de cette catégorie sont basés sur le gain en termes d'information que peut apporter une variable. Il est défini pour une variable X comme la différence entre l'incertitude *a priori* et celle *a posteriori*, c'est-à-dire avant et après sélection de la variable X. Parmi deux variables alternatives, la variable sélectionnée est bien sûr celle qui apporte le plus d'information. L'entropie ou l'information mutuelle sont des critères appartenant à cette catégorie.

4.1.2.3 Mesures de dépendance

Elles sont également appelées mesure de corrélation ; elles permettent de mesurer la capacité d'une variable à prédire la valeur d'une autre. Ce critère peut être utilisé afin de mesurer la corrélation entre une variable et une classe de la modélisation. Si la corrélation d'une variable X avec une classe C est plus importante que la corrélation de la variable Y avec C alors X est choisie. Une variante peut également être utilisée ; elle consiste à mesurer la dépendance entre primitives et ainsi obtenir un indicateur de redondance entre les variables.

4.1.2.4 Mesures de cohérence

Ce sont des mesures apparues plus récemment dans la littérature ; elles sont différentes des autres dans le sens où elles se basent fortement sur le corpus d'apprentissage et qu'elles

utilisent le *Min-Features Bias*. Cette procédure permet de sélectionner le plus petit sous-ensemble de variables basé sur des hypothèses de cohérence forte. Un algorithme utilisant cette approche est *FOCUS*, il est détaillé, ainsi que l'approche *Min-Features Bias*, dans [3, 4].

4.1.2.5 Mesures de l'erreur de classification

Pour cette catégorie le classificateur est utilisé afin d'évaluer le sous-ensemble de variables ou primitives. Les algorithmes de sélection de caractéristiques utilisant cette mesure sont appelés *Wrapper*. Les résultats obtenus par de tels algorithmes sont fiables mais au prix d'un temps de calcul important.

Les algorithmes de sélection de caractéristiques sont classés en deux groupes par John *et al.* [77], ainsi que d'autres auteurs, en fonction du critère d'évaluation utilisé : *Filter Model* et *Wrapper Model*. L'approche de *Filter* est réalisée comme un pré-traitement. C'est-à-dire que la sélection se fait sans tenir compte de son influence sur les performances du système. Par opposition, l'approche *Wrapper* considère cette influence en utilisant le système pour évaluer la qualité du sous-ensemble de caractéristiques sélectionnées.

4.1.3 Procédure de recherche

La procédure de recherche consiste à générer les sous-ensembles de caractéristiques qui seront évalués. Si l'ensemble de départ contient n caractéristiques, le nombre total de sous-ensembles possibles est $2^n - 1$. Leur évaluation exhaustive n'est pas envisageable, même pour des valeurs moyennes de n. Différentes techniques permettent alors de générer les sous-ensembles candidats. Dans [1] les auteurs regroupent les approches en trois catégories : génération complète, heuristique et aléatoire. Nous allons les passer en revue dans la suite de cette section.

4.1.3.1 Génération complète

Aussi quelquefois appelées exponentielles, les approches regroupées dans cette catégorie effectuent une recherche complète du sous-ensemble optimal par rapport à la fonction d'évaluation choisie. Plusieurs fonctions heuristiques peuvent être utilisées de manière à réduire l'espace de recherche. De ce fait la recherche n'est pas exhaustive. Un point important est que l'optimalité du sous-ensemble, toujours par rapport à la fonction d'évaluation choisie, est garantie car ces approches utilisent une procédure de *Backtracking*. Cette phase est réalisée à l'aide d'algorithmes comme : *Branch and Bound* , *Best First Search* ou *Beam Search*.

4.1.3.2 Génération heuristique

Également appelée séquentielle, cette catégorie regroupe les algorithmes itératifs pour lesquels chaque itération permet de sélectionner ou rejeter une ou plusieurs variables. Ils ont l'avantage d'être assez simples et rapides. Cependant ils ne permettent de parcourir généralement qu'un petit sous-espace de l'espace total des possibilités. Au sein de l'approche heuristique, la littérature fait mention généralement de trois sous catégories :

1. *Forward* : cette approche part d'un ensemble de variables vide auquel, à chaque itération sont ajoutées une ou plusieurs variables. Elle est également appelée approche ascendante.
2. *Backward* : c'est l'approche inverse ; l'ensemble total des variables est considéré au départ de la procédure itérative, chaque itération permet d'en supprimer. Une autre appellation est approche descendante.
3. *Stepwise* : cette dernière approche est une version hybride des deux précédentes car elle consiste à ajouter ou retirer successivement des variables à l'ensemble déjà sélectionné.

Un grand nombre d'algorithmes ont été développés, basés sur une de ces trois approches de génération de sous-ensembles.

4.1.3.3 Génération aléatoire

Cette dernière catégorie est la plus récente dans le domaine de la sélection de caractéristiques. Les techniques employées ne parcourent qu'une partie de l'espace des solutions. Chaque procédure de génération aléatoire nécessite le choix de différents paramètres. L'obtention de bons résultats à l'aide de ces techniques nécessite un choix judicieux de ces paramètres. Les algorithmes génétiques ainsi que les techniques de recuit simulé font partie de cette catégorie.

La procédure de recherche associée à la sélection de variables tient une place importante. En effet il est quasiment impossible d'effectuer une recherche exhaustive de l'espace des solutions. La technique de recherche choisie conditionnera donc le sous-espace exploré et donc l'optimalité de la solution finale.

4.1.4 Critère d'arrêt

Une fois le critère d'évaluation des variables et la méthode de recherche définis, tous les sous-ensembles proposés par cette dernière sont évalués et celui retenu est le plus pertinent dans le sens du critère d'évaluation.

Dans le cas des méthodes de sélection séquentielle, le critère d'arrêt est fortement conditionné par la mesure de pertinence des variables. La recherche est arrêtée lorsqu'aucune des variables restantes n'est jugée pertinente. La pertinence d'une variable peut être obtenue par le calcul de tests statistiques.

Lorsque le calcul de ces tests n'est plus possible, à cause de leur complexité, une dernière solution est l'utilisation d'heuristiques. Une des heuristiques couramment utilisée est le calcul d'une estimation de l'erreur de généralisation pour les différents sous-ensembles testés. Le sous-ensemble choisi à la fin de la procédure est bien sûr celui qui donne les

meilleures performances. L'erreur de généralisation peut être calculée à l'aide d'un ensemble de validation, par validation croisée ou par d'autres estimations algébriques [101].

4.1.5 Procédure de validation

Dash et Liu [27] proposent d'ajouter une quatrième composante à un algorithme de sélection de caractéristiques : une procédure de validation. Deux alternatives sont proposées en fonction de la nature des données utilisées lors de cette procédure : artificielles ou réelles. Généralement une base de données synthétiques est construite dans le but de tester un concept ou une application particulière. De ce fait les variables pertinentes sont connues et identifiées. La validation d'un algorithme sera alors directe puisqu'il suffit de vérifier si le sous-ensemble retenu contient bien les variables pertinentes.

Dans le cas de données réelles, les variables pertinentes ne sont généralement pas connues. La procédure consiste alors à évaluer la précision de la classification obtenue avec le sous-ensemble de variables sélectionnées par l'intermédiaire d'un classificateur (classificateur de Bayes,...). Cette dernière peut alors être comparée à d'autres approches ou à celle obtenue par des techniques classiques.

4.2 Revue des différentes méthodes

Un algorithme de sélection de caractéristiques est composé de plusieurs modules. Étant donné que plusieurs approches sont possibles pour leur développement, il en résulte qu'un grand nombre d'algorithmes est disponible. Il est possible de trouver dans la littérature du domaine plusieurs articles traitant de la comparaison de différentes méthodes : [1, 27, 74, 94]. En particulier dans [27] les auteurs décrivent 32 algorithmes différents. Pour cela ils utilisent une catégorisation basée sur les techniques utilisées pour développer le module d'évaluation et celui de génération. Leur analyse montre qu'il n'existe aucun algorithme pour certaines combinaisons de ces deux composantes. Cela signifie qu'il existe des possibilités de développement de nouveaux algorithmes de sélection de caractéristiques.

Dans [95], les auteurs comparent des algorithmes de sélection de variables pour la classification. Un regroupement des méthodes est effectué en fonction de l'optimisation réalisée par la méthode. La première catégorie regroupe les algorithmes qui permettent de trouver le sous-ensemble de variables d'une dimension spécifiée, permettant la meilleure discrimination entre les classes de la modélisation. Ceux optimisant la taille du sous-ensemble sous une contrainte de discrimination minimale sont regroupés dans une seconde catégorie. La troisième contient ceux qui recherche un compromis entre la taille du sous-ensemble et son pouvoir de discrimination des classes.

Différentes approches de la sélection de caractéristiques seront décrites dans la suite de cette section. Une catégorisation a été utilisée, elle est fonction de la technique utilisée pour la procédure de génération des sous-ensembles. Pour chacune d'elles, les approches seront alors présentées suivant le critère d'évaluation utilisé.

4.2.1 Procédures de recherche complète

4.2.1.1 Mesure de distance

Un des algorithmes les plus anciens en sélection de caractéristiques est l'algorithme *Branch and Bound* (*B&B*) [121]. Il fournit une solution optimale mais à une condition : le critère d'évaluation des sous-ensembles doit être monotone. Cette contrainte limite bien sûr le choix de la mesure de distance. Généralement celles qui sont utilisées sont la distance de Mahalanobis, la distance de Bhattacharya, le critère de Fisher, la fonction discriminante et la divergence. Cette approche couvrant une grande partie de l'espace des solutions, elle nécessite un grand nombre d'opérations. Elle apporte tout de même une réduction par rapport à une approche exhaustive. Cependant, lorsque le nombre de variables devient important (> 30), il n'est tout de même pas raisonnable d'utiliser cette approche.

Dans [43] les auteurs proposent une version de l'algorithme *B&B* non-optimale mais permettant d'utiliser des critères d'évaluation non monotones. Pour cela ils définissent le

concept de monotonicité approximative et proposent une version relaxée de cet algorithme (*Relaxed Branch and Bound*).

Kudo et Sklansky [95] proposent une autre variante en définissant le concept de k-monotonicité, permettant de violer ce critère sous certaines conditions.

4.2.1.2 Mesure de cohérence

L'algorithme *FOCUS* [3, 4] fait également partie des méthodes appelées complètes. Contrairement à *B&B* qui utilise une mesure de distance, ce dernier algorithme utilise un critère d'évaluation des variables basé sur la cohérence. L'inconvénient de cet algorithme est qu'il ne fonctionne pas correctement lorsque les données sont bruitées. De plus il ne peut être utilisé que pour des problèmes de classification à deux classes. Il existe des variantes de cet algorithme ne permettant pas cependant d'abolir ces restrictions.

4.2.1.3 Mesure de l'erreur de classification

Il existe également plusieurs algorithmes utilisant l'erreur de classification comme critère d'évaluation des variables. Une variante de l'algorithme *B&B*, appelée *Approximate Monotonic Branch & Bound*, est présentée dans [43]. L'algorithme *Beam Search* en fait également partie.

4.2.2 Procédures de recherche heuristique

Cette catégorie est celle qui compte le plus d'algorithmes. La principale raison est que la mise en œuvre de ces derniers est relativement simple et rapide en terme de génération de résultats.

4.2.2.1 Mesure de distance

Le principal représentant de cette catégorie est l'algorithme *Relief* [84]. La procédure de sélection est réalisée par l'intermédiaire d'une pondération des différents caractéristiques. Premièrement l'utilisateur doit fixer le nombre d'échantillons que l'algorithme choisira aléatoirement dans le corpus d'apprentissage. Pour chacun d'eux il recherche au sein de ce sous-ensemble l'échantillon de la même classe le plus proche et celui d'une classe différente le plus proche également. Les poids des caractéristiques sont mis à jour en fonction de ces valeurs. À la fin du processus les caractéristiques sélectionnées sont celles qui ont une pondération supérieure à un seuil donné. Ce dernier peut être déterminé automatiquement. L'algorithme *Relief* permet une sélection efficace lors de la présence de variables corrélées. Cependant il ne détecte pas la présence de caractéristiques redondantes. Une restriction importante est qu'il ne fonctionne que dans le cas de la présence de deux classes. Konnenko propose une extension aux problèmes multi-classes [93]. Cette version de l'algorithme est appelée *Relief-F*.

4.2.2.2 Mesure d'information

Un premier algorithme utilisant une procédure de recherche heuristique et une mesure d'information est *Decision Tree Method (DTM)* [18]. L'approche consiste à utiliser l'algorithme de construction d'arbre C4.5 [134] sur un corpus de données. Les primitives sélectionnées sont alors celles appartenant aux différentes branches de l'arbre de décision résultant.

Une deuxième méthode est présentée par Koller et Sahami dans [92]. Elle est basée sur l'intuition que quelle que soit la primitive, si celle-ci n'apporte que très peu ou pas d'information par rapport aux autres, elle est soit non-pertinente, soit redondante et doit alors être éliminée. La technique présentée utilise le concept d'enveloppe de Markov. Certaines approximations quant à son implantation conduisent à l'obtention de solutions sous optimales.

4.2.2.3 Mesure de dépendance

La méthode *POE1ACC* pour *(Probability of Error & Average Correlation Coefficient)* [120] fait partie de cette catégorie. Le point de départ de cet algorithme est le choix de la caractéristique conduisant à la plus petite erreur de probabilité. Le processus consiste ensuite à sélectionner celle qui conduit à la somme pondérée des probabilités d'erreur minimal et au plus petit coefficient de corrélation moyen (ACC). Ce dernier correspond à la moyenne des coefficients de corrélation entre la caractéristique candidate et celles déjà sélectionnées. Le processus s'arrête lorsque le nombre de variables désirées est atteint.

Une seconde approche est présentée dans [117], son appellation est *PRESET*. Elle utilise la théorie des ensembles d'approximations (*Rough Sets*). La première étape consiste à trouver un sous-ensemble de caractéristiques permettant de classifier les échantillons aussi bien que l'ensemble global. Les caractéristiques n'appartenant pas à ce sous-ensemble sont alors éliminées. Les caractéristiques sont alors triées en fonction de leur signifiance. Cette mesure permet de quantifier l'importance qu'une caractéristique a pour la classification des échantillons. Elle est basée sur la dépendance des variables.

4.2.2.4 Mesure de l'erreur de classification

Cette combinaison est la plus représentée en nombre d'algorithmes dans la littérature, en raison de la simplicité de sa mise en œuvre. Les algorithmes de cette catégorie sont appelés *Wrappers*. Leur spécificité vient de l'utilisation de l'algorithme d'apprentissage ou d'induction du système pour évaluer la pertinence des caractéristiques.

Les deux algorithmes les plus communs de cette catégorie sont certainement : *Sequential Forward Selection (SFS)* et *Sequential Backward Selection (SBS)*. D'après [132], ils ont été respectivement présentés en 1963 par Marill et Green et 1971 par Whitney. Le point de départ de l'algorithme *SFS* est un sous-ensemble de primitives vide. La première étape consiste à évaluer toutes les primitives individuellement et à sélectionner celle qui conduit

à la meilleure performance de classification. Par la suite, chaque primitive non encore sélectionnée est testée conjointement avec le sous-ensemble de celles qui le sont déjà. La primitive sélectionnée à chaque itération est toujours celle qui conduit à la meilleure performance. La procédure s'arrête lorsque la performance du système n'est plus augmentée. L'algorithme *SBS* procède de manière similaire sauf que le point de départ est l'ensemble complet de caractéristiques et que chaque étape permet d'en supprimer une. Dans ce cas la primitive supprimée est celle qui conduit à la moins bonne performance.

L'approche *Forward* est plus rapide que l'approche *Backward*, du fait que son point de départ est en ensemble vide. Ces deux méthodes souffrent de ce qui est appelé le *Nesting Effect*. Il traduit le fait que, pour l'approche *Forward*, une primitive sélectionnée au départ sera conservée jusqu'à la fin de la procédure et respectivement pour l'approche *Backward*, qu'une primitive écartée ne pourra plus être sélectionnée. Ce problème n'est pas négligeable du fait que le meilleur sous-ensemble de n primitives ne contient pas forcément le meilleur sous-ensemble de $n-1$ primitives. Il est fait mention dans la littérature que l'algorithme *SBS* donne, dans certaines conditions, de meilleurs résultats que *SFS*. La raison invoquée est que l'algorithme *SBS* permet d'évaluer la contribution d'une variable dans le contexte de toutes les autres variables, alors que *SFS* évalue la pertinence d'une primitive dans le contexte réduit de celles déjà sélectionnées. Aha et Bankert dans [1] proposent une comparaison complète de ces deux approches.

Une version généralisée de ces deux algorithmes $GSFS(g)$ et $GSBS(g)$ a été présentée par Kittler [85]. Dans ces versions, g variables sont évaluées conjointement plutôt qu'une seule et ainsi un groupe de g variables est choisi, pour addition dans le cas de $GSFS(g)$ et pour suppression dans le cas $GSBS(g)$.

Une autre variante consiste à alterner ajout et suppression de variables. Les méthodes utilisant cette approche sont connues sous le nom de *Stepwise*. L'algorithme *Plus-l Take-away-r* ($PTA(l,r)$) en fait partie. Il consiste à utiliser l fois l'algorithme *SFS* de manière à

ajouter l variables, puis d'utiliser r fois l'algorithme *SBS* afin d'en supprimer r. Ces étapes sont alors réitérées jusqu'à l'obtention du critère d'arrêt. Il existe également une version généralisée de cet algorithme : $GPTA(l, r)$. Il consiste à utiliser $GSFS(l)$ et $GSBS(r)$ au lieu de *SFS* et *SBS*.

Pudil *et al.* [132] proposent une version flottante des algorithmes *SFS* et *SBS* : *SFFS* et *SFBS*. Leur objectif est de supprimer le problème de *Nesting Effect* en utilisant l'algorithme $PTA(l, r)$ ou $GPTA(l, r)$. La dimension du sous-ensemble à chaque étape sera alors dépendante des valeurs de l et r. Les valeurs optimales de ces paramètres ne pouvant pas être déterminées théoriquement, les auteurs proposent de les laisser flottantes au cours du processus de sélection afin de se rapprocher au maximum de la solution optimale. Les auteurs concluent de leur expérimentation que ces deux algorithmes obtiennent des résultats équivalents à l'algorithme optimal *Branch & Bound*, mais beaucoup plus rapidement. Une autre alternative est présentée dans [54].

Il existe également d'autres variantes des deux algorithmes de base que sont *SFS* et *SBS*, nous venons de passer en revue les principaux. Pour plus d'informations à leur sujet, il est possible de consulter la revue effectuée par Dash et Liu [27].

4.2.3 Procédures de recherche aléatoire

Il n'existe que peu d'algorithmes de sélection de caractéristiques utilisant une procédure aléatoire de génération et une mesure d'évaluation différentes du taux d'erreur de classification.

4.2.3.1 Mesure de cohérence

La principale approche utilisant comme critère d'évaluation une mesure de cohérence est l'algorithme *Las Vegas Filter* (*LVF*) proposé par Liu et Setiono [107]. Son nom lui vient de l'utilisation de l'algorithme *Las Vegas* [14] pour la génération des sous-ensembles de

caractéristiques. La mesure de cohérence utilisée est différente de celle de l'algorithme *FOCUS*. L'algorithme *LVF* met à jour un compteur d'incohérence pour chaque sous-ensemble testé. Ce dernier est basé sur l'intuition que la classe la plus représentée parmi les échantillons rattachés au sous-ensemble évalué est la classe la plus probable. Les sous-ensembles obtenant un score d'incohérence supérieur à un seuil fixé sont éliminés. Cet algorithme n'évalue le critère d'incohérence que pour les sous-ensembles de taille inférieure ou égale à celle du meilleur jusqu'alors testé. Il est efficace, facile à mettre en œuvre et garantit l'obtention du sous-ensemble optimal. Un des inconvénients de cette approche est le temps nécessaire à la génération de résultats. En effet, pour certains problèmes, il peut devenir plus important que celui d'une approche heuristique.

4.2.3.2 Mesure de l'erreur de classification

L'utilisation conjointe d'une procédure de recherche aléatoire et d'une mesure d'évaluation basée sur l'erreur de classification est représentée par plusieurs algorithmes : *Las Vegas Wrapper* (*LVW*) [106], les algorithmes génétiques [133, 136, 151], le recuit simulé [28], *Random Mutation Hill Climbing* (*RMHC*) [116, 145].

L'algorithme *LVW* génère les sous-ensembles de manière tout à fait aléatoire. Dans le cas des algorithmes génétiques et du recuit simulé, l'approche consiste à utiliser une procédure spécifique de génération des sous-ensembles, conservant une certaine continuité, mais intégrant des composantes aléatoires. L'inconvénient majeur de ces deux dernières approches est qu'elles ne garantissent pas l'optimalité de la solution finale.

L'algorithme *RMHC-PF1* (*PF* pour *Feature and Prototype*) [145] effectue une sélection d'échantillons prototypes et de caractéristiques dans un même temps. Un vecteur de codage unique contient ces informations. À chaque itération une composante du vecteur choisie aléatoirement subit une mutation et ainsi produit un nouveau sous-ensemble. La fonction d'évaluation utilisée est la précision de classification par l'algorithme du plus

proche voisin, sur le corpus d'apprentissage. L'algorithme s'arrête lorsqu'il a atteint un certain nombre d'itérations fixé au départ.

Une des contraintes liées à l'ensemble de ces algorithmes est qu'il faut fixer de manière adéquate un certain nombre de paramètres, comme le nombre d'itérations, de manière à assurer leur convergence.

4.2.4 Wrapper *versus* Filter

Comme nous l'avons mentionné dans la section 4.1.2, une classification des algorithmes de sélection de caractéristiques est faite en fonction du critère d'évaluation utilisé. L'approche *Wrapper* consiste à utiliser la précision en termes de classification et donc l'algorithme d'apprentissage ou d'induction du système. Pour la seconde approche, *Filter*, la sélection est réalisée sans tenir compte de son influence sur la performance du système. Dans le cas d'un système de reconnaissance, elle peut être assimilée à une étape de pré-traitement. Elle est bien sûr plus rapide que l'approche *Wrapper* en terme de génération de résultats. Cependant cette dernière à l'avantage de fournir généralement des résultats plus pertinents pour la classification.

Hen et Liu dans [21] énoncent que si l'objectif désiré de la sélection de caractéristiques est l'obtention d'une grande précision alors l'utilisation d'un classificateur est conseillée pour l'évaluation des variables. Autrement dit, l'approche *Wrapper* est préférable. Par contre si le but de la procédure de sélection est la réduction de la dimension et la suppression de variables redondantes, il vaut mieux utiliser une mesure moins coûteuse en termes de temps de calcul et donc utiliser une approche *Filter*.

4.2.5 Conclusions

Comme nous venons de le voir dans cette section, la sélection de caractéristiques est un domaine de recherche très actif, proposant un grand nombre d'algorithmes satisfaisant un

grand nombre de configurations. Au cours de cette analyse du domaine, nous avons remarqué que la plupart des algorithmes développés le sont par des scientifiques travaillant dans les domaines de l'apprentissage automatique (*Machine Learning*) et de l'analyse exploratoire de données (*Data Mining*), plus qu'en reconnaissance de formes. Cette constatation nous a menés à énoncer que pour le domaine de la reconnaissance de formes, les travaux concernant l'extraction d'information pertinente ne sont pas effectués dans la même optique. Les applications de la reconnaissance de formes étant généralement soumises à des considérations de temps réel, l'étape d'extraction doit être la plus rapide possible. D'un autre coté les applications de l'apprentissage automatique et de l'analyse exploratoire de données n'ayant pas les mêmes contraintes, la quantité d'information extraite peut être beaucoup plus importante. Il est alors naturel de développer des algorithmes permettant de sélectionner les plus pertinentes.

Une autre remarque est que pour la plupart des algorithmes de sélection de caractéristiques étudiés, les données à traiter doivent être étiquetées. Pour une application de classification, cela revient à considérer un apprentissage supervisé du système. Finalement ces algorithmes sont généralement appliqués à des problèmes de taille réduite ou moyenne, c'est-à-dire que le nombre de variables de départ est inférieur à 50. Pour un nombre supérieur il semble que peu d'algorithmes soient efficaces.

L'augmentation quasi exponentielle de la puissance des calculateurs conduit à traiter une masse de données de plus en plus importante. De ce fait, la réduction du nombre de variables utilisées dans une application peut paraître inutile. Cependant la quantité d'information disponible à traiter a augmenté de manière proportionnelle. La sélection de caractéristiques sera toujours employée de manière à supprimer les variables redondantes ou celles qui introduisent du bruit.

En conclusion, la sélection de caractéristiques reste un problème difficile à cause de sa non-monotonie. Cette propriété se traduit par le fait que le meilleur sous-ensemble de m variables ne contient pas forcément le meilleur sous-ensemble de p variables ($p < m$).

4.3 Un nouvel algorithme de sélection de primitives

Le but de notre projet est d'augmenter les performances de classification d'un système de reconnaissance de l'écriture manuscrite. L'analyse approfondie de ce dernier a permis d'orienter nos efforts vers l'amélioration de la représentation de l'information, contenue dans les images de l'écriture et utilisée par le système de reconnaissance. Pour cela nous avons proposé d'extraire de nouvelles primitives à partir de plusieurs espaces de représentation. L'étape suivante consiste à intégrer ces différentes sources d'information dans le système de reconnaissance. L'approche jusqu'alors utilisée ne convient plus, du fait qu'un trop grand nombre de primitives sont mises en jeu. Une alternative est alors de sélectionner les primitives ou caractéristiques les plus pertinentes pour la classification et donc utiliser un des algorithmes présentés à la section précédente.

Étant donné notre système de reconnaissance, deux approches sont possibles. La première consiste à prendre en compte des variables continues pour la sélection de caractéristiques. Chaque caractéristique de la sélection correspondra alors à un des axes des différentes représentations prises en compte (par exemple 64 axes pour le vecteur caractéristique DDD). La seconde approche prend en compte des variables booléennes. La procédure consiste alors à créer un ensemble de primitives pour chaque espace de représentation pris en compte et de considérer un vecteur caractéristique contenant l'ensemble des primitives. Le choix de l'algorithme de sélection à mettre en œuvre sera alors conditionné par ces considérations.

Cependant nous proposons de développer une nouvelle alternative, basée sur le concept d'amélioration de la représentation de l'information et non pas de sélection. En effet notre analyse du système a montré que parmi les primitives utilisées certaines sont très discrimi-

nantes alors que d'autres ne le sont pas. Dans le cas de la sélection de caractéristiques ces dernières seraient tout simplement éliminées. Notre approche considère qu'elles peuvent apporter tout de même une information, particulièrement dans le contexte d'une représentation discrète. En effet la caractérisation d'un graphème par une primitive implique qu'elle n'a pas les propriétés associées aux autres primitives de l'ensemble considéré. Cette considération peut être utilisée de manière à orienter la recherche d'informations complémentaires et donc guider la définition de nouvelles primitives. Une telle représentation de l'information introduira des relations hiérarchiques entre les primitives.

La suite de cette section permettra de présenter ce nouvel algorithme. Après une description générale de son fonctionnement, les différents modules le composant seront détaillés en identifiant les contraintes auxquelles ils sont soumis et en spécifiant le choix de leur mise en œuvre.

4.3.1 Présentation générale de l'algorithme de sélection et d'intégration d'information

Le principe de base de notre algorithme est d'améliorer la représentation de l'information d'un ensemble de primitives de départ de manière itérative. Nous présentons une description schématique de son fonctionnement sur la figure 43. Le point de départ est un ensemble de primitives noté F_1. Son obtention passe bien évidemment par une étape d'extraction de caractéristiques et possiblement une phase de quantification vectorielle. Cet ensemble permettra de mettre en œuvre l'étape d'apprentissage et d'obtenir un système de reconnaissance. Ce sont les performances de ce dernier que nous désirons améliorer.

À partir de l'ensemble de primitives F_1 et du système de reconnaissance associé, nous proposons d'évaluer les distributions de probabilités des classes de notre modélisation étant données les primitives. Les données obtenues permettront d'évaluer le pouvoir discriminant de chaque primitive. Cet indicateur sera alors utilisé de manière à ordonner les primitives de l'ensemble F_1. La définition dynamique d'un seuil de pouvoir discriminant τ_1

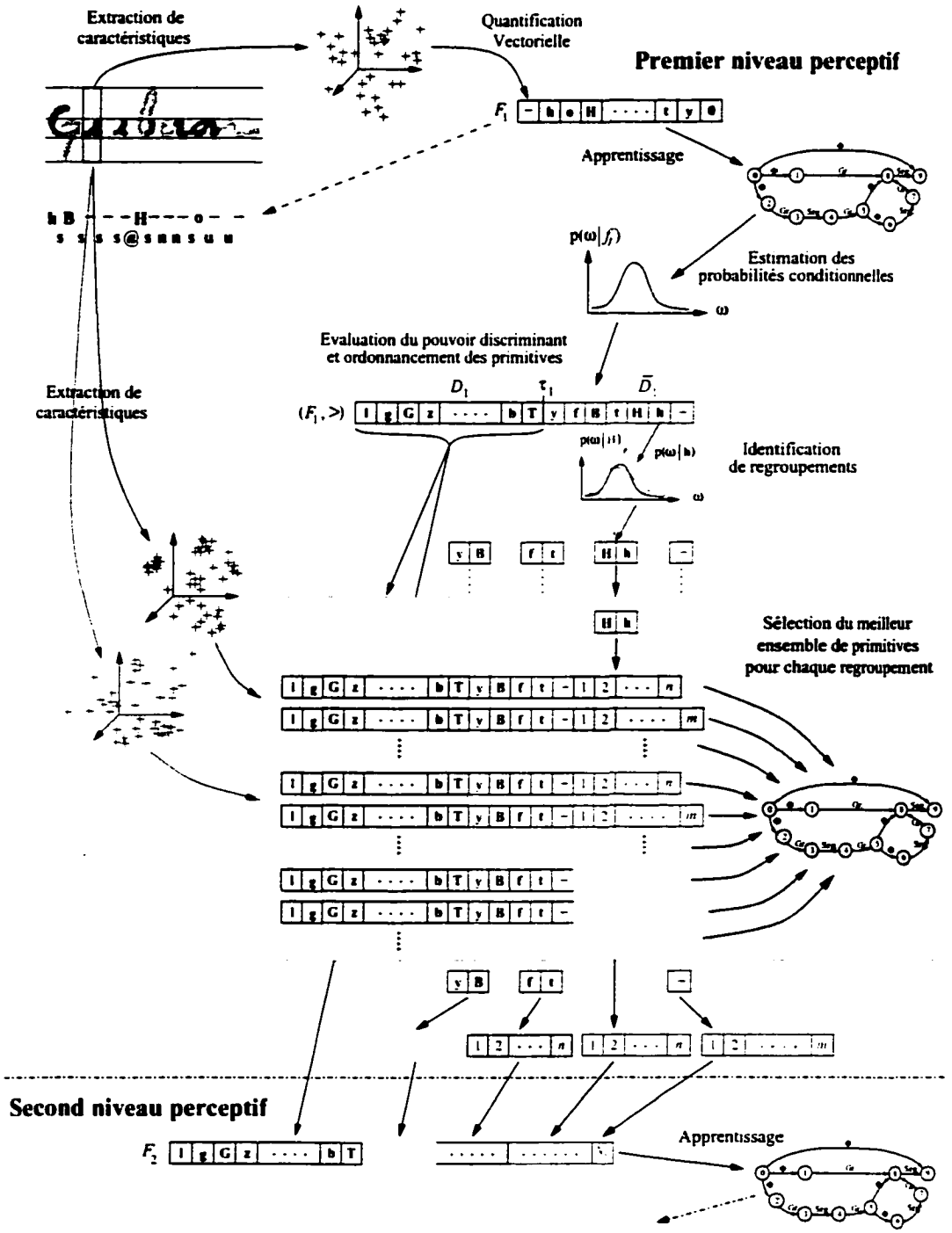


FIGURE 43 Description schématique du fonctionnement de l'algorithme de sélection et d'intégration d'information.

permettra d'isoler le sous-ensemble de primitives jugées non suffisamment discriminantes, en fonction de l'évaluation de toutes les primitives de l'ensemble. Ce sous-ensemble est noté $\bar{D}_1$ sur la figure 43. L'étape suivante consiste à identifier au sein de ce sous-ensemble des regroupements possibles de primitives. Pour cela la distribution de probabilités sur les classes de modélisation associée à chaque primitive est comparée à celle des autres primitives du sous-ensemble. Si les distributions sont similaires cela signifie que les primitives considérées caractérisent les mêmes classes de modélisation et donc les mêmes caractères. Il est donc judicieux de les regrouper. De plus, les différents regroupements peuvent être vus comme une décomposition de la tâche de classification en plusieurs sous-problèmes, puisque les primitives d'un même regroupement caractérisent seulement un sous-ensemble des classes de la modélisation.

Chaque regroupement sera alors remplacé par une nouvelle source d'information. Une phase de sélection est mise en œuvre de manière à obtenir le meilleur ensemble de primitives permettant de substituer un regroupement. À partir des espaces de représentation disponibles, plusieurs nouveaux ensembles de primitives sont construits en utilisant uniquement les données associées au regroupement considéré. Cette stratégie permet de spécialiser ces ensembles pour le sous-problème identifié par le regroupement. Chacun de ces sous-ensembles est évalué individuellement en le concaténant avec les primitives de l'ensemble de départ qui ne font pas partie du regroupement considéré. Étant donné notre objectif final, le critère de sélection utilisé est la performance du système de reconnaissance évaluée sur la base de validation et construit à partir de chaque nouvel ensemble.

Cette étape de sélection permet d'obtenir autant de nouveaux ensembles de primitives que de regroupements identifiés. La concaténation de ces ensembles avec les primitives de l'ensemble de départ jugées discriminantes (le sous-ensemble D_1 sur la figure 43) permet d'obtenir l'ensemble de primitives du second niveau perceptif F_2. Le terme *perceptif* est utilisé car l'ensemble de primitives est la seule représentation de l'information dont dispose le système, il peut être considéré comme sa perception des données. La suppression et

DESCRIPTION GLOBALE DE L'ALGORITHME DE SÉLECTION ET D'INTÉGRATION D'INFORMATION

Définir un ensemble de primitives de départ F_1
Réaliser l'apprentissage du système avec l'ensemble de primitives F_1
Évaluer la performance du système de reconnaissance
Évaluer et ordonner les primitives de l'ensemble F_1
Répéter
  Déterminer le sous-ensemble $\bar{D}_i$ des primitives non-discriminantes de F_i
  Déterminer les regroupements ou classes de primitives $C_{i,j}$ dans le sous-ensemble $\bar{D}_i$
  Choisir un nouvel ensemble de primitives $E_{i,j}$ pour substituer chaque classe $C_{i,j}$
  Regrouper les primitives pour former l'ensemble du niveau suivant F_{i+1}
  Réaliser l'apprentissage du système avec l'ensemble de primitives F_{i+1}
  Évaluer la performance du système de reconnaissance
  Évaluer et ordonner les primitives de l'ensemble F_{i+1}
Tant que le critère d'arrêt n'est pas atteint
Validation de l'ensemble de primitive final F_{Final}

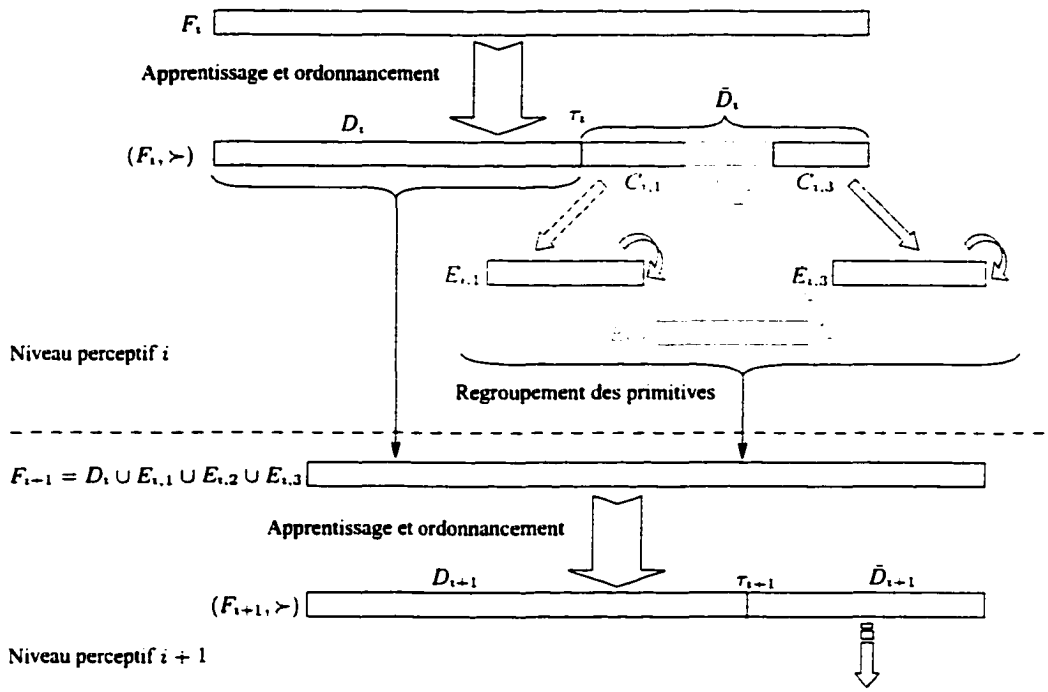


FIGURE 44 Pseudo-code et schéma de fonctionnement de notre algorithme d'intégration de plusieurs sources d'information à l'aide de différents ensembles de primitives.

l'ajout de primitives sont alors une modification de sa perception. L'ensemble F_2 permet la construction d'un système de reconnaissance. Les étapes d'évaluation et de sélection peuvent être réitérées jusqu'à l'obtention du critère d'arrêt. Ce dernier est bien sûr basé sur la performance de ce système, plus exactement sur le gain de performance apporté par la descente d'un niveau perceptif.

Le résultat final du déroulement de l'algorithme est une représentation de l'information sous la forme de plusieurs niveaux perceptifs. Cette dernière peut être vue comme une hiérarchie de primitives, où celles du premier conditionnent les suivantes. De plus l'information contenue dans les primitives des niveaux supérieurs est intégrée implicitement dans celles des niveaux inférieurs, puisque cette information est utilisée dans la définition des nouvelles primitives.

Nous présentons sur la figure 44 le pseudo-code ainsi qu'une représentation graphique de la procédure d'intégration d'information, du point de vue des primitives. En effet sur cette figure, chaque rectangle correspond à une ou plusieurs primitives. Le cas de l'identification de plusieurs regroupements de primitives a été représenté. La notation utilisée sur cette figure est celle que nous allons adopter pour le reste de notre description.

La suite de cette section est consacrée à la description des différentes étapes de notre algorithme. Pour chacune d'elles nous allons présenter les contraintes qui y sont associées, puis la solution choisie ainsi que les justifications correspondantes.

4.3.2 L'ensemble de primitives de départ

Le point de départ de notre algorithme est un ensemble de primitives, c'est-à-dire une première représentation de l'information présente dans l'image ou encore la première perception de l'image qu'a le système. Actuellement notre système de reconnaissance utilise deux ensembles de primitives conjointement, ce qui conduit à considérer un total de $27 \times 14 = 378$ primitives. L'analyse de leur fréquence d'occurrence a montré que certaines

n'apparaissent que très rarement. Un point à signaler est que notre algorithme conduira à une augmentation du nombre de primitives utilisées par le système. Celle-ci s'accompagne d'un accroissement de la complexité du système. La contrepartie est qu'il faut utiliser un nombre d'échantillons d'apprentissage plus important, de manière à conserver une certaine fiabilité dans l'estimation des paramètres. De ce fait nous proposons d'utiliser un ensemble de primitives de cardinalité réduite comme point de départ.

L'approche développée consiste à remplacer une ou plusieurs primitives par un groupe de nouvelles primitives afin d'accroître la qualité de la représentation de l'information. Il semble alors judicieux d'utiliser une première représentation de la forme assez globale puis au fur et à mesure des itérations chercher une information plus locale et/ou contextuelle.

Certaines de nos analyses montrent que l'écriture cursive est plus difficile à reconnaître. En fait le plus gros problème est de définir des primitives permettant de bien caractériser tous les types d'écriture. Il serait alors fort intéressant de disposer au premier niveau de notre représentation d'un ensemble de primitives permettant de distinguer les différents types d'écriture. Cette information permettraient de guider la définition des différents ensembles de primitives des niveaux suivants.

4.3.3 Apprentissage du système de reconnaissance

Une fois l'ensemble de primitives de départ choisi il est utilisé afin de construire un premier système de reconnaissance. La procédure utilisée est celle décrite dans la section 2.3.5. Pour sa mise en œuvre, deux corpus de données sont nécessaires, un d'apprentissage et un de validation. Nous utilisons ceux décrits dans la section 2.3.8 et dans le tableau I. Cette étape n'est pas spécifique à notre algorithme d'intégration d'information, elle ne sera donc pas plus détaillée ici.

4.3.4 Évaluation de la performance du système

Afin d'évaluer l'ensemble de primitives de chaque niveau perceptif il faut disposer d'un indicateur. Notre but final étant d'améliorer les performances du système, nous allons les utiliser afin d'évaluer les différents ensembles de primitives. Après une description des qualités que doit avoir un tel indicateur, nous poursuivrons en présentant notre choix ainsi que les justifications associées.

4.3.4.1 Définition et contraintes de l'indicateur de performance

L'indicateur de la performance d'un système de reconnaissance doit permettre d'obtenir une information quant à son pouvoir de généralisation. C'est-à-dire qu'il doit mesurer la faculté du système, après apprentissage, à classifier des données inconnues. Une autre qualité recherchée pour cet indicateur est sa faculté à permettre la comparaison de la qualité de différents systèmes de reconnaissance.

Globalement, la comparaison de systèmes de reconnaissance n'est pas une chose aisée. En effet, ces derniers font intervenir un nombre de facteurs important : la modélisation utilisée, le nombre de paramètres, les données d'apprentissage utilisées,.... Cependant il n'est pas raisonnable de vouloir comparer des systèmes de reconnaissance trop hétérogènes. Concernant notre algorithme, les bases de données ainsi que la modélisation seront identiques à tous les systèmes construits. La différence majeure viendra du nombre de primitives utilisées. Ceci se traduit au niveau des systèmes par un nombre de paramètres différents. En effet le fait d'ajouter ou de supprimer des primitives à l'ensemble de départ conduit à modifier le nombre d'observations possibles et donc le nombre des probabilités a_{ijk} (voir section 1.1.2). L'indicateur de performance doit être indépendant de ce paramètre. Dans l'absolu, un indicateur de performance ne devrait être dépendant que du pouvoir de généralisation du système testé.

4.3.4.2 Choix d'un indicateur de performance

La meilleure estimation des performances d'un système est l'évaluation des taux de reconnaissance sur un corpus de données différent de celui ayant servi à son apprentissage. Cette technique permet d'évaluer efficacement le pouvoir de généralisation du système de reconnaissance. Le corpus de test (voir tableau I) de notre base de données a été construit dans ce but.

L'algorithme que nous présentons est de type itératif. Ceci implique que l'étape d'évaluation sera réalisée plusieurs fois. Cette dernière ayant une influence dans le déroulement de l'algorithme, cela revient à prendre en compte d'une certaine manière les données utilisées dans la construction du système de reconnaissance final. L'utilisation du corpus de test pour cette évaluation interdira son utilisation pour la validation finale.

Nous proposons de conserver le corpus de test pour cette étape finale et d'utiliser celui de validation afin d'évaluer la performance du système lors des différentes itérations. Cette alternative est possible car ce dernier n'est pas utilisé pour l'estimation des paramètres du système, mais juste pour évaluer la qualité du modèle au cours de l'apprentissage et assurer une certaine généralisation.

4.3.4.2.1 Les taux de reconnaissance

L'évaluation des performances d'un système utilisant une modélisation markovienne se fait généralement par l'intermédiaire de l'algorithme de Viterbi 1.1.3.2. La complexité algorithmique de ce dernier est $O(N^2T)$, où N est le nombre d'états du graphe associé au mot testé et T la longueur de la séquence d'observations. Dans la section 2.3.7 nous présentons le protocole utilisé pour l'évaluation des performances d'un système. Différentes tailles de lexique sont utilisées durant cette étape de manière à mieux évaluer l'influence des modifications apportées au système. L'évaluation des taux de reconnaissance nécessite alors l'application de l'algorithme de Viterbi autant de fois qu'il y a d'entrées dans

le lexique utilisé. Dans le cadre de notre algorithme, nous proposons d'utiliser 1 000 entrées dans le lexique pour effectuer l'évaluation de la performance de notre système. Ce choix est un compromis entre la sensibilité de l'évaluation et le temps de calcul nécessaire. En effet l'application répétée de l'algorithme de Viterbi va demander un temps de calcul non-négligeable.

Afin d'évaluer les performances de notre système sans avoir à recourir à la phase de test et avec pour motivation la réduction du temps de calcul, nous avons développé un indicateur de la performance d'un système après apprentissage [59]. Sa définition est basée sur cette observation : à la fin de l'apprentissage la probabilité d'observation ou vraisemblance de la base de validation donne une information pertinente sur le pouvoir de généralisation du système.

4.3.4.2.2 Vraisemblance du corpus de validation

Au cours de chaque itération de la procédure d'apprentissage, la probabilité d'observation ou vraisemblance du corpus de validation est calculée à l'aide de la procédure *Forward* (voir section 1.1.3.1). Cette dernière permet d'obtenir la probabilité qu'une séquence d'observation $\mathcal{O} = o_0 o_1 \cdots o_{T-1}$ soit produite par un modèle Λ. Notons que sa complexité algorithmique est de $O(N^2T)$, identique à celle de l'algorithme de Viterbi.

Les équations 1.11 et 1.12, décrivant cette procédure, montrent clairement que la probabilité d'observation $\Pr(\mathcal{O}|\Lambda)$ est un produit des probabilités de transition $a_{ijo_{t-1}}$ et $a'_{ij\Phi}$. De ce fait $\Pr(\mathcal{O}|\Lambda)$ est fortement dépendante de la longueur T de la séquence d'observations $\mathcal{O}$, ainsi que de la valeur moyenne des probabilités de transition. Pendant l'apprentissage, cette probabilité est calculée pour chaque séquence d'observations $\mathcal{O}^i$ du corpus de validation $\mathbf{O}_{Val}$, à l'aide de l'étiquette du nom de ville correspondant w^i. Elle peut donc s'écrire : $\Pr(\mathcal{O}^i|w^i, \Lambda)$. Le logarithme de la vraisemblance du corpus de validation s'exprime alors

de la manière suivante :

$$\mathcal{L}(\mathbf{O}_{Val}, \Lambda) = \frac{1}{Q} \sum_{i=1}^{Q} \log \Pr(\mathcal{O}^i | w^i, \Lambda) \tag{4.1}$$

où Q est le nombre d'échantillons dans le corpus de validation. La vraisemblance peut être interprétée comme la probabilité moyenne d'observation d'une séquence du corpus. Elle est estimée à chaque itération de l'apprentissage et sa maximisation garantit la faculté de généralisation du système résultant. Notre remarque précédente permet de conclure que la quantité $\mathcal{L}(\mathbf{O}_{Val}, \Lambda)$ est dépendante des valeurs moyennes des probabilités de transitions $a_{ijo_{t-1}}$ et $a'_{ij\Phi}$ du modèle.

4.3.4.2.3 Définition d'un indicateur du pouvoir généralisation des modèles

De par ses propriétés, il semble judicieux d'utiliser la quantité $\mathcal{L}(\mathbf{O}_{Val}, \Lambda)$ obtenue en fin d'apprentissage afin d'évaluer et comparer différents systèmes. Le meilleur système doit obtenir la plus forte vraisemblance. Ceci est possible et correct car les données utilisées pour évaluer cette quantité sont différentes de celles utilisées pour entraîner le système. De ce fait la probabilité de vraisemblance est indépendante de l'apprentissage et permet réellement de tester le pouvoir de généralisation du système après apprentissage.

Cependant, elle est fortement dépendante de M la taille de l'ensemble des observations possibles. En effet pour une transition donnée $i \rightarrow j$ émettant un symbole, sans information *a priori*, la probabilité moyenne peut être exprimée par :

$$a^{moy}_{ijo_{t-1}} = \frac{1}{M} \tag{4.2}$$

De ce fait la valeur de $\Pr(\mathbf{O}_{Val}|\Lambda)$ ne peut pas être comparée directement entre différents systèmes ne possédant pas le même nombre de paramètres.

Nous proposons de définir un indicateur, basé sur la vraisemblance, permettant de comparer différents systèmes. Pour une base de données fixe et différents systèmes, la seule constante dans l'expression $\Pr(\mathcal{O}^i|w^i,\Lambda)$ est l'étiquette w^i de l'échantillon. Cette remarque est à la base de la définition de notre indicateur. L'évaluation de la probabilité *a posteriori* de l'étiquetage w^i étant donnés la séquence d'observations $\mathcal{O}^i$ et le modèle Λ : $\Pr(w^i|\mathcal{O}^i,\Lambda)$ est plus pertinente pour comparer différents systèmes. Cette probabilité *a posteriori* est reliée à la probabilité d'observation d'une séquence par la règle de Bayes :

$$\Pr(w^i|\mathcal{O}^i,\Lambda) = \frac{\Pr(\mathcal{O}^i|w^i,\Lambda) \times \Pr(w^i)}{\Pr(\mathcal{O}^i)} \tag{4.3}$$

où $\Pr(w^i)$ est la probabilité *a priori* de l'étiquette aussi appelé modèle de langage. Cette quantité est difficile à estimer quelle que soit l'application. De plus elle est constante quel que soit le système utilisé et donc n'influence en aucune manière la comparaison de différents systèmes. Par conséquent elle peut être négligée. Le dénominateur de l'équation 4.3 est la probabilité *a priori* de la séquence d'observations. Cette quantité est directement reliée au nombre de primitives M considérées dans l'application. Lors de l'estimation de différents systèmes et pour une séquence d'observations donnée $\mathcal{O}^i$, $\Pr(\mathcal{O}^i)$ sera différente mais elle correspondra à une même référence (ou borne) pour les différents systèmes. Cette quantité peut être considérée comme un facteur de normalisation.

Nous proposons alors d'utiliser comme indicateur du pouvoir de généralisation d'un système entraîné Λ, la moyenne sur le corpus de validation, de la probabilité *a posteriori* de l'étiquetage modifié. Plus formellement nous définissons notre indicateur du pouvoir de généralisation d'un modèle $I_P(\Lambda)$ de la manière suivante :

$$I_P(\Lambda) = \frac{1}{Q}\sum_{i=1}^{Q} \log \frac{\Pr(\mathcal{O}^i|w^i,\Lambda)}{\Pr(\mathcal{O}^i)} \tag{4.4}$$

Les probabilités $\Pr(\mathcal{O}^i|w^i, \Lambda)$ sont disponibles à chaque itération de l'algorithme d'apprentissage. En effet elles sont estimées par l'intermédiaire de l'algorithme *Forward* afin d'évaluer la vraisemblance du corpus de validation. Les probabilités des séquences d'observations $\Pr(\mathcal{O}^i)$ sont quant à elles totalement indépendantes de l'apprentissage et peuvent être calculées une seule fois avant le début de l'apprentissage. La valeur de cet indicateur est disponible directement à la fin de la phase d'apprentissage. Il permet ainsi d'économiser le temps de calcul associé à l'évaluation de la performance du système.

Deux possibilités sont envisageables pour l'estimation des probabilités $\Pr(\mathcal{O}^i)$, suivant l'hypothèse d'indépendance des observations posée. Si nous considérons les observations d'une même séquence comme indépendantes, il suffit d'estimer les probabilités d'observation de chacune d'elles, en les considérant équiprobables par exemple ou en évaluant leur fréquence sur un corpus de données. Le cas de dépendance entre les observations, plus réaliste étant donnée notre application, conduit à estimer la probabilité d'une séquence à l'aide de n-grams. Leur estimation nécessite une base de données conséquente. En effet il est fréquent qu'un nombre non-négligeable de n-grams ne soit jamais observé sur le corpus d'évaluation, ce qui conduit à des probabilités nulles dans le modèle résultant. Afin de résoudre ce problème, une technique de lissage des probabilités doit être employée.

L'indicateur du pouvoir de généralisation est en fait le rapport de la probabilité *a posteriori* de la séquence d'observations $\mathcal{O}^i$ sur la probabilité *a priori* de cette séquence. Logiquement ce rapport doit être supérieur à 1, car l'apprentissage du système doit conduire à une meilleure probabilité de la séquence d'observation. Dans le cas contraire, cela signifierait que l'apprentissage est inutile et que la probabilité *a priori* suffit pour effectuer la reconnaissance. La valeur normale de notre indicateur à la fin de la phase d'apprentissage est donc strictement positive. Plus elle est élevée, meilleur est le pouvoir de généralisation du système.

4.3.4.2.4 Une technique de lissage pour pallier le manque de données

Les n-grams sont utilisés dans le domaine de la reconnaissance de la parole, particulièrement pour mettre en œuvre le modèle de langage [76]. Nous avons étudié ce domaine afin de choisir une technique de lissage. Notre choix s'est arrêté sur le lissage de Katz [78] utilisant un modèle *Back-off*. Cette technique est jugée comme satisfaisante par la communauté scientifique de la reconnaissance de la parole et est l'une des plus utilisées. De plus elle est simple et rapide de mise en œuvre.

Cette technique étend l'intuition du lissage de Good/Turing [55] en combinant des modèles d'ordre inférieur. Dans l'approche de Good/Turing l'idée de base est la suivante : pour tout n-gram apparu r fois, on prétend qu'il est apparu r^* fois :

$$r^* = (r+1)\frac{n_{r+1}}{n_r} \tag{4.5}$$

où n_r est le nombre de n-grams apparu exactement r fois dans le corpus utilisé. Tout n-gram η est alors estimé à l'aide de ces comptes modifiés :

$$\Pr_{GT}(\eta) = \frac{r^*}{N'} \quad \text{avec} \quad N' = \sum_{r=0}^{\infty} r^* n_r \tag{4.6}$$

N' est alors le nombre total de comptes modifiés. Il est bien égal au compte N de départ :

$$N' = \sum_{r=0}^{\infty} n_r r^* = \sum_{r=0}^{\infty} n_r (r+1)\frac{n_{r+1}}{n_r} = \sum_{r=0}^{\infty} n_{r+1}(r+1) \tag{4.7}$$

$$N' = \sum_{r=1}^{\infty} n_r r = N$$

Le lissage de Good/Turing comporte certaines restrictions. En particulier pour le n-gram le plus fréquent, apparaissant r fois, la quantité n_{r+1} n'existe pas, cela conduit à lui attribuer une fréquence nulle. De plus cette technique ne peut pas être appliquée si un des comptes

n_r est nul. En pratique le lissage n'est effectué que pour les n-grams apparaissant peu de fois, 10 par exemple.

Katz propose une extension de cette approche en introduisant lors de l'estimation des modèles n-grams, une combinaison de modèles d'ordre inférieur : $n-1$,.... C'est de cette considération que l'approche prends son nom de *Back-off Model*, qui peut se traduit en français par méthode de repli.

L'idée intuitive est d'effectuer un décompte des n-grams apparaissant dans le corpus afin de le redistribuer sur ceux non rencontrés. Cette étape s'effectue en fonction de la distribution du niveau inférieur $n-1$-grams. Cela se traduit dans le cas des bigrammes par :

$$\Pr_K(o_i|o_{i-1}) = \begin{cases} \hat{\Pr}(o_i|o_{i-1}) & \text{si } |o_{i-1}^{i}| > 0 \\ \alpha_K(o_{i-1})\Pr(o_i) & \text{sinon} \end{cases} \tag{4.8}$$

où $|o_{i-1}^{i}|$ est le nombre d'occurrences de la séquence d'observations $o_{i-1}o_i$ dans le corpus utilisé. La généralisation de cette formule est alors la suivante :

$$\begin{aligned} \Pr_K(o_i|o_{i-n+1}^{i-1}) &= \hat{\Pr}(o_i|o_{i-n+1}^{i-1}) \\ &+ \theta_K\left(\Pr(o_i|o_{i-n+1}^{i-1})\right)\ \alpha_K\left(\Pr(o_i|o_{i-n+1}^{i-1})\right)\ \Pr_K(o_i|o_{i-n+2}^{i-1}) \end{aligned} \tag{4.9}$$

avec

$$\theta_K(x) = \begin{cases} 1 & \text{si } x = 0 \\ 0 & \text{sinon} \end{cases}$$

La quantité $\hat{\Pr}$ caractérise le décompte des n-grams rencontrés dans le corpus. La fonction α_K permet de redistribuer ce décompte sur les n-grams n'apparaissant pas. Katz propose d'effectuer ce décompte par l'intermédiaire d'un facteur de décompte d_r, fonction du nombre de leurs occurrences r. De plus il propose de n'effectuer le décompte que pour des valeurs de r inférieures à un seuil k, avec $5 \leq k \leq 8$. Le but de cette technique est de ne

pas décompter les n-grams dont le nombre d'occurrences permet d'obtenir une estimation fiable. Ce facteur de décompte d_r doit être obtenu de manière à satisfaire deux contraintes. Premièrement les décomptes doivent être proportionnels à ceux obtenus par l'approche de Good/Turing :

$$1 - d_r = \mu \left(1 - \frac{r^*}{r}\right) \tag{4.10}$$

Deuxièmement le nombre total d'occurrences décomptées de la distribution globale doit être égal au nombre total d'occurrences que l'approche de Good/Turing assigne aux n-grams n'apparaissant pas. Cela se traduit par la relation suivante :

$$\sum_{r=1}^{k} n_r \, (1 - d_r) \, r \; = \; n_1 \tag{4.11}$$

La solution permettant de satisfaire à ces deux contraintes est la suivante :

$$d_r \; = \; \frac{\frac{r^*}{r} - \frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}} \tag{4.12}$$

Il faut alors que la masse de probabilité décomptée d'un côté soit égale à celle distribuée de l'autre. C'est-à-dire que :

$$\sum_{o_i} \Pr_K(o_i|o_{i-1}) \; = \; 1 \tag{4.13}$$

$$\sum_{o_i:|o_{i-1}^i|>0} \hat{\Pr}(o_i|o_{i-1}) + \sum_{o_i:|o_{i-1}^i|=0} \alpha_K(o_{i-1}) \Pr(o_i) = 1 \tag{4.14}$$

Ces considérations conduisent alors à :

$$\alpha_K(o_{i-1}) \; = \; \frac{1 - \sum_{o_i:|o_{i-1}^i|>0} \hat{\Pr}(o_i|o_{i-1})}{\sum_{o_i:|o_{i-1}^i|=0} \Pr(o_i)} \; = \; \frac{1 - \sum_{o_i:|o_{i-1}^i|>0} \hat{\Pr}(o_i|o_{i-1})}{1 - \sum_{o_i:|o_{i-1}^i|>0} \Pr(o_i)} \tag{4.15}$$

Le lissage de Katz pour les n-grams d'ordre supérieur est alors obtenu à partir du modèle d'ordre $n-1$.

Nous venons de détailler l'étape d'évaluation des performances d'un système de reconnaissance. Cette dernière est réalisée dans le but d'obtenir un indicateur de la qualité de l'ensemble de primitives utilisés pour construire ce système. Elle correspond à l'étape d'évaluation du processus de sélection de primitives représenté à la figure 42. Deux approches ont été proposées pour la mise en œuvre de cet indicateur. L'utilisation des taux de reconnaissance garantit une bonne évaluation de l'ensemble. Il faut évaluer expérimentalement le nouvel indicateur que nous avons proposé.

4.3.5 Évaluation individuelle des primitives

Notre algorithme nécessite également une évaluation individuelle des primitives afin de les ordonner. Dans ce but, nous proposons d'utiliser un indicateur de leur pouvoir discriminant. Ce dernier permet de quantifier la capacité qu'une primitive à pour associer un graphème à une classe de la modélisation mise en œuvre. En pratique nous allons utiliser le même indicateur que pour l'évaluation des deux ensembles de primitives du système standard présenté dans la section 2.4.5, à savoir la perplexité conditionnelle des classes étant données les primitives. Les différentes justifications de l'utilisation de cet indicateur ainsi que sa mise en œuvre ont été présentées dans la section 2.4.5.

Cet indicateur permet d'obtenir une évaluation individuelle ainsi que globale des primitives d'un ensemble. Cependant une des propriétés de l'entropie est d'être croissante avec l'augmentation du nombre de partition de l'espace de probabilité associé à la variable aléatoire considérée. En d'autres termes, l'augmentation du nombre de primitives d'un ensemble de départ conduira obligatoirement à une diminution de l'entropie associée. De ce fait seuls deux ensembles de primitives de même cardinalité peuvent être comparés directement avec cet indicateur. Dans les autres cas il nous renseigne uniquement sur la

quantité d'information apportée par les primitives concernant les classes de la modélisation. En conclusion, la perplexité conditionnelle peut être utilisée afin de caractériser le pouvoir discriminant des primitives individuellement, mais ne peut pas réellement être utilisé afin de comparer deux ensembles de primitives différents.

4.3.5.1 Ordonnancement d'un ensemble de primitives

Une fois que les valeurs de perplexité conditionnelle ont été calculées pour chaque primitive de l'ensemble évalué, ce dernier peut être ordonné. L'ordonnancement consiste simplement à appliquer une des relations d'ordre naturel de l'ensemble des réels, à savoir $\leq$ et $\geq$, suivant l'indice du pouvoir discriminant utilisé. Nous avons choisi arbitrairement de les ranger de la plus discriminante vers la moins discriminante. La notation d'un ensemble de primitives ordonné est alors la suivante : $(F, \succ)$. Elle est utilisée sur la figure 44. L'expression $f_i \succ f_j$ signifie que la primitive f_i est plus discriminante que la primitive f_j.

La mesure de qualité des primitives utilisée est la perplexité conditionnelle. Plus la valeur de perplexité associée à une primitive est proche de 1, plus cette dernière est discriminante. De ce fait nous allons utiliser la relation d'ordre naturel $\leq$ afin d'ordonner les différents ensembles de primitives à l'aide de cet indicateur.

4.3.6 Identification des primitives non-discriminantes

Une fois l'ensemble de primitives ordonné, il faut déterminer celles qui ne sont pas suffisamment discriminantes. Pour cela nous proposons de définir un seuil dans l'échelle d'évaluation des primitives. Nous allons maintenant détailler cette étape.

4.3.6.1 Définition et contrainte d'un seuil de pouvoir discriminant

Le seuil de pouvoir discriminant, noté τ_i sur la figure 44, va permettre de scinder l'ensemble de primitives F_i en deux sous-ensembles D_i et $\bar{D}_i$, où D_i contiendra les primi-

tives discriminantes alors que $\bar{D}_i$ celles que ne le sont pas suffisamment. Ces deux sous-ensembles sont complémentaires et bien sûr ordonnés :

$$F_i = D_i \cup \bar{D}_i$$
$$D_i \cap \bar{D}_i = \emptyset$$
$$D_i = \{f_1, f_2, \ldots, f_i\} \quad \text{et} \quad \bar{D}_i = \{f_{i+1}, f_{i+2}, \ldots f_M\}$$
$$f_1 \succ f_2 \succ \ldots \succ f_i \succ f_{i+1} \succ \ldots \succ f_{N-1} \succ f_N$$

La définition du seuil de pouvoir discriminant τ_i n'est pas directe. Il faut prendre en compte certaines considérations. Premièrement l'utilisation d'un seuil fixe n'est pas envisageable. En effet l'indicateur de pouvoir discriminant utilisé est dépendant du nombre de primitives dans l'ensemble. De ce fait l'évaluation d'une primitive est relative à l'ensemble de primitives global uniquement. Le choix de la valeur du seuil doit donc être dynamique au cours des différentes itérations de l'algorithme, afin de prendre en compte le fait que la taille de l'ensemble de primitives augmentera.

La détermination de la valeur du seuil doit considérer une référence pour chaque ensemble évalué. Plusieurs possibilités sont envisageables. La première consiste à prendre comme référence une estimation globale de l'ensemble de primitives. Cette dernière doit bien sûr être directement liée aux estimations individuelles des primitives. Une technique générique permettant de caractériser un ensemble de valeurs est d'évaluer sa moyenne et son écart type. Ceci revient à considérer que toutes les primitives qui n'ont pas un pouvoir discriminant supérieur à la valeur moyenne ne sont pas assez discriminantes. Cette proposition induit qu'un nombre important de primitives, de part la définition de la moyenne, sera remplacé à chaque itération.

Une seconde possibilité est de prendre comme référence la ou les valeurs extrêmes de pouvoir discriminant de l'ensemble. L'utilisation d'une heuristique permettra de déterminer la

valeur numérique du seuil τ_t. Par exemple, les primitives qui ont un pouvoir discriminant supérieur à 75% de celui associé à la primitive la moins discriminante. Cette approche conduit à utiliser une heuristique ; sa valeur doit être déterminer, arbitrairement ou expérimentalement.

Une autre alternative consiste à combiner les deux approches et considérer la moyenne ainsi que la valeur maximum de l'ensemble lors de la définition du seuil τ_t. Ce dernier doit permettre d'isoler un nombre raisonnable de primitives. Si l'utilisation de la valeur globale conduit à un nombre trop important de primitives l'indicateur du maximum sera utilisé. Cette possibilité implique la définition d'un nombre maximum de primitives pouvant être substituées à un niveau donné. Ce dernier peut facilement être déterminé comme un pourcentage du nombre de primitives contenues dans l'ensemble F_t. Cependant cette éventualité ajoute une heuristique à notre algorithme.

4.3.6.2 Choix du seuil de pouvoir discriminant

L'indicateur de pouvoir discriminant que nous utilisons est la perplexité conditionnelle. Il permet d'effectuer une évaluation globale de l'ensemble de primitives à partir des valeurs individuelles et des fréquences d'occurrence des primitives (voir équation 1.66). En fait l'évaluation globale est une moyenne pondérée des valeurs de perplexité individuelle par les fréquences d'occurrence des primitives. Elle permet de prendre en compte dans l'évaluation globale le fait qu'une primitive fréquente a une plus grande influence sur les performances du système.

Dans un premier temps nous pensions utiliser directement cette valeur comme seuil de pouvoir discriminant. La prise en compte de la pondération lors de son calcul conduit à un comportement différent d'une moyenne classique. En effet une primitive très fréquente avec un pouvoir discriminant faible influencera fortement cet indicateur global et réduire le nombre de primitives du sous-ensemble $\bar{D}_t$. Nous pouvons constater ce phénomène

par l'intermédiaire du tableau V où seulement quatre primitives perceptuelles sont jugées non-discriminantes.

Cette constatation, ainsi que diverses expériences, nous a menés à conclure qu'il faut prendre en compte la nature de l'ensemble de primitives pour la définition du seuil τ_i. En effet, il existe une différence entre un ensemble de primitives résultant d'une quantification vectorielle d'un autre ensemble explicitement défini, comme l'ensemble perceptuel du système standard. La quantification vectorielle consiste à diviser un espace de représentation en M cellules sous une contrainte de distorsion minimale. Si les échantillons d'apprentissage sont uniformément distribués, chaque cellule contiendra un nombre d'éléments similaires, ce qui se traduit par des fréquences des primitives proches de la valeur $1/M$. Dans le cas de primitives explicitement définies, les fréquences peuvent être beaucoup plus hétérogènes, comme le prouve le cas de l'ensemble perceptuel. De ce fait il est justifié de procéder différemment pour la détermination du seuil en fonction de la nature de l'ensemble de primitives.

Un autre paramètre pour la détermination du seuil τ_i a été identifié : le niveau perceptif en cours. En effet il est facile de définir arbitrairement une valeur de ce seuil au premier niveau perceptif où toutes les primitives proviennent du même espace de représentation. Par la suite cela devient beaucoup plus difficile.

Suite à ces observations et différentes expérimentations, nous proposons deux stratégies pour la détermination du seuil τ_i. Si un ensemble de primitives explicitement défini est utilisé au premier niveau, le seuil sera déterminé arbitrairement après analyse des différentes valeurs de pouvoir discriminant. Dans les autres cas le seuil sera déterminé à l'aide des valeurs résultantes de l'estimation du pouvoir discriminant des primitives de l'ensemble considéré. Dans un premier temps nous voulions utiliser l'évaluation globale de l'ensemble. Cependant expérimentalement nous nous sommes rendu compte que cette stratégie conduisait à considérer beaucoup de primitives comme non-discriminantes, en

particulier lors de l'utilisation au premier niveau d'un ensemble issu d'une quantification vectorielle. Ceci implique une augmentation rapide du nombre de primitives total, ce qui n'est pas le but recherché. En effet l'obtention d'un trop grand nombre de primitives va conduire à diminuer la fiabilité de l'estimation des paramètres de nos modèles. Il est donc préférable de limiter le nombre de primitives.

Nous avons alors choisi de définir un seuil du pouvoir discriminant moins contraignant. En plus de la valeur de perplexité globale de l'ensemble de primitives, la valeur maximale a été prise en compte. Le seuil de perplexité d'un niveau perceptif donné i est alors défini comme la moitié de la somme de la perplexité globale de l'ensemble de primitives plus la perplexité de la primitive la moins discriminante :

$$\tau_i = \frac{2^{H(C|F_i)} + 2^{\max_j H(C|f_j)}}{2} \tag{4.16}$$

Le choix du seuil de pouvoir discriminant n'est pas une tâche aisée comme nous venons de le constater. L'approche proposée est arbitraire est peut certainement être améliorée.

4.3.7 Identification de regroupements de primitives

Une fois le sous-ensemble des primitives non-discriminantes $\bar{D}_i$ identifié, notre algorithme se propose de les remplacer par de nouvelles. Il faut alors construire de nouveaux ensembles de primitives. Dans l'absolu, chaque primitive de $\bar{D}_i$ pourrait être remplacée par un nouvel ensemble. Cette hypothèse conduit à une augmentation rapide du nombre total de primitives, ce que nous désirons éviter. De ce fait il est intéressant de trouver une technique pour effectuer des regroupements de primitives au sein de l'ensemble $\bar{D}_i$ et de cette manière réduire et limiter le nombre de primitives ajoutées à chaque itération de l'algorithme.

4.3.7.1 Définition des regroupements de primitives

La définition de regroupements de primitives nécessite la détermination d'un nouveau critère. Une primitive est la représentation symbolique d'une combinaison de caractéristiques extraites des graphèmes. Deux primitives distinctes représentent alors deux combinaisons différentes de ces caractéristiques de base. Cependant il est possible que ces combinaisons diffèrent en partie seulement et donc que deux primitives partagent un certain nombre de caractéristiques communes. Cette possibilité permet d'envisager le regroupement de certaines primitives en fonction de leurs points communs.

Un *regroupement de primitives* $C_{i,j}$ d'un ensemble F_i est défini comme un sous-ensemble de F_i dont tous les éléments f_k possèdent au moins une propriété commune. Cette propriété peut être vue comme une contrainte dans l'espace de représentation des primitives permettant de définir un sous-espace. Prenons comme exemple l'espace de représentation des primitives perceptuelles, un regroupement de primitives peut être défini comme le sous-ensemble des primitives contenant un dépassement haut. Cet espace de représentation est de type qualitatif et comporte un nombre d'axes réduit. Un de ces axes est celui des dépassements hauts, il comporte trois modalités : pas de dépassement haut, un petit dépassement haut et un grand dépassement haut. La définition d'un regroupement de primitives revient à considérer le sous-espace réduit par la contrainte : la modalité des primitives sur l'axe des dépassements hauts doit être petit ou grand. Cela permet de définir un sous-ensemble de primitives de l'ensemble global F_i. Le regroupement de primitives sera alors composé de l'intersection de ce sous-ensemble avec celui des primitives non-discriminantes $\bar{D}_i$.

Nous venons de présenter un exemple dans le cas d'un espace de représentation discret. Dans le cas continu la définition de regroupements de primitives est identique. L'espace continu associé à l'exemple présenté dans le paragraphe précédent consiste à évaluer la taille du dépassement (en pixels par exemple). Une valeur x_{depH} sera alors attribuée à

chaque graphème. À ce moment le regroupement de primitives défini ci-dessus sera caractérisé par la contrainte $x_{depH} > 0$ ou plus exactement $x_{depH} > \sigma_H$, où σ_H est un seuil au dessus duquel un dépassement est jugé significatif. Cette contrainte permet de la même manière que dans le cas discret de réduire l'espace de représentation.

Dans les deux cas présentés ci-dessus, nous avons restreint l'espace de représentation. Le regroupement de primitives est alors le sous-ensemble de primitives appartenant à ce sous-espace de représentation. L'étape suivante consiste à réaliser l'intersection entre ce sous-ensemble et celui contenant les primitives non-discriminantes, afin de définir les regroupements à faire dans $\bar{D}_i$.

4.3.7.2 Choix d'un critère de regroupement

La définition de regroupements de primitives présentée ci-dessus fait appel à l'espace de représentation des primitives. Elle est facilement interprétable dans le cas d'espaces de représentation de type qualitatif, où chacun des axes est associé à une caractéristique structurelle des graphèmes. Dans le cas d'espaces de représentation numérique, où il n'est pas facile d'associer à chaque axe une propriété particulière, comme celui des concavités, il est alors moins facile de définir de tels regroupements. Afin de pallier ce problème, nous proposons de ne plus considérer l'espace de représentation des primitives. Nous allons prendre en compte le positionnement des primitives dans l'espace des classes de la modélisation, par l'intermédiaire de leurs distributions de probabilités conditionnelles étant données les primitives : $\Pr(\omega_i|f_j)$. Si les distributions associées à deux primitives distinctes sont semblables, cela signifie qu'elles servent à caractériser des graphèmes de formes similaires. Il est donc judicieux de regrouper des primitives ayant des distributions de probabilités semblables.

De manière à effectuer ces regroupements, nous devons disposer d'un indicateur permettant de mesurer la similitude entre les distributions de probabilités de deux primitives. Dans un premier temps nous pensions utiliser la distance de *Kullback-Leibler*, encore ap-

pelée divergence ou entropie relative [26]. Elle est définie de la manière suivante :

$$KL(P\|Q) = \sum_{S} P(x_i) \log \frac{P(x_i)}{Q(x_i)} \tag{4.17}$$

où P et Q sont deux distributions de probabilité sur un même alphabet S. Plus la valeur de cet indicateur est grande, plus les distributions sont différentes. Son minimum est zéro, il est atteint lorsque les deux distributions sont identiques. L'application de cette distance pour comparer la distribution de deux primitives f_1 et f_2 s'écrira :

$$KL(f_1\|f_2) = \sum_{i=1}^{N_C} \Pr(c_i|f_1) \log \frac{\Pr(c_i|f_1)}{\Pr(c_i|f_2)} \tag{4.18}$$

Cependant cette mesure n'est pas réellement une distance car elle n'est pas symétrique et ne satisfait pas l'inégalité du triangle. De plus elle n'est définie que si les distributions sont non-nulles sur l'alphabet S. Ce dernier point pose un problème dans le cas de notre application. En effet la plupart des primitives ne sont pas rencontrées par toutes les classes de la modélisation. De ce fait certaines probabilités seront nulles, ce qui pose un problème lors du calcul de la distance de *Kullback-Leibler*.

Une autre mesure de distance est donc nécessaire. Notre choix s'est arrêté sur la norme L_1. Elle est obtenue à partir d'une classe de métrique générale appelée la métrique de Minkowski ou encore la norme L_k [31]. Pour deux vecteurs a et b de dimension d, elle est définie de la manière suivante :

$$L_k(a,b) = \left(\sum_{i=1}^{d} |a_i - b_i|^k \right)^{1/k} \tag{4.19}$$

La norme L_2 correspond à la distance euclidienne. La norme L_1 entre deux fonctions de densité de probabilité P et Q est obtenue de la manière suivante :

$$L_1(P,Q) = \|P-Q\|_1 = \sum_{i=1}^{d} |P(i)-Q(i)| \tag{4.20}$$

Plus particulièrement pour l'application que nous désirons mettre en œuvre, la mesure de distance entre les distributions de probabilité conditionnelle associées à deux primitives f_1 et f_2 s'écrira :

$$L_1(f_1,f_2) = \sum_{i=1}^{N_C} |\Pr(c_i|f_1) - \Pr(c_i|f_2)| \tag{4.21}$$

Cette métrique est appropriée à la mesure que nous désirons effectuer. La comparaison de fonctions de densité de probabilités implique que les valeurs numériques seront comprises entre 0 et 1. L'utilisation de la norme L_1 plutôt que L_2 permettra de donner un plus grand poids aux grandes différences entre les deux distributions. Cet avantage est particulièrement intéressant pour notre application. En effet, si la première primitive est utilisée majoritairement pour caractériser une classe et que la seconde ne l'est jamais, nous désirons que la mesure de distance les sépare au maximum.

Un avantage de l'utilisation de la norme L_1 est la relative simplicité de son interprétation. Sa valeur sera comprise entre 0 et 2, le maximum correspondant à la somme des deux distributions de probabilités. Une valeur de 1 correspondra donc à un recouvrement de 50% des deux distributions.

Il faut maintenant définir une valeur en dessous de laquelle deux primitives seront considérées comme suffisamment proches pour être regroupées. Nous avons décidé de fixer ce seuil à 1. Cette valeur peut sembler élevée, mais elle nous semble justifiée du fait de la grande variabilité intrinsèque à l'écriture et de la non-optimalité des différents pré-

traitements. Ces deux points conduisent à observer une grande variabilité de formes associées à une classe donnée.

4.3.7.3 Méthode proposée pour le regroupement des primitives

Une fois défini le critère permettant de regrouper les primitives, il faut effectuer les regroupements. Il permettra d'obtenir pour chaque primitive la distance qui la sépare des autres. Une technique de classification, comme la quantification vectorielle, peut permettre d'effectuer les regroupements. Cependant elle nécessite la définition *a priori* du nombre de ces regroupements. De plus le nombre d'échantillons, c'est-à-dire de primitives, est faible pour la mise en œuvre d'une approche statistique.

Cette alternative n'étant pas satisfaisante, notre choix a été orienté vers une approche basée sur la théorie des graphes. Nous proposons d'effectuer une partition du sous-ensemble des primitives non-discriminantes par la technique des cliques maximales. Une clique est un sous graphe d'un graphe non-orienté pour lequel tous les nœuds sont connectés. À partir de l'ensemble de primitives $\bar{D}_i$ un graphe sera construit. Une connexion entre deux primitives sera réalisée si la valeur de la norme L_1 est inférieure à 1. La recherche de la clique maximale permettra d'obtenir le plus gros regroupement de primitives respectant la contrainte de distance.

Afin de mettre en œuvre cette technique, l'algorithme de *Bron-Kerbosh* [15] a été utilisé. C'est un algorithme itératif efficace qui permet de trouver la clique maximale d'un graphe. Dans notre implémentation, nous utilisons récursivement cet algorithme afin d'obtenir une partition de l'ensemble de primitives. Pour cela l'algorithme est appliqué une première fois sur l'ensemble des primitives, puis celles faisant partie de la clique maximale sont supprimées et l'algorithme de *Bron-Kerbosh* est appliqué de nouveau. Cette étape est réitérée jusqu'à ce qu'il ne reste plus de primitives dans l'ensemble de départ. Cette technique permet d'obtenir une partition de notre ensemble de primitives.

4.3.7.4 Interprétations des regroupements des primitives

La définition d'un regroupement revient à imposer une contrainte dans l'espace de représentation. Lorsqu'une quantification vectorielle a été utilisée, un regroupement de primitives revient à regrouper différentes cellules dans l'espace de représentation. Cela peut être alors considéré comme une étape de post-traitement de la quantification vectorielle. Une autre interprétation est que les regroupements conduisent à diviser la tâche de reconnaissance en plusieurs sous-problèmes. En effet chaque regroupement permet de ne considérer qu'un sous-espace de l'espace de représentation global. La technique employée pour former les regroupements garantit que les formes présentent dans ce sous-espace sont associées par le système à un même sous-ensemble de classes de la modélisation. Nous allons alors retourner aux formes de manière à effectuer de nouvelles mesures et donc créer un nouvel espace de représentation associé au regroupement. Dans ce dernier uniquement un sous-ensemble de classes de la modélisation sera présent. Cette remarque permet effectivement d'énoncer le fait que la prise en compte de regroupements conduit à diviser la tâche globale de reconnaissance en différents sous-problèmes. Une difficulté consiste alors à choisir chaque nouvel espace de représentation de manière à optimiser la séparation des différentes classes.

Une remarque concernant le sous-espace associé à un regroupement est qu'il n'est pas forcément continu. En effet la technique de regroupement de primitives ne garantit pas que les cellules associées aux primitives du regroupement sont voisines.

Nous venons de décrire l'étape permettant d'effectuer des regroupements de primitives parmi le sous-ensemble des non-discriminantes $\bar{D}_i$. C'est une étape cruciale et innovante de l'algorithme que nous proposons. Le fait de prendre en compte les distributions de probabilités entre les primitives et les classes peut être vu comme l'intégration d'une information discriminante, puisque la définition de nouvelles primitives est guidée par l'information apportée par les primitives du niveau supérieur sur les classes de la modélisation.

4.3.8 Construction et choix des nouveaux ensembles de primitives

Une fois les regroupements de primitives identifiés, notre algorithme se propose de les remplacer par de nouvelles primitives. Pour chaque regroupement un nouvel espace de représentation doit être choisi. Ce dernier permettra alors la construction d'un ensemble de primitives spécifique au sous-problème caractérisé par le regroupement. À ce moment la difficulté identifiée est le choix du meilleur ensemble de primitives permettant la substitution de chaque regroupement.

Globalement notre approche consiste à disposer au début de l'application de l'algorithme d'un certain nombre d'espaces de représentation. Pour chaque regroupement identifié à un niveau perceptif donné, chaque espace disponible sera testé. En effet, il peut être facile de préjuger de la pertinence d'un espace de représentation pour substituer un regroupement, lorsque les primitives sont définies explicitement comme notre ensemble de primitives perceptuelles (considérons le regroupement caractérisant la présence d'un dépassement haut dans les graphèmes, la définition d'un nouvel ensemble peut être fonction de cette information). Cela est beaucoup plus difficile, voire impossible, lorsqu'un ensemble issu d'une quantification vectorielle est utilisé. De plus nous désirons que l'algorithme fonctionne automatiquement. Ces considérations nous ont conduits à adopter l'approche consistant à tester tous les espaces de représentation disponibles.

4.3.8.1 Construction des nouveaux ensembles de primitives

Pour chaque regroupement et à partir de chaque espace de représentation, un ensemble de primitives doit être construit. Pour cela, deux cas doivent être pris en compte, suivant la nature de l'espace de représentation considéré :

- il est de type discret ou qualitatif comme celui des primitives perceptuelles, alors un nombre de primitives fini a été défini.

- il est de type continu comme celui des concavités. Il faut alors avoir recours à une phase de quantification vectorielle afin de le discrétiser.

Dans la première alternative, il est possible que la prise en compte de regroupements réduise considérablement la fréquence de certaines primitives de l'ensemble global associé à l'espace de représentation. En effet, ce dernier est généralement défini à partir d'observations réalisées sur la totalité des échantillons d'apprentissage. La prise en compte de regroupements à un niveau donné revient à ne considérer que le sous-ensemble d'échantillons situé dans le sous-espace associé au regroupement considéré. Ceci implique que certaines primitives d'un ensemble, associées à un espace de représentation discret, risquent de ne pas être observées sur le corpus d'apprentissage. Cependant il est préférable de les conserver car nous ne pouvons pas être sûrs qu'elles ne le seront pas sur d'autres corpus.

Lorsqu'une quantification vectorielle est nécessaire, se pose alors un autre problème : le choix du nombre de primitives ou nombre de cellules à créer. En effet il est difficile de déterminer *a priori* ce paramètre. L'approche que nous préconisons est expérimentale. L'algorithme de quantification utilisé, LBG (voir section 1.3.3), conduit à la création d'ensemble contenant un nombre de primitives égal à une puissance de 2 et limite donc le nombre d'ensembles de primitives potentiels. Il est alors envisageable de les évaluer tous. Certaines considérations permettront tout de même de limiter le nombre d'ensembles de primitives à construire. Il s'agit premièrement du nombre d'échantillons regroupés par une classe de primitives. En effet la quantification est une approche statistique, il faut donc un nombre d'échantillons minimal afin d'obtenir des résultats fiables. Un second paramètre sera également pris en compte. Il s'agit du nombre de classes de modélisation caractérisées par les primitives du regroupement considéré. Cette donnée est obtenue au cours de l'évaluation individuelle des primitives du niveau supérieur. Plus le nombre de classes est important plus il est justifié de créer un ensemble de primitives de grande taille.

Afin de limiter le nombre d'ensembles de primitives créés à partir d'un espace de représentation continu ces deux paramètres ont été pris en compte dans la définition d'une heuristique. En fait nous utilisons les pourcentages de classes de modélisation et d'échantillons associés au regroupement j considéré. Sa définition est la suivante :

$$\eta_j = 1000 \times \frac{NbClasRgrp_j}{NbClasTotal} \times \frac{NbEchRgrp_j}{NbEchTotal} \tag{4.22}$$

Le facteur numérique utilisé a été choisi arbitrairement. Sa valeur peut être interprétée comme le nombre maximum de primitives autorisées dans le cas de l'utilisation de l'ensemble des échantillons (expérimentalement nous avons remarqué que l'utilisation d'un si grand nombre de primitives détériore les performances du système). Le calcul de cette heuristique permet d'obtenir une estimation du nombre maximal de centres de gravité à rechercher lors de la quantification vectorielle. Pour cette dernière le nombre de primitives doit être une puissance de 2. L'exposant correspondant est alors obtenu en calculant :

$$\mathfrak{N}_{max} = (ent)(\log_2 \eta_j + 0,5) \tag{4.23}$$

où (ent) est la partie entière. Afin de limiter le temps de calcul, nous proposons de ne pas construire plus de cinq ensembles de primitives à partir d'un espace de représentation. Étant donné que le nombre de primitives doit être un exposant de deux, la différence entre la taille maximale et minimale de l'ensemble sera tout de même importante. Afin d'obtenir l'exposant correspondant il suffit de soustraire 5 à la valeur obtenue ci-dessus :

$$\mathfrak{N}_{min} = \max(1, \mathfrak{N}_{max} - 5) \tag{4.24}$$

La fonction max est utilisée afin d'obtenir un ensemble d'au moins deux primitives. En effet il n'y a aucun intérêt à remplacer un regroupement par une seule primitive.

Concernant la construction des ensembles de primitives, l'utilisation d'un algorithme de quantification vectorielle capable de déterminer le nombre de partitions permettrait de simplifier cette étape. De tels algorithmes procèdent généralement par une phase de division de l'espace de représentation puis par leur agglomération suivant divers critères. Cette alternative nécessite l'ajustement de différents paramètres afin d'obtenir une solution acceptable. De plus, la répétition des deux étapes entraîne un temps de calcul important.

4.3.8.2 Choix de l'ensemble de primitives pour substituer un regroupement

Maintenant que différents ensembles de primitives sont disponibles pour remplacer un regroupement, il faut choisir le meilleur. Nous proposons d'utiliser le même critère de sélection que pour l'ensemble de primitives d'un niveau perceptif, à savoir l'évaluation des performances du système de reconnaissance. Ce choix implique la construction de plusieurs systèmes. Lors de l'identification de plusieurs regroupements de primitives deux possibilités sont envisageables : évaluation individuelle ou combinée des ensembles de primitives associées aux différents regroupements. Nous avons rapidement écarté la seconde alternative du fait du nombre important de combinaisons à évaluer. Considérons le cas où N regroupements ont été identifiés et que M espaces de représentation différents sont disponibles. Si un seul ensemble de primitives est construit à partir de chaque espace, le nombre de combinaison est de M^N. Pour des valeurs relativement petites de M et N, le nombre de combinaisons devient vite très élevé ($N = 5, M = 4 \rightarrow 1024$ combinaisons possibles). Bien que cette alternative permette de prendre en compte l'interaction entre les différentes primitives lors de leur évaluation, elle est difficilement envisageable.

Nous proposons donc d'utiliser une évaluation individuelle. Cette approche permet de réduire la quantité M^N à $M \times N$. Chaque regroupement sera considéré séparément. Chaque ensemble de primitives potentiel conduira à la construction d'un système de reconnaissance. Pour cela les primitives du niveau perceptif supérieur sont utilisées, mises à part celles du regroupement en cours qui sont remplacées par le nouvel ensemble. Les per-

formances du système construit à partir de cet ensemble de primitives seront mesurées et serviront d'évaluation de l'ensemble de primitives candidat. Celui obtenant la meilleure performance sera utilisé pour substituer le regroupement. Cette approche revient à effectuer la descente d'un niveau perceptif en ne considérant qu'un seul regroupement. Les étapes de construction et d'évaluation du système sont identiques à celles décrites auparavant dans cette section.

Cette approche ne permet pas de prendre en compte l'interaction entre les différentes primitives mises en jeu, mais réduit le nombre de combinaisons possibles. Elle va tout de même nécessiter un temps de calcul important puisqu'il faut construire et évaluer un certain nombre de systèmes de reconnaissance.

4.3.9 Regroupement des différentes sources d'information

Cette étape est directe. Elle consiste uniquement à former l'ensemble de primitives F_{i+1} en regroupant les primitives discriminantes du niveau supérieur, le sous-ensemble D_i, avec les différents ensembles de primitives créés à ce niveau $E_{i,j}$. Cette étape est représentée graphiquement sur la figure 44. À ce moment nous disposons de l'ensemble de primitives du niveau perceptif suivant. Les étapes d'apprentissage, d'évaluation des performances du système et du pouvoir discriminant des primitives sont identiques à celles décrites dans les sous-sections 4.3.3, 4.3.4 et 4.3.5.

4.3.10 Critère d'arrêt

L'algorithme décrit dans cette section est de type itératif. L'intégration de nouvelles sources d'information sera réalisée par l'intermédiaire de plusieurs niveaux perceptifs. Un critère d'arrêt est donc nécessaire afin de terminer le processus d'intégration. Le but final étant d'augmenter les performances d'un système de reconnaissance, ce critère doit dépendre du gain de performances résultant de la descente d'un niveau perceptif.

Afin de caractériser le gain de performance entre deux itérations, nous allons utiliser la différence relative de l'indicateur de performance. À une itération donnée, le gain $\delta_P(i)$ est défini en fonction de l'indicateur de performance I_P de la manière suivante :

$$\delta_P(i) = \frac{I_P(i) - I_P(i-1)}{I_P(i) + I_P(i-1)} \tag{4.25}$$

Si il n'est plus jugé significatif, le déroulement de l'algorithme est arrêté. Cette condition se traduit par la relation suivante :

$$\delta_P(i) < \varepsilon \tag{4.26}$$

où ε est un seuil à définir. L'indicateur de performance utilisé peut être un des deux présentés dans la section 4.3.4. La formulation du gain le rend indépendant de l'indicateur utilisé. La valeur du seuil ε a été fixé à 10^{-2} arbitrairement. En dessous de ce seuil, l'amélioration des performances du système n'est plus vraiment significative.

L'augmentation du nombre de primitives à chaque itération conduit à un accroissement du nombre de paramètres du système. Il peut être pertinent de prendre en compte ce dernier dans la définition du critère d'arrêt. En effet un trop grand nombre de paramètres conduit à une diminution de la fiabilité de leur estimation, dans le cas de l'utilisation d'une base de donnée de taille fixe. Il est alors peut être préférable d'obtenir une performance plus faible mais avec une meilleure fiabilité des paramètres. Cette combinaison peut se traduire par un meilleur pouvoir de généralisation du modèle final. Nous ne nous sommes pas consacrés à cette tâche par manque de temps, mais elle fait partie des perspectives de ce travail.

4.3.11 Validation du système de reconnaissance

Une fois le critère d'arrêt atteint, nous disposons d'un ensemble de primitives potentiellement plus discriminant que celui de départ et de taille plus importante. L'étape de validation consistera alors à évaluer la performance du système construit à partir de cet ensemble.

Pour cela des données non vues par le système seront utilisées, celles de notre corpus de test.

À ce niveau il est certain que ce sont les taux de reconnaissance du système qui nous intéressent. Cependant l'autre indicateur de performance proposé peut être utilisé également. Il peut servir à comparer deux applications de notre algorithme avec des paramètres différents, comme l'ensemble de primitives de départ où encore les différents espaces de représentation possibles lors de la définition des nouveaux ensembles de primitives.

4.3.12 Discussion

Nous venons de décrire les différentes étapes de notre algorithme. Ce dernier permet d'intégrer plusieurs sources d'information dans un système de reconnaissance dans le but de rendre plus discriminante la représentation de l'information dont il dispose. Notre approche, bien que réalisant une sélection de caractéristiques, diffère des techniques classiques de ce domaine. Pour un algorithme de cette catégorie, toutes les primitives à évaluer sont disponibles au début de son déroulement. Son but est alors de sélectionner le sous-ensemble de primitives le plus pertinent pour la tâche à réaliser. C'est une approche statique dans le sens où les caractéristiques sont fixées au départ.

Par opposition notre approche est dynamique. En effet les primitives potentielles sont définies au cours des différentes itérations, en fonction d'évaluations réalisées sur une première représentation de l'information. L'approche proposée pour la sélection de caractéristiques est innovante. Elle permet d'obtenir une représentation de l'information hiérarchisée, à l'aide de différents ensembles de primitives. Nous avons qualifié les différents niveaux de représentation de perceptifs. La raison est qu'ils caractérisent effectivement la seule perception des formes dont le système de reconnaissance dispose. Une relation hiérarchique réelle existe entre les différents niveaux perceptifs puisqu'une primitive conditionnera la définition de celles situées plus bas dans la hiérarchie. Ces dernières quant

à elles, intégreront implicitement l'information des primitives situées plus haut dans la décomposition.

Une exploitation de cette représentation hiérarchique consiste à adopter une représentation globale de la forme aux niveaux supérieurs, puis de choisir des primitives permettant d'extraire une information plus locale et contextuelle aux niveaux inférieurs. Une des remarques mentionnées au cours de notre analyse des performances du système de base est que les exemples cursifs sont plus difficiles à reconnaître que les bâtons, surtout en utilisant un même ensemble de primitives. Une solution envisageable est d'utiliser au premier niveau perceptif un ensemble permettant de séparer ces deux types d'écriture puis ensuite d'utiliser des primitives adaptées à chaque style.

Nous avons proposé le concept de regroupements ou classes de primitives. C'est également un point innovant de notre approche. Le but principal de ces regroupements est de limiter le nombre de primitives introduites à chaque niveau. Cependant ils permettent également d'effectuer une analyse supplémentaire concernant les primitives. Pour cela les distributions de probabilités des primitives sur les classes de la modélisation sont utilisées. Cette approche permet de prendre en compte une composante discriminante supplémentaire, puisque la définition de nouvelles primitives est réalisée en intégrant l'information mesurée sur les classes du niveau supérieur.

Une fois l'application de notre algorithme réalisée, nous disposons d'une représentation hiérarchique de l'information. Son obtention requiert un temps de calcul important puisqu'elle nécessite la construction et l'évaluation d'un grand nombre de systèmes de reconnaissance. Du point de vue exploitation, cette approche accélérera plutôt le processus de reconnaissance. Pour chaque graphème d'un échantillon à traiter, l'information du premier niveau perceptif sera extraite par le système. Si la primitive résultante fait partie des non-discriminantes alors seulement une nouvelle extraction est réalisée. Dans le cas

contraire on passe au graphème suivant. Cette approche minimise ainsi le temps consacré à l'extraction d'information.

Cet algorithme a été développé dans l'optique d'une application particulière. À notre avis il est possible d'étendre les concepts introduits et d'obtenir un formalisme plus général. Les principaux concepts sont la prise en compte de regroupements et l'aspect dynamique de la sélection de caractéristiques, qui conduisent à une représentation hiérarchisée de l'information.

4.4 Résumé

Dans ce chapitre, nous avons traité du domaine de la sélection de caractéristiques. Dans un premier temps une revue du domaine a été présentée. Elle a permis de mettre en évidence l'activité de ce domaine de recherche, particulièrement dans les communautés de l'apprentissage automatique et de l'analyse exploratoire des données. Après avoir exposé les différentes composantes nécessaires à un algorithme de sélection de caractéristiques, les alternatives possibles pour leurs mises en œuvre ont été présentées. Un certain nombre d'algorithmes a alors été décrit en fonction de la combinaisons de composantes utilisées.

La seconde section a permis de présenter un nouvel algorithme permettant d'effectuer une sélection de caractéristiques. Cependant l'approche proposée est beaucoup plus dynamique que les approches classiques, du fait que les caractéristiques sont définies au cours du processus de sélection. L'algorithme proposé utilise une représentation hiérarchisée de l'information en niveaux perceptifs. Un nouveau concept a alors été introduit, celui de regroupements ou classes de primitives. À un niveau perceptif donné, il permet de regrouper des primitives en fonction de leurs distributions de probabilités sur les classes de la modélisation. Ceci signifie que les primitives d'un même regroupement permettent de caractériser le même sous-ensemble de classes. C'est une information discriminante qui sera alors utilisée pour la définition de nouveaux ensembles de primitives. Cette ap-

proche permet alors d'intégrer implicitement cette information dans les niveaux perceptifs suivants.

Afin de valider cet algorithme, différentes expériences ont été réalisées. Elles seront présentées dans le chapitre suivant. Une description des différentes expériences réalisées lors de l'évaluation des différents espaces de représentation mis en œuvre précédera.

CHAPITRE 5

EXPÉRIMENTATIONS

Le but final de notre projet est d'augmenter les performances d'un système de reconnaissance de l'écriture manuscrite. Après une analyse approfondie de ce dernier, nous avons conclu que sa principale faiblesse se situe au niveau de la représentation de l'information utilisée, elle n'est pas suffisamment discriminante. Afin d'y remédier, nous avons proposé d'extraire de nouvelles sources d'information sous la forme d'ensembles de primitives. Dans cette optique plusieurs espaces de représentation ont été proposés à la section 3.2. Dans l'objectif d'obtenir les primitives les plus discriminantes possibles, nous avons proposé à la section 3.3 d'utiliser différentes techniques : l'algorithme LDA, la technique de *zoning* et en association avec cette dernière une technique de pondération des zones. La première section de ce chapitre nous permettra d'évaluer l'ensemble des espaces de représentation proposés ainsi que l'apport de ces techniques sur le pouvoir discriminant des primitives.

Le système de reconnaissance standard du SRTP, point de départ de notre étude, utilise deux ensembles de primitives. La technique permettant d'intégrer ces deux sources d'information consiste à considérer le produit cartésien des deux ensembles de primitives. Cette approche ne peut pas être utilisée afin d'incorporer les nouveaux ensembles de primitives car elle conduit à une augmentation exponentielle du nombre de paramètres. De ce fait nous nous sommes intéressés à une technique permettant de sélectionner les meilleures primitives pour la tâche de reconnaissance visée. Un nouvel algorithme a été développé dans cet objectif. Il permet de choisir et d'intégrer différentes sources d'information dans le système de manière à augmenter les performances globales du système. La mise en œuvre de cet algorithme a conduit à la réalisation d'un certain nombre d'expériences, elles sont décrites dans la seconde section de ce chapitre.

La dernière section de ce chapitre permettra de présenter nos conclusions concernant les différents espaces de représentation ainsi que l'algorithme d'intégration d'information.

5.1 Évaluation des différents espaces de représentation

Dans la section 3.2 ont été présenté les trois espaces de représentation que nous avons choisi de développer. Le premier est basé sur l'extraction de concavités (CCV), le second sur les distributions de distances directionnelles (DDD) et le dernier sur l'histogramme de directions (HD) extrait à partir du contour des graphèmes. Un quatrième espace de représentation a été créé en combinant celui des concavités et celui des histogrammes de directions (CCV+HD). L'utilisation de la stratégie de *zoning* présentée à la section 3.3.3 ainsi que de la pondération des zones conduit à multiplier le nombre d'espaces de représentation. Pour chacun d'eux, une quantification vectorielle a permis de créer cinq ensembles de primitives, contenant 64, 128, 256, 512 et 1 024 primitives. Dans le but d'augmenter le pouvoir discriminant des primitives, nous avons proposé une stratégie permettant d'utiliser l'algorithme LDA. Concrètement, les cinq systèmes construits sont utilisés afin d'effectuer un étiquetage automatique de notre corpus de données. Cette information est alors utilisée par l'algorithme LDA et permet d'obtenir une matrice de transformation. Cette dernière est utilisée afin de projeter l'ensemble de nos échantillons dans l'espace de représentation transformé. Une nouvelle quantification vectorielle est alors réalisée dans ce dernier. Elle conduit à la construction d'un ensemble de primitives de cardinalité égale à celle de l'ensemble de départ. À partir des cinq premiers systèmes de reconnaissance, nous en construisons donc cinq nouveaux.

La prise en compte de ces différents paramètres conduit à la construction d'un grand nombre de systèmes de reconnaissance. L'évaluation de la performance de chacun a été réalisée en utilisant différentes tailles de lexiques et pour les différents types d'écriture considérés. La quantité de données récoltées de cette phase expérimentale est importante. Dans cette section nous allons présenter principalement les conclusions découlant

de l'analyse des résultats obtenus. La présentation est structurée en trois parties. Premièrement la comparaison des différents espaces de représentation est réalisée par l'intermédiaire des systèmes utilisant une extraction de primitives sur les rectangles englobant des graphèmes. L'influence de la prise en compte de zones lors de l'extraction est ensuite étudiée. Finalement l'impact des deux stratégies de pondération de ces zones, proposées dans la section 3.3.4, sera présentée.

5.1.1 Extraction des primitives sur le rectangle englobant des graphèmes

Pour l'ensemble des espaces de représentation, l'extraction de caractéristiques a été réalisée dans un premier temps sur le rectangle englobant des graphèmes. C'est l'approche classique d'extraction de caractéristiques. Elle est plutôt dédiée à la reconnaissance de caractères isolés, pour laquelle l'information contextuelle est moins importante que pour la reconnaissance de caractères liés. Dans le cas de notre application, il est évident que cette approche conduira à de meilleures performances pour les exemples bâtons. En effet, pour ce type d'écriture, la plupart des graphèmes contient un caractère entier, comme le prouve nos statistiques sur la segmentation présentées dans le tableau IV.

Pour chaque espace de représentation, cinq systèmes ont été construits en augmentant le nombre de centres lors de la quantification vectorielle. Ces derniers permettent alors la mise en œuvre de l'algorithme LDA et donc la construction de cinq nouveaux systèmes de reconnaissance. Notre corpus de test a été utilisé afin d'évaluer la performance de ces dix systèmes. Différentes tailles de lexiques sont utilisées durant cette phase. Cependant nous ne présentons dans le tableau VIII que les taux de reconnaissance obtenus lors de l'utilisation d'un lexique de taille 1 000. Ce dernier permet de bien caractériser les performances d'un système de reconnaissance. L'utilisation d'une taille de lexique plus grande conduit à un comportement similaire. Afin de faciliter la lecture de ces données, nous avons mis en évidence pour chaque espace de représentation la valeur maximale associée.

TABLEAU VIII

Performances des systèmes de reconnaissance utilisant l'extraction de caractéristiques sur les rectangles englobants – Évaluations réalisées avec un lexique de taille 1 000.

Nombre de primitives			64	128	256	512	1 024
CCV	Avant LDA	TR(1)	84,9%	86,9%	**88,3%**	87,9%	88,0%
		I_P	5,18	5,52	5,73	**6,32**	6,23
	Après LDA	TR(1)	86,9%	88,2%	89,0%	**90,0%**	89,1%
		I_P	5,67	6,03	6,64	**6,69**	6,62
DDD	Avant LDA	TR(1)	77,8%	82,0%	83,6%	**85,9%**	**85,9%**
		I_P	4,21	4,71	5,20	5,84	**5,94**
	Après LDA	TR(1)	90,9%	92,1%	92,8%	92,5%	**92,9%**
		I_P	7,00	7,76	7,89	**7,94**	7,81
HD	Avant LDA	TR(1)	85,9%	87,3%	88,2%	**88,5%**	88,2%
		I_P	5,45	5,87	5,97	6,03	**6,29**
	Après LDA	TR(1)	86,7%	88,3%	**89,1%**	**89,1%**	**89,1%**
		I_P	5,80	6,14	6,32	**6,77**	6,61
CCV+HD	Avant LDA	TR(1)	86,8%	88,5%	89,5%	**90,2%**	90,0%
		I_P	5,68	6,20	6,41	**6,91**	6,84
	Après LDA	TR(1)	89,5%	90,8%	**91,5%**	**91,5%**	91,1%
		I_P	6,57	6,85	6,90	**7,42**	7,34

En plus de ce taux de reconnaissance, la valeur de l'indicateur de performances I_P, défini par l'équation 4.4, est incluse dans le tableau. Lors de son évaluation, nous avons étudié la possibilité d'utiliser différents modèles pour l'estimation des probabilités $\Pr(\mathcal{O}^t)$. La présentation de l'indicateur I_P, ainsi que les choix concernant son évaluation est incluse dans [59]. La principale conclusion de ce travail est que le modèle bigrammes doit être utilisé pour l'estimation des probabilités *a priori* des séquences d'observations $\mathcal{O}^t$. L'alternative des tri-grammes n'est pas envisageable du fait du manque d'échantillons permettant l'estimation du modèle.

Les taux de reconnaissance associés aux différents systèmes sont également présentés de manière graphique sur la figure 45. Dans la suite de cette thèse, nous avons indiqué par une ligne discontinue noire la performance atteinte par le système standard, c'est-à-dire celui utilisant conjointement les ensembles de primitives perceptuelles et bâtons, dans les mêmes conditions d'évaluation.

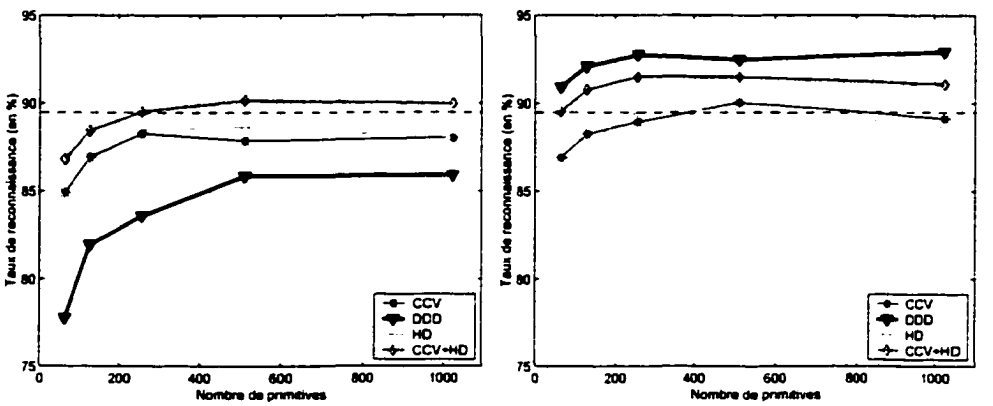
5.1.1.1 Analyse des performances globales

Plusieurs conclusions découlent de l'analyse des performances des différents systèmes. Premièrement la différence observée entre les deux graphiques de la figure 45 montre clairement que l'utilisation de l'algorithme LDA permet une augmentation du pouvoir discriminant des primitives. Avant son application, seulement les systèmes utilisant l'espace de représentation CCV+HD obtiennent une performance supérieure au système standard. L'apport est particulièrement important pour l'espace de représentation DDD. Dans tous les cas, le gain lié à l'application de l'algorithme LDA est inversement proportionnel à la taille de l'ensemble de primitives, c'est-à-dire qu'un gain plus important est observé pour les ensembles de primitives de petite taille.

Une seconde conclusion découlant de l'analyse des performances est que le nombre de primitives utilisées a une influence importante. En effet les ensembles de petites tailles ne sont pas suffisamment discriminants. Le nombre de classes mises en œuvre par notre modélisation est de 209. Il est logique qu'avec seulement 64 primitives le système de reconnaissance ne puisse pas effectuer une classification correcte de tous les graphèmes. Il est difficile de déterminer le nombre optimal de primitives associées à notre problème de classification. Si la variation intra-classes associées à notre problème était faible, l'optimal s'approcherait du nombre de classes. La spécificité de notre application est justement la grande variabilité intra-classe. De ce fait il est certain que plusieurs primitives sont nécessaires pour capter cette dernière pour certaines classes.

Nous pouvons également constater la complémentarité des espaces de représentation CCV et HD puisque leur combinaison mène toujours à des performances supérieures à leur utilisation individuelle.

Le gain en pouvoir discriminant apporté par l'algorithme LDA à l'espace de représentation DDD est important. Nous avons proposé d'utiliser ce dernier suite à la lecture de l'article [122] qui annonce un grand pouvoir discriminant de la technique d'extraction proposée.



(a) Avant LDA (b) Après LDA

FIGURE 45 Taux de reconnaissance des différents systèmes utilisant l'extraction de primitives sur les rectangles englobants.

Cependant l'auteur utilise des réseaux de neurones comme classificateur. Cette approche permet d'effectuer la discrimination nécessaire par l'intermédiaire de couches cachées. Dans notre cas, nous devons passer par une analyse discriminante de manière à obtenir des résultats similaires. Ceci explique le faible pouvoir discriminant remarqué avant LDA.

Une dernière constatation est que les espaces de représentation DDD et CCV+HD permettent, après LDA, d'obtenir des performances meilleures que celle du système standard, même en n'utilisant que 64 ou 128 primitives. Ceci traduit le bon pouvoir discriminant de ces deux espaces de représentation.

5.1.1.2 Analyse de l'indicateur de performances I_P

Dans le développement de notre algorithme d'intégration d'information, nous avons proposé l'utilisation d'un indicateur du pouvoir de généralisation du système, disponible à la fin de l'apprentissage (voir section 4.3.4.2.3). De manière à valider sa pertinence, il a été évalué à la fin de l'apprentissage de chaque système de reconnaissance construit.

L'analyse des valeurs obtenues n'est pas aisée. En effet, contrairement à nos attentes, elles ne sont pas entièrement corrélées avec les taux de reconnaissance. Le but premier de cet

indicateur est de pouvoir comparer des systèmes de reconnaissance utilisant un nombre de primitives différent. L'analyse des valeurs par espace de représentation permet de remarquer que l'indicateur I_P est souvent corrélé avec le taux de reconnaissance. Par contre la comparaison des valeurs obtenues pour deux espaces différents est difficile. Par exemple sa valeur est de 6,32 pour le système *CCV-avLDA-512* et pour le système *HD-apLDA-256* alors que les taux de reconnaissance sont respectivement de 87,87% et 89,05%. Cette remarque signifie que des systèmes de reconnaissance hétérogènes ne peuvent pas être comparés de manière fiable avec cet indicateur.

Lors de l'évaluation des performances d'un système, un tirage aléatoire est réalisé afin de choisir, dans le lexique global, les différents intitulés qui seront alignés par l'intermédiaire de l'algorithme de Viterbi, avec la séquence de primitives extraites de l'échantillon testé. D'un point de vue pratique, nous avons fixé l'initialisation du générateur de nombres aléatoires pour les différentes expériences réalisées. Cela permet de fixer le lexique utilisé pour un échantillon donné et ainsi de pouvoir mieux évaluer l'impact d'un paramètre sur notre système complexe. Cependant cette approche consiste à ne disposer que d'une estimation ponctuelle de la performance d'un système. Afin de disposer d'une évaluation plus fiable de la performance, il faudrait réaliser plusieurs fois cette étape en modifiant les lexiques utilisés, puis estimer la moyenne des performances obtenues. L'inconvénient est que cette approche nécessite un temps de calcul important. Notre but étant d'améliorer le système, en fixant les lexiques utilisés lors de l'évaluation de la performance, nous pouvons directement estimer l'impact des modifications apportées par l'intermédiaire de la performance obtenue.

Cette remarque conduit à modérer notre première conclusion sur l'indicateur I_P. En effet il est possible que l'indicateur pointe le meilleur système dans certain cas alors qu'il est en opposition avec le taux de reconnaissance, comme par exemple pour l'espace de représentation CCV avant LDA. En effet l'augmentation du nombre de primitives conduit à une diminution de la fiabilité des paramètres du système. Il semblerait plus logique

que les taux de reconnaissance décroissent de manière régulière par rapport au nombre de primitives. Dans ce cas l'indicateur I_P pointe peut être le système au meilleur pouvoir de généralisation. Pour un modèle donné permettant l'estimation des probabilités des séquences d'observations, l'indicateur I_P ne peut prendre qu'une seule valeur.

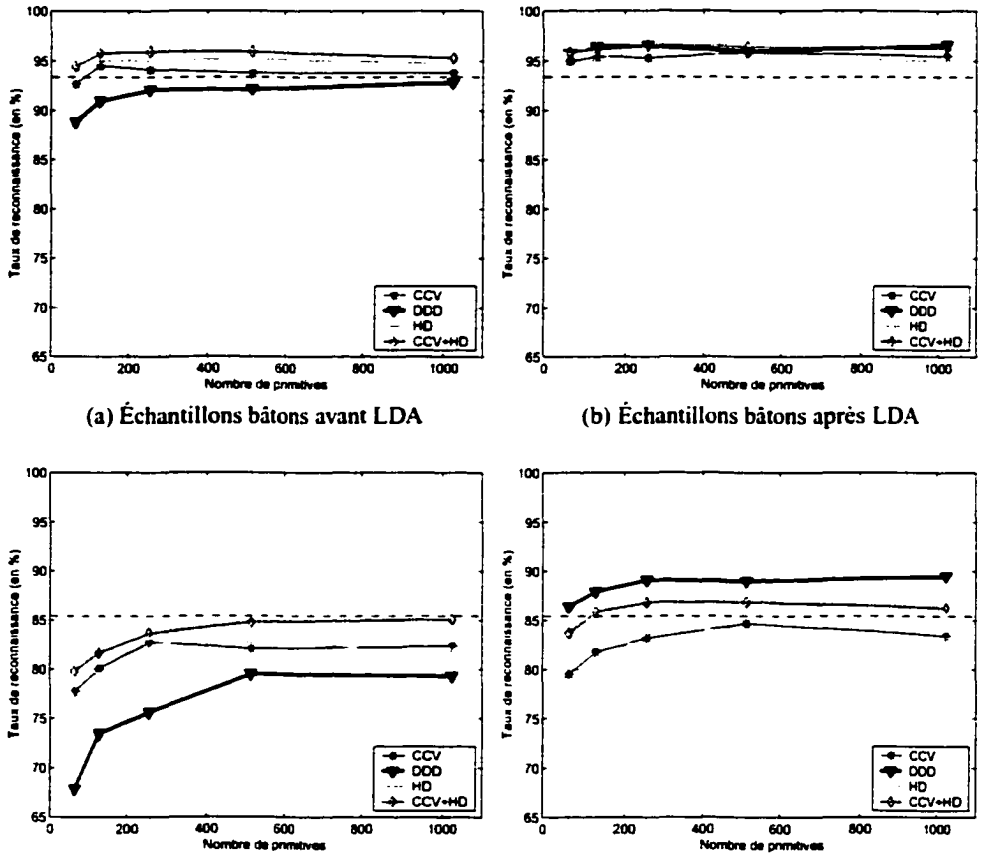
La conclusion de cette analyse est que l'indicateur proposé permet de comparer des systèmes dont la seule différence est le nombre de primitives utilisées. Dans les autres cas il n'est pas suffisamment fiable. Une étude supplémentaire est nécessaire afin de mieux évaluer son comportement. Elle n'a pas encore été réalisée par manque de temps.

5.1.1.3 Analyse des performances par type d'écriture

Afin de compléter l'étude de l'influence de différents paramètres, nous présentons sur la figure 46 les performances des différents systèmes en fonction du type d'écriture des échantillons de test. De même que sur la figure 45, la performance du système standard dans les mêmes conditions d'utilisation est représentée sur les graphiques par une ligne discontinue noire.

Une conclusion flagrante est que l'extraction de caractéristiques sur les rectangles englobants, comme prévu, permet de mieux caractériser l'écriture bâton. En effet cette approche de l'extraction ne permet pas la prise en compte d'information contextuelle locale, qui est très importante pour la reconnaissance de l'écriture cursive.

Pour l'écriture bâton, nous pouvons remarquer également que les taux de reconnaissance obtenus sont supérieurs à celui du système standard, mis à part pour l'espace DDD avant LDA. Par contre, dans le cas de l'écriture cursive, l'application de l'algorithme LDA est nécessaire pour obtenir des performances meilleures que celle du système standard. En particulier avec seulement 64 primitives DDD, nous obtenons un meilleur taux de reconnaissance que le système standard qui en utilise 378. Cette remarque confirme le bon pouvoir discriminant de cet espace de représentation.



(a) Échantillons bâtons avant LDA

(b) Échantillons bâtons après LDA

(c) Échantillons cursifs avant LDA

(d) Échantillons cursifs après LDA

FIGURE 46 Taux de reconnaissance des différents systèmes utilisant l'extraction de caractéristiques sur les rectangles englobants – Présentation en fonction du type d'écriture des échantillons.

Une autre remarque concerne l'apport de l'algorithme LDA. En effet il a une influence plus grande sur les échantillons cursifs. L'explication est que la variation intra-classe des classes associées à ce type d'écriture est plus importante. L'analyse discriminante linéaire permet de la réduire de manière considérable.

5.1.1.4 Conclusions sur l'extraction sur le rectangle englobant

La première conclusion de nos analyses est que l'utilisation de l'algorithme LDA permet bien d'augmenter le pouvoir discriminant des primitives, ce qui se traduit par une

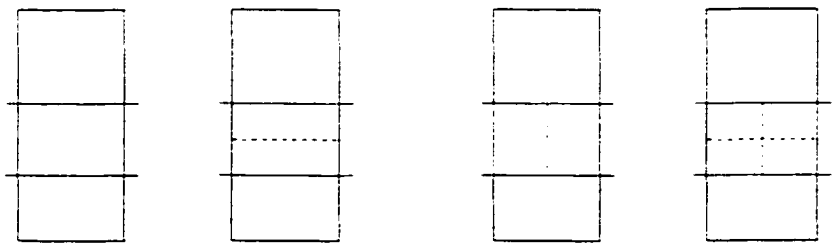
meilleure classification des graphèmes et donc une augmentation des taux de reconnaissance. Une seconde remarque est que les espaces de représentation CCV et HD permettent d'obtenir des performances similaires. L'augmentation des taux de reconnaissance remarquée lors de leur combinaison indique qu'ils sont complémentaires. Les meilleures performances sont généralement remarquées lors de l'utilisation d'un ensemble contenant 512 primitives. Finalement les taux de reconnaissance obtenus par les systèmes utilisant les espaces de représentation DDD et CCV+HD et des ensembles de primitives de petite taille montrent le bon pouvoir discriminant de ces espaces de représentation.

Notre analyse des valeurs de l'indicateur de performances I_P montre que son utilisation doit être restreinte à la comparaison de systèmes de reconnaissance construits à partir d'un même espace de représentation. Nous proposons d'effectuer une analyse plus minutieuse afin de mieux évaluer son comportement. Ce travail fait partie de nos perspectives.

5.1.2 Prise en compte des zones d'écriture

Dans la section 3.3.3, nous avons proposé de prendre en compte l'information contextuelle présente dans les graphèmes en utilisant la technique de *zoning*. Pour cela les zones d'écriture, définies au cours des pré-traitements, sont utilisées afin de découper les graphèmes. Plusieurs stratégies de division de la zone médiane ont été proposées, afin de raffiner la prise en compte de cette information contextuelle. Nous rappelons sur la figure 47 les différentes approches développées. La zone médiane, contenant le corps du mot, est celle qui contient le plus d'information. Il est donc justifié d'essayer d'en extraire une représentation plus précise.

Le tableau IX présente les performances des différentes systèmes utilisant une extraction de primitives en considérant les trois zones d'écriture. Les taux de reconnaissance indiqués sont ceux obtenus lors de l'utilisation d'un lexique de taille 1 000. Les valeurs associées de l'indicateur de performances I_P sont incluses dans ce tableau à titre indicatif uniquement. En effet, leur analyse conduit aux mêmes conclusions que celles associées à l'extraction



(a) 3 zones (b) 4 zones découpage horizontal (c) 4 zones découpage vertical (d) 6 zones

FIGURE 47 Présentation des quatre stratégies de *zoning* proposées pour l'extraction des primitives.

sur les rectangles englobants. Les figures 48, 49, 50 et 51 permettent d'évaluer l'influence de la stratégie de *zoning* sur les différents espaces de représentation.

TABLEAU IX

Performances des systèmes de reconnaissance prenant en compte les trois zones d'écriture – Évaluations réalisées avec un lexique de taille 1 000.

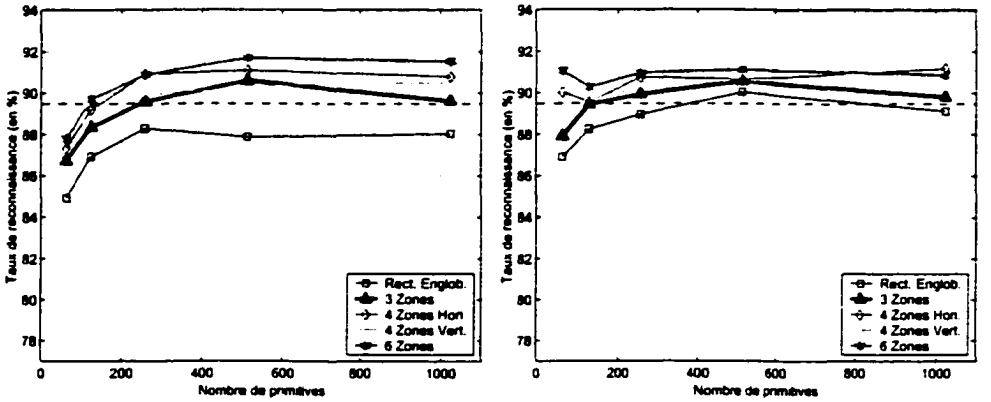
Nombre de primitives			64	128	256	512	1 024
CCV	Avant LDA	TR(1)	86,7%	88,4%	89,6%	**90,6%**	89,6%
		I_P	5,50	5,88	6,15	6,26	**6,69**
	Après LDA	TR(1)	87,9%	89,5%	89,9%	**90,6%**	89,8%
		I_P	5,83	6,13	6,80	**6,97**	6,86
DDD	Avant LDA	TR(1)	80,6%	83,4%	86,3%	**87,7%**	87,6%
		I_P	4,55	4,98	5,37	6,06	**6,11**
	Après LDA	TR(1)	91,7%	92,8%	**93,2%**	93,1%	**93,2%**
		I_P	7,28	7,49	7,61	**8,05**	8,00
HD	Avant LDA	TR(1)	87,0%	88,9%	90,5%	**90,9%**	90,6%
		I_P	5,61	6,18	6,50	6,59	**6,98**
	Après LDA	TR(1)	88,7%	89,4%	91,0%	**91,1%**	90,7%
		I_P	6,14	6,27	7,03	**7,04**	6,99
CCV+HD	Avant LDA	TR(1)	89,1%	90,9%	91,4%	91,6%	**92,0%**
		I_P	5,90	6,52	6,82	6,98	**7,27**
	Après LDA	TR(1)	91,4%	91,4%	91,7%	**91,8%**	91,5%
		I_P	6,86	6,89	7,02	**7,46**	7,33

5.1.2.1 Analyse des performances globales

L'analyse des différents graphiques montre qu'avant application de l'algorithme LDA, la prise en compte de zones permet d'augmenter significativement les performances des systèmes, par rapport à l'extraction de caractéristiques sur les rectangles englobants. Cette remarque indique que la prise en compte de la technique de *zoning* permet d'obtenir des primitives conduisant à une meilleure discrimination des formes associées à l'écriture manuscrite.

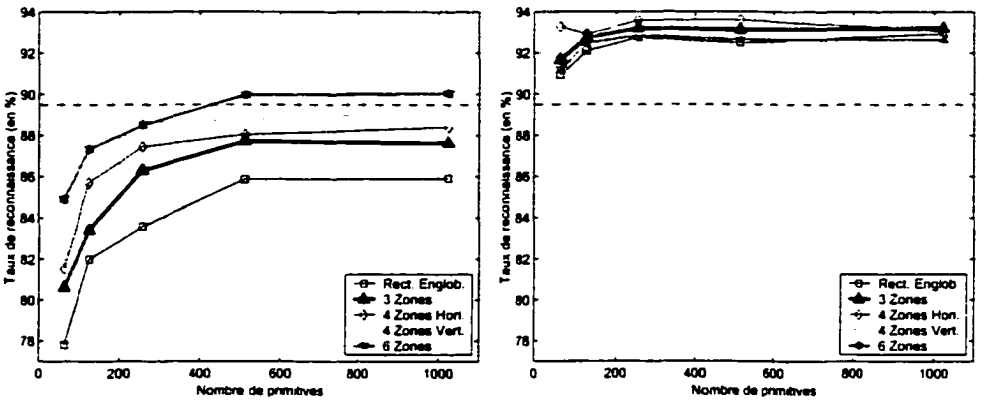
La seconde constatation est que les différentes stratégies de *zoning* ne conduisent pas aux mêmes performances. L'augmentation du nombre de zones s'accompagne d'un accroissement des taux de reconnaissance avant LDA, sauf pour l'espace de représentation HD. Ce dernier étant basé sur l'analyse du contour, l'information est moins dense. L'augmentation du nombre de zones conduit alors à la présence de plusieurs composantes nulles dans le vecteur caractéristique. Ce phénomène ainsi que l'accroissement du nombre de dimensions de l'espace considéré rendent la quantification vectorielle peu fiable. Mis à part l'espace HD, la stratégie utilisant six zones conduit aux meilleurs taux de reconnaissance avant LDA. Concernant la division de la zone médiane en deux, nous ne pouvons pas dire de l'approche verticale ou horizontale laquelle est la meilleure. Cela dépend de l'espace de représentation utilisé.

Concernant l'application de l'algorithme LDA plusieurs remarques peuvent être faites. La première est qu'elle permet toujours d'augmenter significativement les performances des systèmes utilisant l'espace de représentation DDD. Pour les autres espaces de représentation (sauf *CCV+HD-6zones*), son application n'est bénéfique que pour les ensembles de primitives de petites tailles. En effet pour ceux contenant plus de 256 primitives, nous pouvons même remarquer une diminution des taux de reconnaissance, en particulier lors de la prise en compte de six zones. Cette baisse de performances ne peut pas être imputée à la grande dimension de l'espace de représentation puisque avant l'application de LDA,



(a) Avant LDA (b) Après LDA

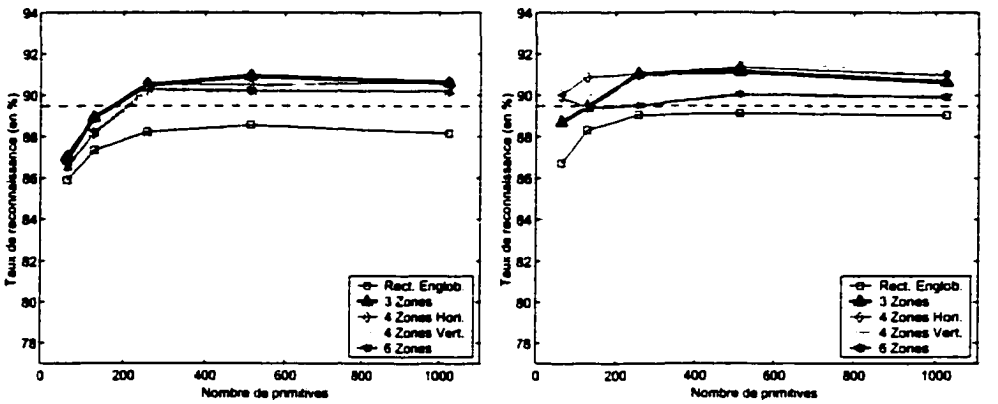
FIGURE 48 Évaluation des stratégies de *zoning* pour les systèmes utilisant des primitives issues de l'espace de représentation CCV.



(a) Avant LDA (b) Après LDA

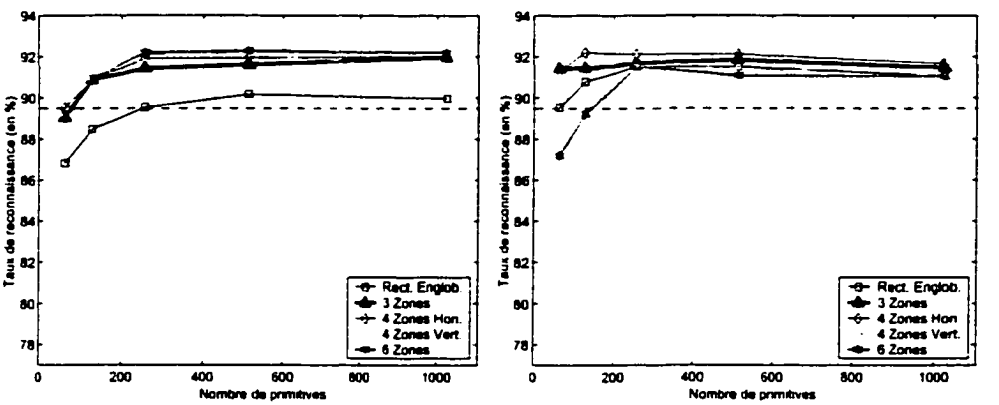
FIGURE 49 Évaluation des stratégies de *zoning* pour les systèmes utilisant des primitives issues de l'espace de représentation DDD.

l'augmentation du nombre de primitives ne conduit pas à une perte de performance. L'algorithme LDA n'est pas directement en cause non plus, puisqu'il apporte un gain pour des ensembles de petites tailles. Notre explication est que pour des espaces de grande taille, la séparation linéaire proposée par l'algorithme LDA ne parvient pas à maximiser correctement les distances inter-classes. Cela implique que lors de la quantification vecto-



(a) Avant LDA (b) Après LDA

FIGURE 50 Évaluation des stratégies de *zoning* sur les systèmes utilisant des primitives issues de l'espace de représentation HD.



(a) Avant LDA (b) Après LDA

FIGURE 51 Évaluation des stratégies de *zoning* sur les systèmes utilisant des primitives issues de l'espace de représentation CCV+HD.

rielle, avec un grand nombre de centres de gravité, certains se retrouvent positionnés à la frontière de plusieurs classes.

Après application de l'algorithme LDA, quasiment l'ensemble des systèmes de reconnaissance obtient une performance supérieure à celle du système standard. L'espace de

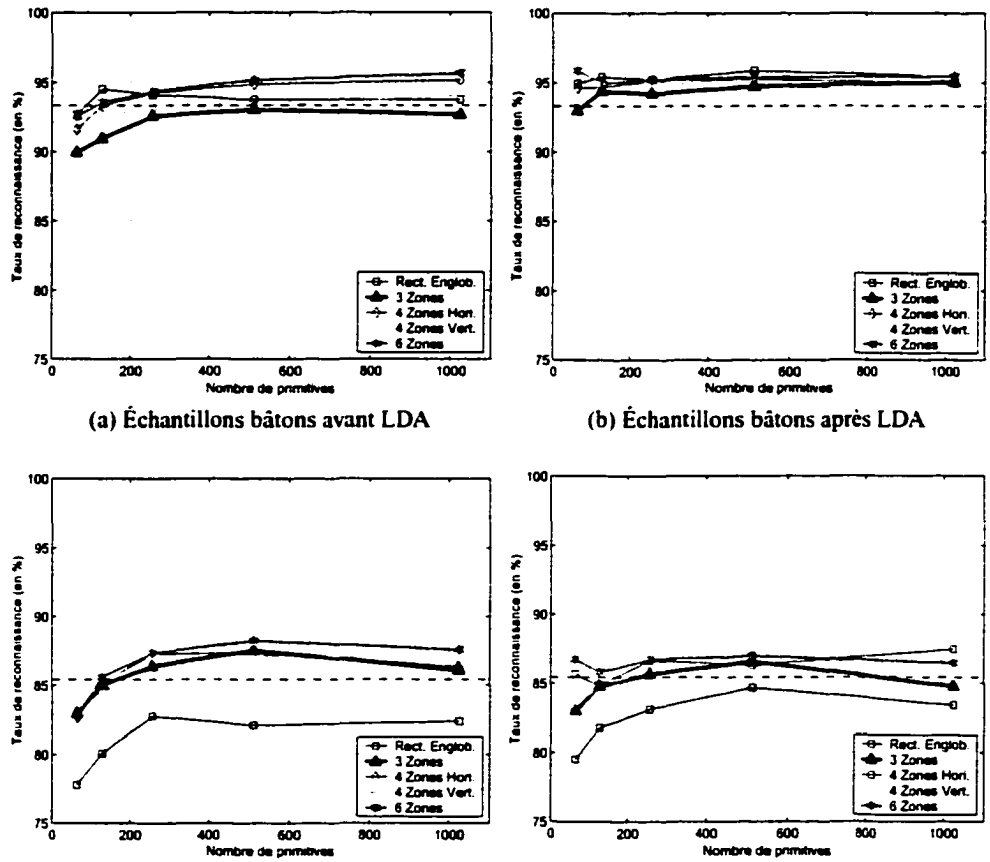
représentation DDD conduit aux meilleures performances, l'espace CCV+HD arrive en second. En particulier le meilleur système est celui utilisant l'espace DDD, quatre zones avec un découpage vertical de la zone médiane et 256 primitives

5.1.2.2 Analyse en fonction du type d'écriture

Afin de compléter cette étude, l'influence de la prise en compte des zones sur la reconnaissance des exemples bâtons et cursifs a été étudiée. Pour cela les taux de reconnaissance obtenus sont présentés, en fonction des espaces de représentation utilisés, aux figures 52, 53, 54 et 55. La ligne discontinue noire sur chacune d'elles représente la performance atteinte par le système standard.

De manière à comparer directement les performances entre échantillons bâtons et cursifs, l'axe des ordonnées des différents graphiques présente la même gamme de valeurs pour les systèmes utilisant un même espace de représentation. Une première constatation s'impose : les taux de reconnaissance observés pour les échantillons bâtons sont toujours supérieurs à ceux obtenus pour les échantillons cursifs. Cette remarque caractérise la difficulté associée à la reconnaissance de l'écriture manuscrite cursive.

Concernant les exemples bâtons, nous pouvons remarquer que l'extraction de primitives sur les rectangles englobants permet globalement d'obtenir de meilleures performances que la prise en compte de zones. Cette remarque est plus flagrante après l'application de l'algorithme LDA. En fonction de l'espace de représentation utilisé, elle ne conduit pas aux mêmes résultats. Pour l'espace DDD l'application de l'algorithme LDA engendre un gain de performance significatif. Par contre, pour les autres espaces de représentation elle conduit parfois à une diminution des taux de reconnaissance, en particulier pour les système utilisant six zones. L'apport de l'algorithme LDA se fait remarquer plus particulièrement pour les ensembles de petites tailles.



(a) Échantillons bâtons avant LDA

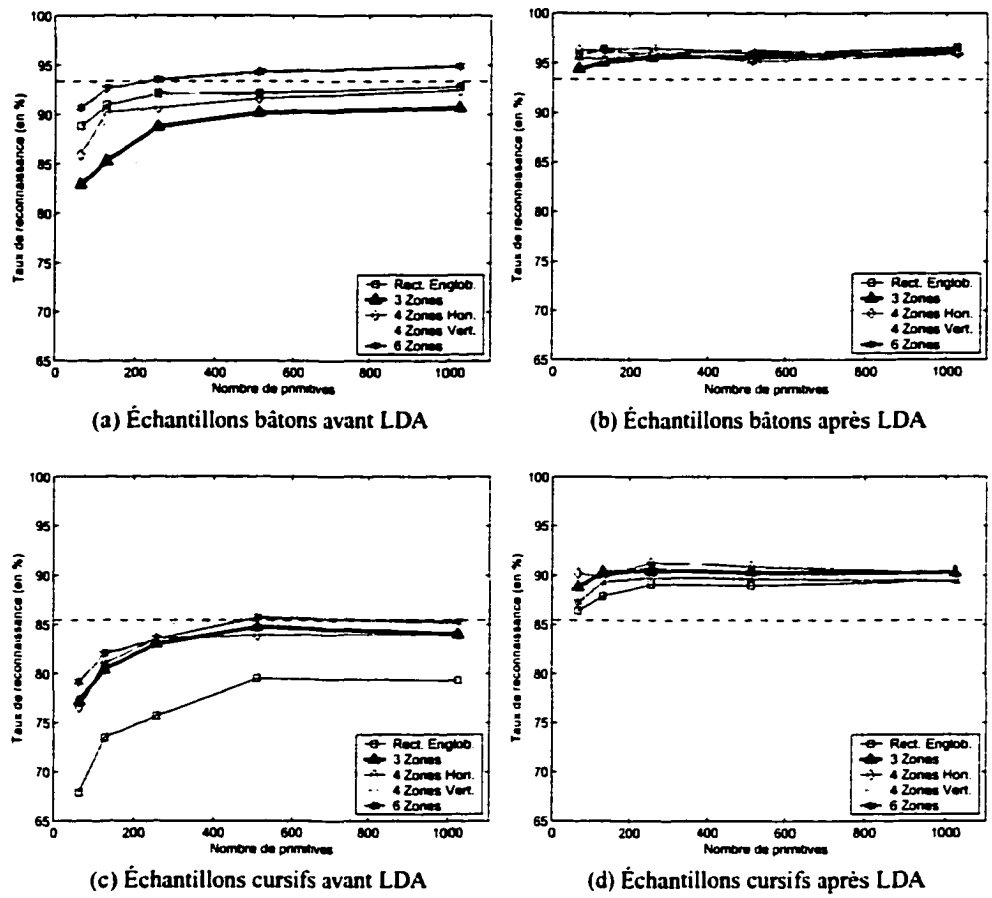
(b) Échantillons bâtons après LDA

(c) Échantillons cursifs avant LDA

(d) Échantillons cursifs après LDA

FIGURE 52 Évaluation des stratégies de *zoning* sur les systèmes utilisant des primitives issues de l'espace de représentation CCV – Présentation en fonction du type d'écriture des échantillons.

L'analyse des graphiques associés à l'écriture cursive montre que la prise en compte des zones conduit toujours à une augmentation significative des taux de reconnaissance par rapport à l'extraction sur les rectangles englobants. Ceci confirme l'intérêt de la technique de *zoning* pour la reconnaissance de l'écriture manuscrite cursive. Pour l'espace de représentation CCV, les différentes stratégies de découpage conduisent à des performances similaires, avec un petit avantage à la prise en compte de six zones. La même remarque est applicable aux espaces DDD et CCV+HD. Par contre pour HD, l'augmentation du nombre de zones conduit à une détérioration des performances. Nous avons déjà disserté



(a) Échantillons bâtons avant LDA

(b) Échantillons bâtons après LDA

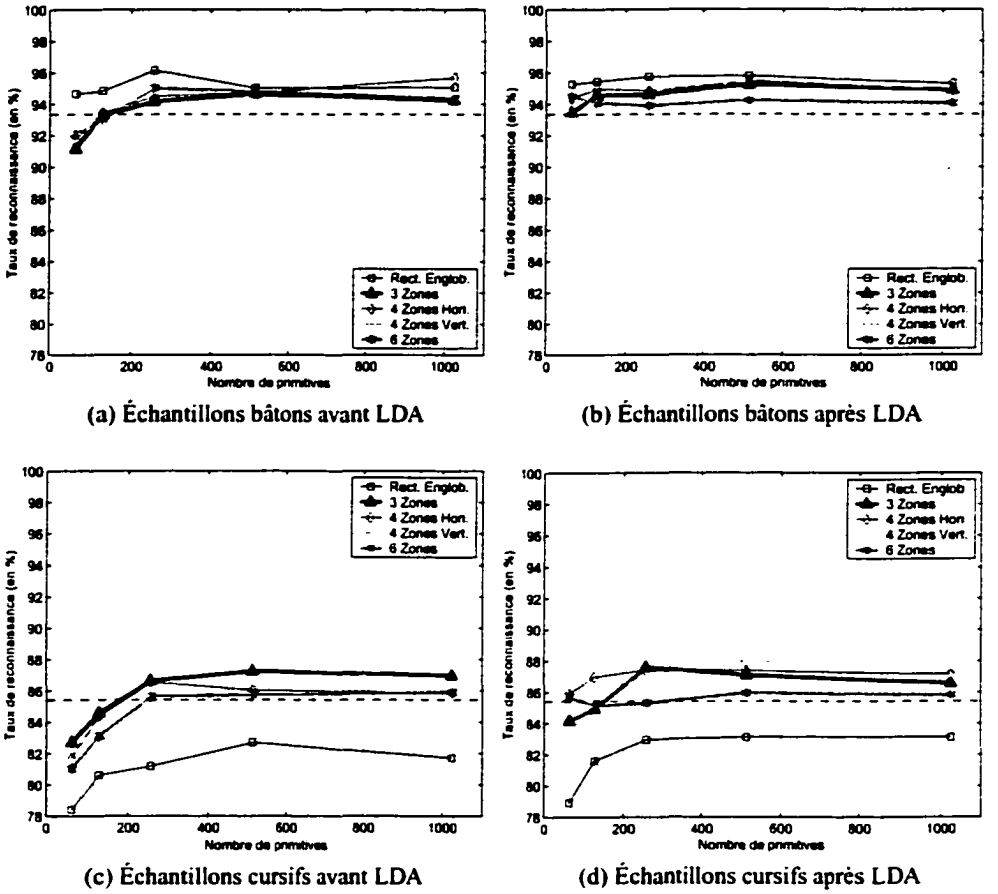
(c) Échantillons cursifs avant LDA

(d) Échantillons cursifs après LDA

FIGURE 53 Évaluation des stratégies de *zoning* sur les systèmes utilisant des primitives issues de l'espace de représentation DDD – Présentation en fonction du type d'écriture des échantillons.

sur la raison de ce phénomène lors de l'analyse des performances globales, il s'agit de la faible densité d'information présente dans le contour des graphèmes.

L'application de l'algorithme LDA sur les échantillons cursifs conduit à une amélioration des performances de l'ensemble des systèmes uniquement pour l'espace de représentation DDD. Pour les autres, il n'est bénéfique que lorsque la taille de l'ensemble de primitives est petite. Nous pensons que ce phénomène est dû à la non-optimalité de l'algorithme de détection des zones. En effet plusieurs exemples d'un même caractère, différemment



(a) Échantillons bâtons avant LDA

(b) Échantillons bâtons après LDA

(c) Échantillons cursifs avant LDA

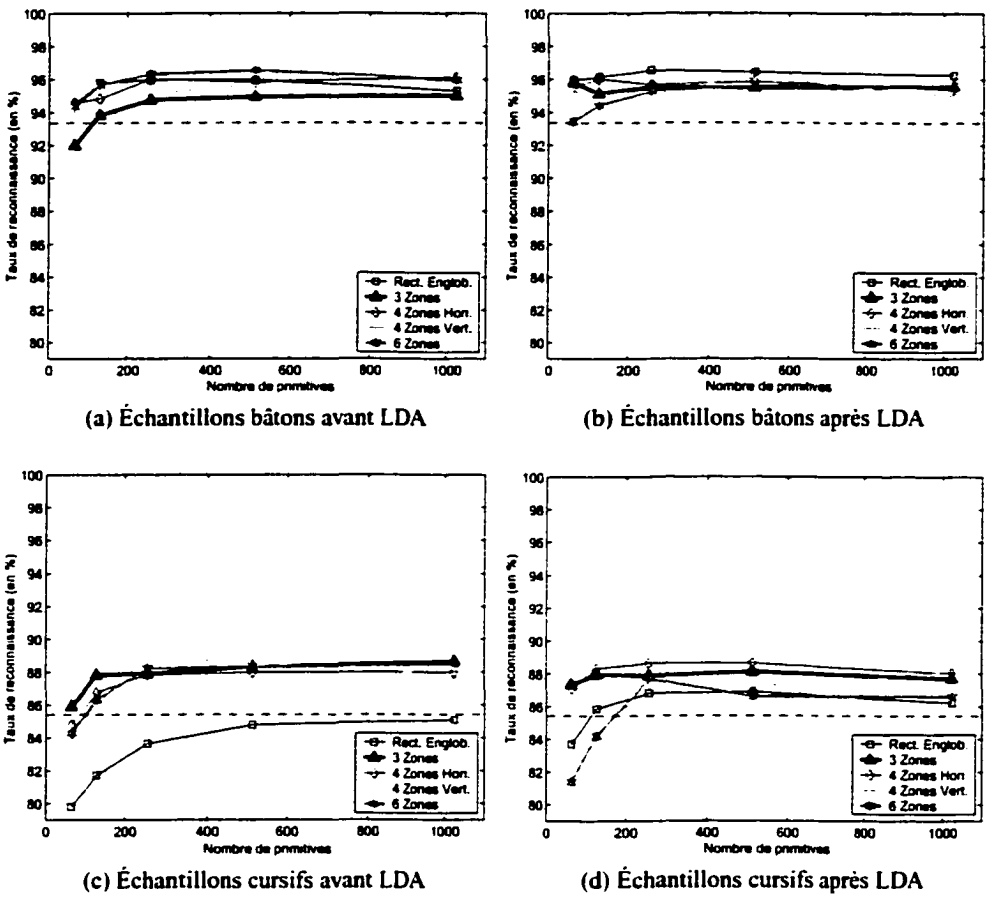
(d) Échantillons cursifs après LDA

FIGURE 54 Évaluation des stratégies de *zoning* sur les systèmes utilisant des primitives issues de l'espace de représentation HD – Présentation en fonction du type d'écriture des échantillons.

positionnés par rapport aux lignes de bases, conduiront à des vecteurs caractéristiques différents, même si la forme des graphèmes est similaire. Il est possible alors que ce phénomène rende les classes de notre modélisation non séparables.

5.1.2.3 Conclusions sur la prise en compte des zones d'écriture

La première conclusion est que globalement la prise en compte des zones d'écriture conduit à une augmentation des performances. En fait cette dernière se remarque particulièrement



(a) Échantillons bâtons avant LDA

(b) Échantillons bâtons après LDA

(c) Échantillons cursifs avant LDA

(d) Échantillons cursifs après LDA

FIGURE 55 Évaluation des stratégies de *zoning* sur les systèmes utilisant des primitives issues de l'espace de représentation CCV+HD – Présentation en fonction du type d'écriture des échantillons.

sur les exemples cursifs. Ceci confirme que la prise en compte de zones permet l'intégration d'information contextuelle dans la définition même des primitives et les rend plus discriminantes pour l'écriture cursive.

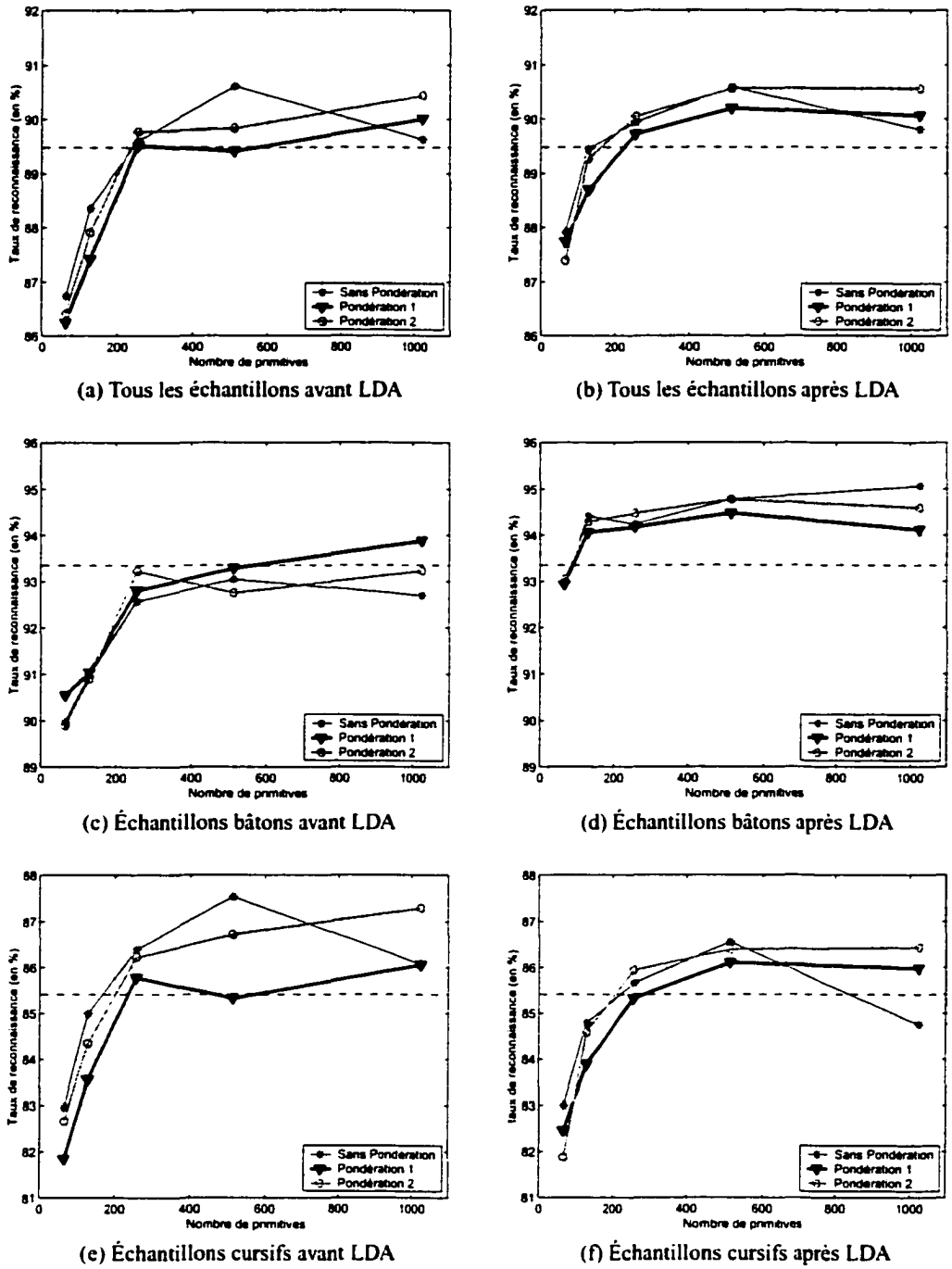
L'application de l'algorithme LDA ne conduit à une nette amélioration des performances que dans le cas de l'espace de représentation DDD. Pour les autres, son effet bénéfique n'est remarqué que pour les ensembles de primitives de petites tailles, c'est-à-dire dans notre cas inférieur ou égal à 128 primitives.

5.1.3 Prise en compte de la pondération des zones

Dans le but de réduire l'influence de la prise en compte de zones sur les échantillons de type bâton, nous avons proposé, dans la section 3.3.4, une stratégie de pondération des zones de dépassement en fonction de leur hauteur relative à la zone médiane. Deux techniques ont été envisagées pour sa mise en œuvre. La première consiste à intégrer la pondération lors du calcul de la distance entre les échantillons et les centres de gravité. Cette approche a l'avantage de ne pas modifier les composantes des vecteurs caractéristiques associés aux échantillons. Cependant elle ne peut plus être utilisée après LDA, où chaque composante de l'espace résultant est une combinaison linéaire de celles de l'espace de départ. La seconde approche proposée consiste à pondérer directement les composantes des vecteurs caractéristiques et donc les modifier définitivement.

Afin d'évaluer l'influence sur les performances de ces deux techniques, nous présentons sur les figures 56 et 57 les taux de reconnaissance obtenus, lors de la prise en compte de trois zones, par les systèmes de reconnaissance utilisant les espaces de représentation CCV et DDD respectivement. Ces données représentent globalement le comportement observé pour tous les espaces. Les différents graphiques n'utilisent pas la même gamme de valeurs en ordonnées afin de mieux évaluer l'influence des stratégies de pondération. L'expression "Pondération 1" fait référence à l'approche ne modifiant pas les composantes du vecteur caractéristique des échantillons par opposition à la notation "Pondération 2".

Notre analyse des performances globales des différents systèmes montre que la prise en compte de la pondération conduit globalement à une légère diminution des taux de reconnaissance. L'analyse des graphiques associés aux échantillons bâtons permet de remarquer que pour ce type d'écriture les techniques de pondération produisent une augmentation des taux de reconnaissance. Cela signifie que les deux techniques permettent d'atteindre l'objectif visé, c'est-à-dire diminuer l'influence de la prise en compte de zones sur l'écriture bâton. Cependant un effet de bord important peut être remarqué sur les échantillons cur-



(a) Tous les échantillons avant LDA

(b) Tous les échantillons après LDA

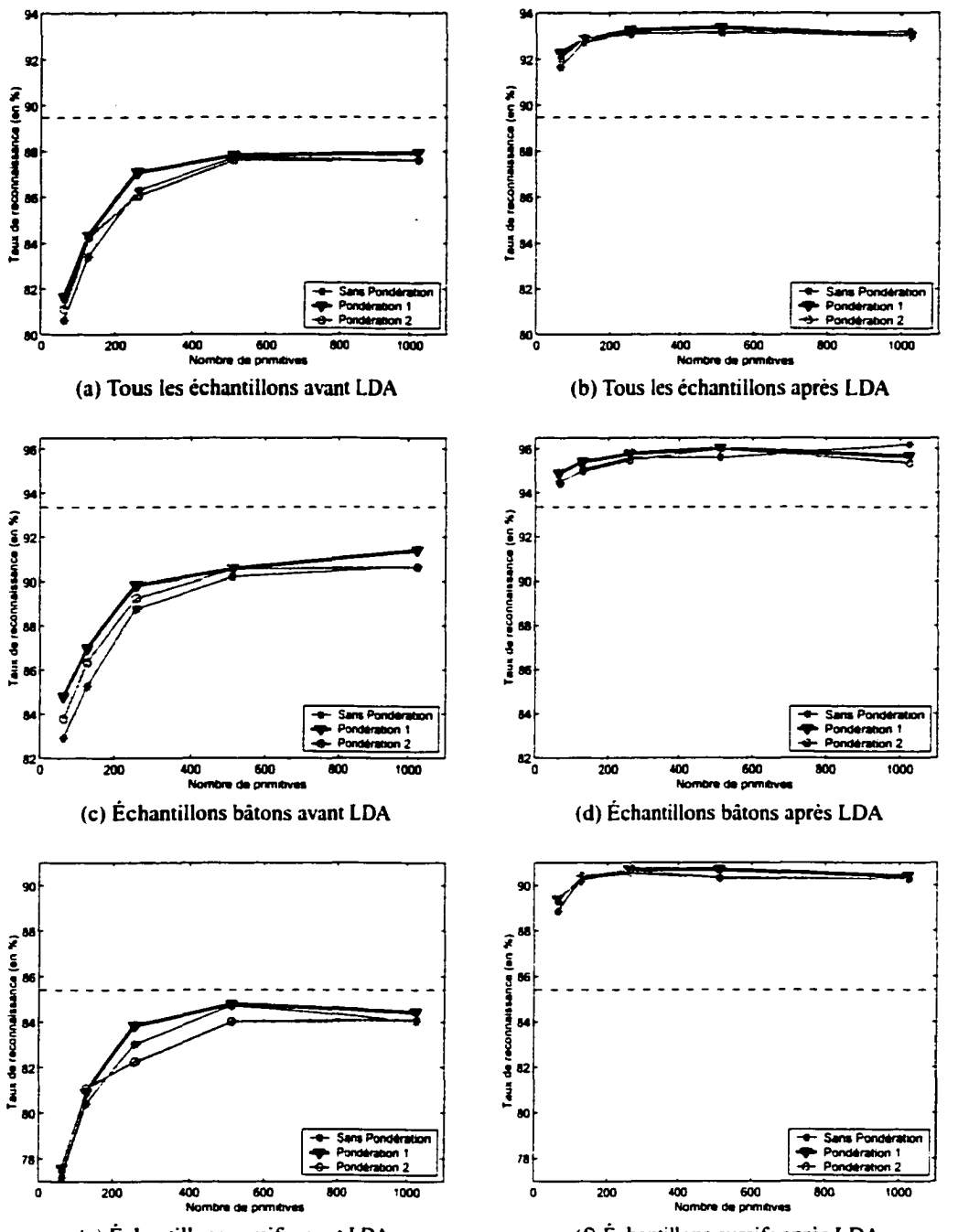
(c) Échantillons bâtons avant LDA

(d) Échantillons bâtons après LDA

(e) Échantillons cursifs avant LDA

(f) Échantillons cursifs après LDA

FIGURE 56 Évaluation de la prise en compte des stratégies de pondération des zones pour l'espace de représentation CCV et en considérant les trois zones d'écriture.



(a) Tous les échantillons avant LDA

(b) Tous les échantillons après LDA

(c) Échantillons bâtons avant LDA

(d) Échantillons bâtons après LDA

(e) Échantillons cursifs avant LDA

(f) Échantillons cursifs après LDA

FIGURE 57 Évaluation de la prise en compte des stratégies de pondération des zones pour l'espace de représentation DDD et en considérant les trois zones d'écriture.

sifs. En effet pour ce type d'écriture la pondération des zones conduit à une diminution des taux de reconnaissance. L'application de l'algorithme LDA ne modifie pas réellement le comportement des différentes courbes.

Nous pouvons remarquer également que suivant l'espace de représentation considéré, les deux stratégies de pondération ne se comportent pas de la même manière. Pour l'espace CCV la seconde stratégie détériore moins les performances obtenues pour les échantillons cursifs que la première, alors que pour l'espace DDD c'est l'inverse. Cela signifie que certains espaces de représentation sont plus sensibles à la modification des valeurs des vecteurs caractéristiques.

En conclusion les stratégies de pondération proposées rencontrent l'objectif fixé, à savoir réduire l'influence de la prise en compte des zones sur les échantillons de type bâton. Cependant elles entraînent un effet de bord important sur les échantillons cursifs. Parmi les deux stratégies proposées, la meilleure dépend de la nature de l'espace de représentation considéré.

5.1.4 Conclusions et amélioration possibles

Dans cette section nous avons présenté les résultats expérimentaux rattachés à l'évaluation de différents espaces de représentation ainsi que de différentes stratégies visant l'amélioration du pouvoir discriminant des ensembles de primitives résultants.

5.1.4.1 Conclusions sur les espaces de représentation

Différentes conclusions découlent de cette analyse. Premièrement l'utilisation de l'algorithme LDA permet d'obtenir des primitives plus discriminantes. Cette remarque est particulièrement vraie lors de l'utilisation d'ensembles de primitives de petite taille (inférieure à 200 primitives).

La deuxième conclusion est que la prise en compte des zones lors de l'extraction de caractéristiques permet également une augmentation des performances globales des systèmes de reconnaissance, quel que soit l'espace de représentation utilisé. Cependant elle conduit à une diminution des performances associées aux échantillons bâtons, par rapport à l'extraction sur les rectangles englobants. Afin de pallier ce problème, nous avons proposé une stratégie de pondération des caractéristiques extraites des zones de dépassements. Elle permet effectivement de diminuer la perte de performance pour les échantillons bâtons, mais elle en induit une pour les échantillons cursifs. Nous avons également mis en évidence la complémentarité des espaces de représentation CCV et HD. En effet leur combinaison mène toujours à des performances plus élevées que leur utilisation individuelle.

Une remarque découlant de cette phase expérimentale est que la connaissance, au moment de l'extraction de caractéristiques, du type de l'échantillon traité permettrait de simplifier cette étape. En effet nous pourrions construire séparément deux ensembles de primitives, l'un dédié à l'écriture cursive et l'autre à l'écriture bâton.

5.1.4.2 Conclusions sur les performances atteintes

Le système conduisant à la meilleure performance globale est celui utilisant l'espace de représentation DDD, la prise en compte de quatre zones avec division verticale de la zone médiane, l'application de l'algorithme LDA et un ensemble de 256 primitives. Il parvient à un taux de reconnaissance de 93,8% lors de l'utilisation d'un lexique de taille 1 000 et de 85,1% pour un lexique de taille 10 000. Pour le système standard, les performances équivalentes sont de 89,5% et 77,3% respectivement. Il est à noter également que le nombre de primitives a été réduit puisque nous passons de 378 à 256 primitives, soit une réduction d'environ un tiers.

Concernant les échantillons bâtons, les taux de reconnaissance les plus élevés sont atteints par le système utilisant un ensemble de 256 primitives extraites sur les rectangles englobant par l'intermédiaire de l'espace de représentation CCV+HD et après LDA. Pour un

lexique de taille 1 000 le taux de reconnaissance atteint est de 96,6%, soit une augmentation de 3,2% par rapport au système standard.

Le système utilisant l'espace de représentation DDD, 256 primitives, la prise en compte de quatre zones avec division horizontale de la zone médiane, après LDA, obtient le taux de reconnaissance le plus élevé pour les échantillons cursifs, soit 91,3%. Ceci représente un gain absolu important (6,9%) par rapport au système standard pour lequel la performance atteinte est de 85,4%.

Ces remarques confirment nos premières constatations concernant la faiblesse de la représentation de l'information utilisée par le système standard, en particulier pour l'écriture cursive.

5.1.4.3 Améliorations possibles et discussion

L'amélioration des techniques d'extraction de caractéristiques est certainement possible. En particulier, la prise en compte du contexte local lors de l'extraction est importante. Nous avons testé une technique dans le but d'intégrer le contexte gauche/droit, en prolongeant les sondes d'analyse dans les graphèmes voisins, lors de l'extraction de concavités. Cependant, cette technique n'a pas conduit aux résultats escomptés. L'augmentation du nombre de dimensions de l'espace de représentation et la distribution de l'information sur les différents axes se sont traduites par une augmentation de la confusion du système. Nous avons également remarqué que l'espace résultant n'est plus séparable linéairement par rapport aux classes de la modélisation. Une autre approche doit être envisagée afin d'intégrer le contexte gauche/droit des graphèmes.

Une technique similaire a été appliquée lors de l'extraction des primitives à partir de l'espace de représentation DDD. Au lieu d'utiliser la technique de *tiling* proposée par l'auteur [122], les graphèmes gauche et droit sont considérés lors de l'analyse. De même que pour

les concavités, les résultats obtenus ne sont pas concluants du fait de l'augmentation de la complexité associée à l'approche.

Les perspectives en termes d'amélioration de la représentation de l'information doivent être orientées vers l'intégration du contexte local des graphèmes, en particulier pour l'écriture manuscrite cursive. Les approches testées n'ont pas conduit aux résultats escomptés. Cependant il faut persister dans cette voie. Pour cela différentes possibilités sont envisageables. Une première consiste à modifier la technique de prise en compte des zones en ajoutant un facteur de recouvrement de ces dernières. Nous pouvons également étudier les possibilités qu'offrent les techniques d'extraction basées sur les projections. En particulier des projections localisées au point de ligature des graphèmes devraient permettre d'introduire plus de contexte local. Cette approche pourrait être utilisée afin d'améliorer l'ensemble de cinq primitives permettant de caractériser les points de segmentation.

Les conclusions et remarques de cette section portent uniquement sur la définition de primitives. Cependant l'étape de reconnaissance permet en fait d'associer une classe de notre modélisation à une forme inconnue à partir de la représentation symbolique qu'est la primitive. Le pouvoir discriminant d'une primitive permet de quantifier sa faculté d'attribuer la bonne classe à une forme inconnue. Un travail au niveau des classes peut également apporter un gain en terme de performance.

Une remarque à ce sujet est que l'application de l'algorithme LDA permet d'obtenir des performances élevées lors de l'utilisation d'ensembles de primitives contenant seulement 64 primitives. Le nombre de classes de notre modélisation étant de 209, cela signifie qu'un tel ensemble permet de regrouper plusieurs classes sous une même primitive. Cette information est intéressante et pourrait certainement être exploitée de manière à lier certaines classes. En effet le grand nombre de ces dernières vient de notre stratégie favorisant la sur-segmentation lors des pré-traitements. Cependant il est certain que plusieurs classes correspondent à des graphèmes de formes similaires. Par exemple les formes associées

aux caractères n et m ou encore i et u seront très proches. Un travail au niveau de la prise en compte de ces remarques devrait permettre une augmentation de la performance globale du système de reconnaissance.

5.2 Évaluation de l'algorithme de sélection

Dans la section précédente, nous avons présenté l'évaluation de plusieurs espaces de représentation. Leur développement a été effectué afin de disposer de plusieurs sources d'information pouvant être intégrées par notre algorithme de sélection de caractéristiques. Ce dernier a pour but l'amélioration de la représentation de l'information utilisée par le système de reconnaissance. Plusieurs expériences ont été réalisées afin d'évaluer sa pertinence. Nous allons les décrire dans cette section.

5.2.1 Validation de l'algorithme

Nous avons réalisé une première expérience de manière à valider notre implémentation de l'algorithme. Dans ce but nous avons utilisé l'ensemble de primitives perceptuelles comme premier niveau perceptif. Notre algorithme est alors appliqué mais en modifiant légèrement son fonctionnement. Premièrement le seuil de pouvoir discriminant τ_1 est fixé à 0. Cela revient à considérer toutes les primitives comme non-discriminantes. L'identification de regroupements de primitives n'est pas réalisée, conduisant à l'amélioration individuelle de chaque primitive de l'ensemble de départ. Finalement seul l'ensemble de primitives bâton est utilisé comme source d'information intégrable. L'ensemble de primitives résultant de ce processus doit être similaire à celui utilisé par le système standard. La comparaison des performances obtenues permettra de valider notre implémentation.

5.2.1.1 Description de la procédure utilisée

Nous allons rappeler brièvement les différentes étapes du déroulement de l'algorithme. À partir de l'ensemble de primitives perceptuelles un système de reconnaissance est construit.

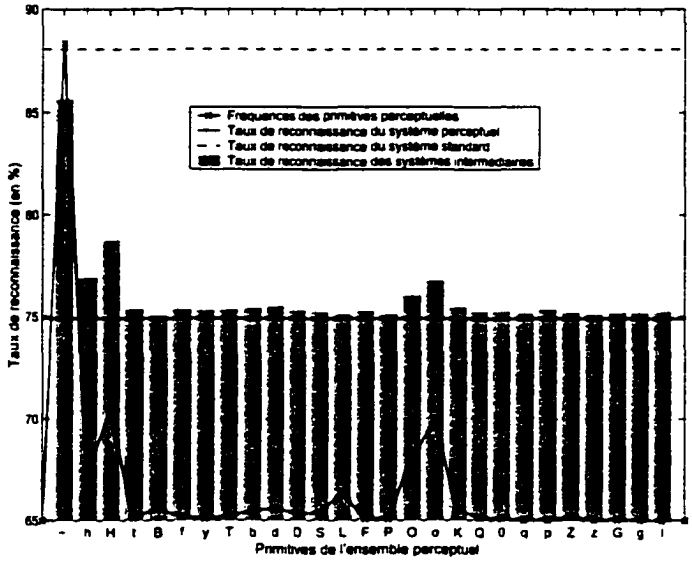


FIGURE 58 Performances globales des différents systèmes intermédiaires lors de l'amélioration de l'ensemble perceptuel par l'ensemble bâton.

L'évaluation de sa performance et du pouvoir discriminant des primitives (voir tableau V) est réalisée. L'ordonnancement de l'ensemble est alors possible. Contrairement au déroulement classique, l'évaluation du seuil de pouvoir discriminant τ_i et la recherche de regroupements n'est pas nécessaire puisque chaque primitive est remplacée individuellement par l'ensemble bâton. Pour cela les graphèmes caractérisés au premier niveau par la primitive cursive considérée sont re-caractérisés par l'une des 14 primitives composant l'ensemble bâton. Cette approche conduit à considérer un ensemble de $27 - 1 + 14 = 40$ primitives (27 étant le nombre de primitives de l'ensemble perceptuel). Chaque primitive étant remplacée individuellement, 27 systèmes de reconnaissance intermédiaires sont construits. L'ensemble de primitives du second niveau perceptif est alors obtenu en regroupant les primitives utilisées par les différents systèmes intermédiaires. Il contient alors $27 \times 14 = 378$ primitives. Le système final peut alors être construit et ses performances évaluées.

5.2.1.2 Résultats obtenus

La performance des différents systèmes intermédiaires est présentée sur les figures 58 et 59. La première montre les taux de reconnaissance globaux obtenus sur le corpus de validation lors de l'utilisation d'un lexique de taille 1 000, alors que la seconde présente ces mêmes performances mais en fonction du type d'écriture des échantillons. Les différents systèmes sont caractérisés par le symbole associé à la primitive substituée (voir annexe 1.1). Ils sont ordonnés en fonction du pouvoir discriminant associé aux primitives remplacées, le plus proche de l'origine correspondant à la substitution de la primitive la moins discriminante (voir tableau V). Nous avons ajouté sur la figure 58 une courbe supplémentaire représentant les fréquences d'occurrence des primitives (données de la 2è colonne du tableau V) afin d'évaluer l'impact de ce paramètre. Deux droites sont également visibles, la continue représentant la performance atteinte par le système avant application de l'algorithme et la discontinue celle du système final. Par contre sur la figure 59, les deux droites présentent la performance avant application de l'algorithme, la continue pour les échantillons bâtons et la discontinue pour les cursifs.

L'analyse de ces deux figures montre que la substitution d'une primitive perceptuelle par l'ensemble bâton conduit toujours à une augmentation des performances, parfois très faible. Nous pouvons également remarquer une certaine corrélation entre le gain de performance et la fréquence de la primitive substituée. En effet, plus une primitive est fréquente, plus le nombre de graphèmes pouvant être mieux classés par l'intermédiaire des nouvelles primitives est grand. Cependant le système substituant la primitive "**B**" a un comportement un peu différent puisque malgré la fréquence de cette primitive, relativement plus élevée, le gain en terme de performance est inférieur à celui des autres primitives. La primitive "**B**" caractérise la présence d'un grand dépassement bas uniquement. Il est fort probable que peu de graphèmes extraits d'échantillons bâtons soient caractérisés par cette primitive. Le fait de la substituer par des primitives dédiées à l'écriture bâton n'a alors pas d'intérêt, ce que traduit la performance obtenue.

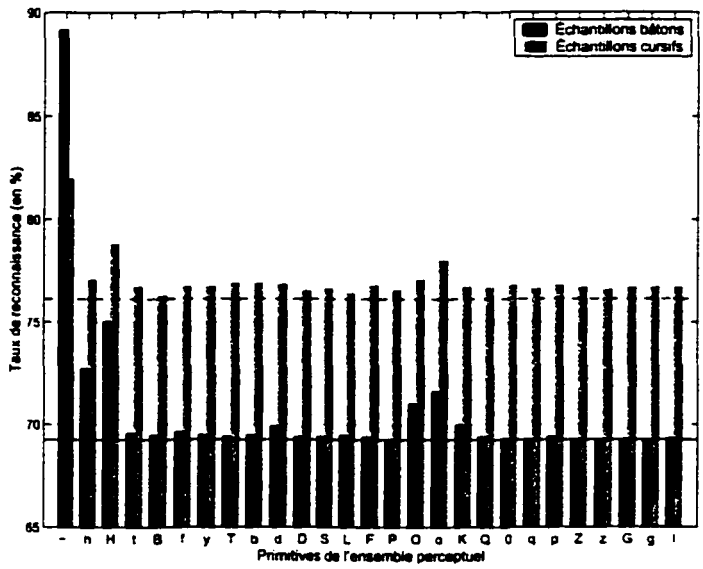


FIGURE 59 Performances des différents systèmes intermédiaires lors de l'amélioration de l'ensemble perceptuel par l'ensemble bâton – Présentation en fonction du type d'écriture des échantillons.

La figure 59 permet de constater que seule la substitution de la primitive "-" permet d'obtenir un taux de reconnaissance plus élevé pour les échantillons bâtons que pour les cursifs. Cela signifie que cette primitive regroupe, lors de l'utilisation de l'ensemble perceptuel seul, une grande partie des graphèmes associés à l'écriture bâton. Nous pouvons également identifier les autre primitives perceptuelles caractérisant une quantité importante de graphèmes liés à l'écriture bâton, en observant le gain apporté par l'intégration des primitives bâtons. Celles qui conduisent au plus forte augmentation du taux de reconnaissance sont les primitives perceptuelles : "-", "**H**", "**h**", "**o**", "**O**" et "**K**". Elles caractérisent toutes des graphèmes sans dépassement bas, ce qui correspond bien aux graphèmes de l'écriture bâton.

Finalement nous présentons dans le tableau X les taux de reconnaissance globaux obtenus sur le corpus de test par le système résultant de l'application de l'algorithme. Nous avons

également indiqué à titre de comparaison les performances équivalentes obtenues par le système utilisant uniquement l'ensemble perceptuel ainsi que celles du système standard.

TABLEAU X

Taux de reconnaissance obtenu par le système résultant de l'amélioration de l'ensemble perceptuel par l'ensemble bâton.

Taille du lexique utilisé	10	100	1 000	5 000	10 000
Système Perceptuel	96,9%	89,6%	75,0%	62,2%	56,9%
Système Perceptuel Amélioré	99,0%	95,7%	89,5%	81,6%	77,4%
Système Standard	98,9%	95,7%	89,5%	82,2%	77,3%

L'analyse de ces valeurs montre que les performances atteintes par le système amélioré sont équivalentes à celle du système standard. Une légère différence peut être remarquée. Les deux systèmes utilisent bien le même ensemble de primitives ainsi que les mêmes procédures d'apprentissage et de test. La seule différence se situe au niveau du codage de l'indice des primitives au sein de notre programme informatique (dans le système amélioré, les primitives du premier niveau sont ordonnées alors que dans le système standard l'ordre est celui de définition des primitives défini dans l'annexe 1.1). L'initialisation aléatoire des paramètres diffère alors même pour une même initialisation du générateur de nombre aléatoire de la machine utilisée.

Cette expérience valide le fonctionnement de notre algorithme d'intégration de différentes sources d'information. Les performances obtenues, similaires à celles du système standard, permettent de confirmer la validité de son implémentation.

5.2.2 Amélioration de l'ensemble de primitives perceptuelles

Le nouvel algorithme présenté à la section 4.3 a été développé dans le but d'intégrer de nouvelles sources d'information au système standard du SRTP. Dans cette optique, nous avons mis en œuvre plusieurs espaces de représentation permettant ainsi d'obtenir dif-

férentes sources d'information. Dans cette section nous allons présenter les expériences réalisées pour améliorer la performance du système standard.

Dans un premier temps nous devons choisir une représentation de l'information de départ parmi les deux ensembles de primitives qu'utilise le système de reconnaissance standard. Notre choix s'est arrêté sur l'ensemble de primitives perceptuelles car leur définition est basée sur des concepts de la perception humaine de l'écriture. Il est donc justifié d'utiliser une telle représentation au premier niveau perceptif de notre système.

5.2.2.1 Choix du seuil de pouvoir discriminant

Un système de reconnaissance est donc construit à partir de l'ensemble de primitives perceptuelles. Après l'évaluation de sa performance et celle du pouvoir discriminant des primitives, il faut déterminer la valeur du seuil τ_1 permettant d'identifier les primitives non-discriminantes. La définition proposée à l'équation 4.16 permet d'obtenir : $\tau_1 = 49.71$. L'utilisation de cette valeur conduit à n'isoler que la primitive "-" dans le sous-ensemble $\bar{D}_1$ (voir tableau V). Son faible pouvoir discriminant par rapport aux autres primitives est à la source de ce phénomène. Lors de notre discussion concernant la détermination de ce seuil, nous avons mentionné que pour un ensemble de primitives défini de manière explicite, il est préférable de choisir la valeur du seuil arbitrairement, après analyse des valeurs de pouvoir discriminant (voir section 4.3.6.2).

Cette étude a été réalisée lors du développement du premier prototype de notre algorithme [60]. Elle nous a conduits à choisir $\tau_1 = 25$, pour deux raisons. Premièrement un saut dans les valeurs de pouvoir discriminant peut être remarqué à ce niveau. Deuxièmement cette valeur permet de regrouper l'ensemble des primitives caractérisant les graphèmes ne contenant pas de boucles.

5.2.2.2 Identification des regroupements de primitives

Cette valeur de seuil permet de considérer sept primitives comme non-discriminantes : "**-**", "**h**", "**H**", "**t**", "**B**", "**f**" et "**y**". Dans [60] nous avons proposé quatre regroupements, obtenus arbitrairement, en fonction de propriétés communes aux différentes primitives :

- "**hH**" : primitives possédant un petit ou un grand dépassement haut seulement,
- "**tf**" : primitives possédant un grand dépassement bas et un petit ou un grand dépassement haut,
- "**By**" : primitives possédant un petit ou un grand dépassement bas seulement,
- "**-**" : primitives ne possédant pas de dépassement ni de boucle.

L'utilisation de la stratégie de regroupement proposée à la section 4.3.7 conduit à l'identification des mêmes regroupements. Cette constatation permet d'affirmer la validité de notre approche, basée sur l'analyse des distributions de probabilités pour effectuer les regroupements.

TABLEAU XI

Valeur des bornes caractérisant la taille des nouveaux ensembles de primitives en fonction des regroupements identifiés.

Regroupement	Fréquence	$\mathfrak{N}_{min}$	$\mathfrak{N}_{max}$
hH	17,56%	3 (8)	7 (128)
tf	0,77%	2 (4)	4 (16)
By	1,58%	2 (4)	4 (16)
-	51,90%	5 (32)	9 (512)

Une fois les regroupements déterminés, l'étape de sélection à proprement parler commence. Pour cela plusieurs ensembles de primitives sont construits afin de substituer chaque regroupement. Afin de limiter leur nombre, nous avons défini deux bornes : $\mathfrak{N}_{min}$ et $\mathfrak{N}_{max}$ (voir section 4.3.8.1). Les valeurs obtenues pour les quatre regroupements sont présentées dans le tableau XI. Entre parenthèses nous avons indiqué l'équivalence en nombre

de primitives. À titre indicatif, la fréquence d'apparition associée aux primitives du regroupement considéré est indiquée. Pour un espace de représentation donné, les valeurs de ces bornes conduisent à la construction de cinq ensembles de primitives pour les regroupements "**hH**" et "**-**" et trois pour les regroupements "**tf**" et "**By**".

5.2.2.3 Les sources d'information intégrables

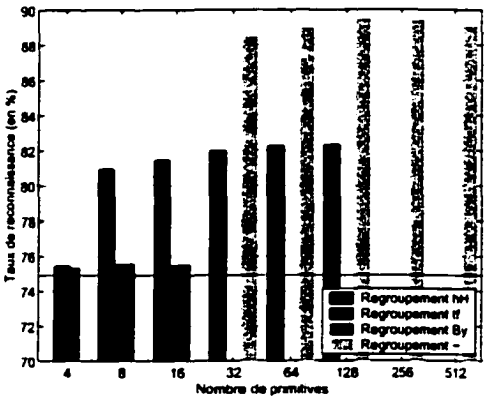
Dans la première partie de ce chapitre, plusieurs espaces de représentation ont été évalués. Afin de limiter le temps de calcul associé à l'application de notre algorithme d'intégration d'information, nous avons décidé de ne pas les utiliser tous. La sélection des espaces de représentation a été effectuée en fonction des performances obtenues lors de leurs évaluations individuelles.

L'apport de l'algorithme LDA étant non négligeable, en particulier pour les ensembles de petite taille, son utilisation est préconisée. Les cinq espaces de représentation retenus sont l'espace CCV en prenant en compte six zones (*CCV-6z-apLDA*), l'espace DDD avec extraction sur les rectangles englobants(*DDD-BB-apLDA*), la prise en compte de quatre zones avec division horizontale de la zone médiane (*DDD-4zH-apLDA*) et finalement l'espace CCV+HD avec extraction sur les rectangles englobants (*CCV+HD-BB-apLDA*) et prise en compte de quatre zones avec division horizontale de la zone médiane (*CCV-4zH-apLDA*).

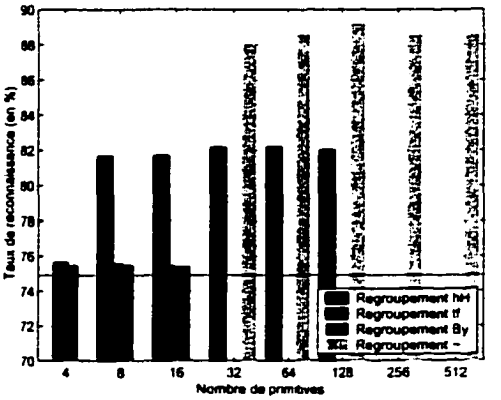
5.2.2.4 Évaluation des systèmes intermédiaires

La prise en compte de cinq espaces de représentation différents conduit à la construction de 25 systèmes de reconnaissance pour les regroupements "**hH**" et "**-**" et de 15 systèmes pour les regroupements "**tf**" et "**By**", soit un total de 80 systèmes différents.

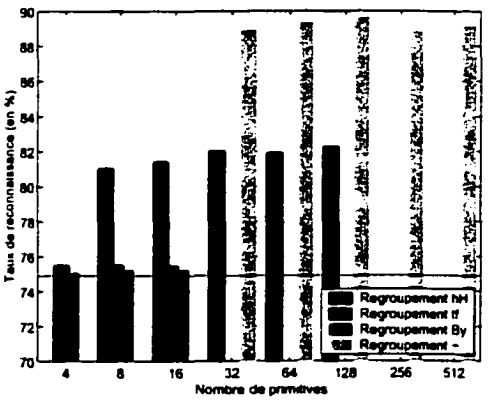
Nous présentons à la figure 60 l'évaluation des performances de ces différents systèmes, c'est-à-dire le taux de reconnaissance obtenu sur la base de validation lors de l'utilisa-



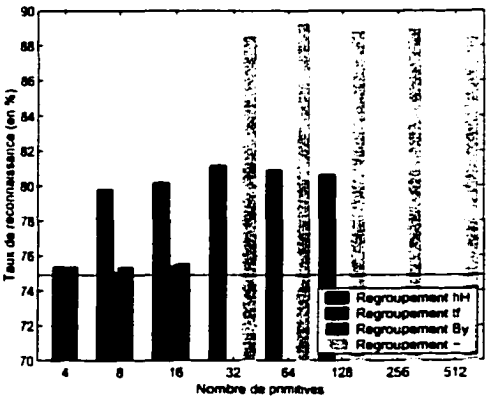
(a) Espace *DDD-BB-apLDA*



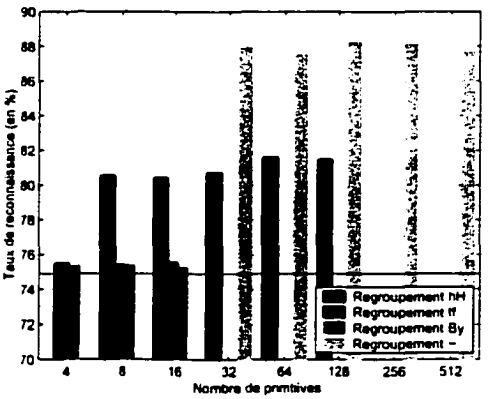
(b) Espace *CCV+HD-BB-apLDA*



(c) Espace *DDD-4zH-apLDA*



(d) Espace *CCV+HD-4zH-apLDA*



(e) Espace *CCV-6z-apLDA*

FIGURE 60 Performances des différents systèmes intermédiaires construits au premier niveau perceptif lors de l'amélioration de l'ensemble de primitives perceptuelles.

tion d'un lexique de taille 1 000. Chaque graphique représente l'évaluation des systèmes construits à partir d'un même espace de représentation. L'axe des abscisses représente le nombre de primitives de l'ensemble utilisé pour construire le système de reconnaissance. La ligne continue représente la performance atteinte par le système de départ (74,9%), c'est-à-dire celui utilisant l'ensemble de primitives perceptuelles.

Nous pouvons constater que les performances atteintes pour les regroupements "**hH**" et "**-**" sont bien plus élevées que celles obtenues pour les deux autres regroupements. Cette différence est liée à la fréquence des primitives associées aux regroupements remplacés. En effet la substitution du regroupement "**tf**" n'aura qu'un impact réduit sur les performances globales, puisqu'elle n'affecte la classification que de moins de 1% des graphèmes.

Nous pouvons remarquer tout de même que tous les systèmes construits conduisent à une amélioration du taux de reconnaissance. Cette constatation confirme la pertinence de notre approche consistant à améliorer la représentation de l'information. L'analyse des graphiques de la figure 60 montre que pour un regroupement donné les performances atteintes sont du même ordre quel que soit l'espace de représentation. Pour les regroupements "**hH**" et "**-**", la prise en compte d'un trop grand nombre de primitives induit une baisse des taux de reconnaissance. Cette remarque indique que les bornes définies par $\mathfrak{N}_{min}$ et $\mathfrak{N}_{max}$ permettent d'obtenir la performance maximale, dans l'espace de recherche parcouru qui est bien sûr limité par l'algorithme de quantification vectorielle.

Le résultat de la sélection du meilleur ensemble pour chaque regroupement est présenté dans le tableau XII. Nous pouvons constater que le même espace de représentation a été choisi pour substituer les trois premiers regroupements. Ce dernier correspond à celui conduisant aux meilleurs taux de reconnaissance pour les échantillons bâtons lors de son évaluation individuelle (voir section 5.1.4.2). Cette constatation confirme que certaines primitives de l'ensemble perceptuel caractérisent un grand nombre de graphèmes provenant de l'écriture bâton, en particulier "**h**", "**H**".

Concernant le regroupement "-" l'ensemble *DDD-4zH* a été choisi. Il s'agit de celui conduisant aux taux de reconnaissance les plus élevés pour les échantillons cursifs. Il est à noter également que sa performance pour les échantillons bâtons est relativement bonne. La grande fréquence d'apparition de la primitive "-" dans notre base de données implique qu'elle caractérise des graphèmes des deux types d'écriture. Le choix de cet ensemble de primitives peut signifier que la quantité de graphèmes associés à l'écriture cursive est plus important que celle des échantillons bâtons. Dans tous les cas cet espace de représentation semble un bon compromis entre les deux styles d'écriture.

TABLEAU XII

Résultat de la sélection des ensembles de primitives au premier niveau perceptif pour les quatre regroupements identifiés.

Regroupement	Espace choisi	Nombre de primitives	Performance
hH	*CCV+HD-BB*	32	82,2%
tf	*CCV+HD-BB*	4	75,6%
By	*CCV+HD-BB*	4	75,4%
-	*DDD-4zH*	128	89,6%

Lors de la description de notre algorithme, nous avons proposé un indicateur du pouvoir de généralisation d'un système de reconnaissance I_P (voir 4.3.4.2.3). Son intérêt est de fournir cette estimation à la fin de l'apprentissage, sans avoir à recourir à l'étape d'évaluation du système. La phase expérimentale concernant l'évaluation des différents espaces de représentation a montré que dans l'état actuel la valeur numérique retournée par cet indicateur n'est pas entièrement fiable. De ce fait, nous utilisons le taux de reconnaissance, évalué sur le corpus de validation avec un lexique de taille 1 000, pour sélectionner le meilleur ensemble de primitives permettant la substitution d'un regroupement. Cependant, nous voulons tout de même mentionner que l'utilisation de l'indicateur de performance, en supprimant l'étape d'évaluation des systèmes, permettrait une réduction du temps de calcul de l'ordre de 20%. Il serait donc intéressant de poursuivre le travail concernant cet

indicateur. À titre indicatif, l'apprentissage d'un système de reconnaissance sur un micro-ordinateur PC, utilisant un processeur AMD Athlon 1,1GHz et 512MB de mémoire RAM, dure en moyenne 14 minutes 20 secondes et la phase de test 3 minutes 25 secondes.

5.2.2.5 Les performances atteintes au second niveau perceptif

Une fois la sélection des nouvelles primitives réalisée, l'étape suivante consiste à les regrouper avec celles qui ont déjà été jugées discriminantes (sous-ensemble D_1) et ainsi former l'ensemble de primitives du niveau perceptif suivant. À partir de ce dernier, un système de reconnaissance est construit, permettant d'évaluer le pouvoir discriminant des primitives de ce nouvel ensemble. L'évaluation de sa performance est également réalisée. Cette dernière est utilisée pour l'évaluation du critère d'arrêt de l'algorithme.

Concernant le pouvoir discriminant des primitives, l'évaluation globale de l'ensemble conduit à une valeur de perplexité égale à 16,95 contre 42,15 pour l'ensemble de départ. La perplexité conditionnelle de la primitive la moins bonne passe de 61,14 à 42,21. L'évaluation des performances du système du second niveau perceptif est présentée dans le tableau XIII. La colonne "Valid" représente la performance obtenue sur le corpus de validation lors de l'utilisation d'un lexique de taille 1 000, les autres performances étant obtenues sur le corpus de test. Nous pouvons constater la nette amélioration de performance induite par l'application de notre algorithme. Les performances atteintes dépassent également largement celles du système standard tout en utilisant moitié moins de primitives. Cette dernière remarque induit également un gain au niveau du système, puisque la diminution du nombre de primitives permet d'obtenir une estimation plus fiable des paramètres, pour une base de données de taille fixe.

L'étape suivante de notre algorithme est l'évaluation du critère d'arrêt. Pour cela le gain relatif de performance est calculé par l'intermédiaire de l'équation 4.25. L'indicateur de performance utilisé est le taux de reconnaissance obtenu sur le corpus de validation. La

TABLEAU XIII

Performances atteintes aux différents niveaux perceptifs lors de l'amélioration de l'ensemble de primitives perceptuelles.

Niveau perceptif	Nb. primitives final	Taux de reconnaissance					
		Valid	10	100	1 000	5 000	10 000
1	27	74,9%	96,6%	89,7%	75,5%	61,7%	55,8%
2	188	91,7%	99,3%	97,1%	92,5%	86,2%	82,7%
3	235	91,9%	99,2%	97,0%	92,4%	86,2%	82,9%
Standard	378	88,4%	98,9%	95,7%	89,5%	82,2%	77,3%

valeur du gain est alors la suivante :

$$\delta(2) = \frac{91,7 - 74,9}{91,7 + 74,9} = 0,101 \tag{5.1}$$

Cette valeur étant supérieure à 10^{-2}, une nouvelle itération de l'algorithme est effectuée.

5.2.2.6 Le troisième niveau perceptif

L'ensemble associé au second niveau perceptif est composé en majorité de primitives issues d'une quantification vectorielle. La détermination du seuil de pouvoir discriminant est alors réalisée à partir de l'équation 4.16. La valeur obtenue permet d'isoler 17 primitives dans le sous-ensemble des non-discriminantes $\bar{D}_2$. Aucune de ces primitives ne provient de l'ensemble de départ, c'est-à-dire du sous-ensemble D_1, à ce niveau. L'analyse du sous-ensemble $\bar{D}_2$ permet de considérer 14 regroupements ou seulement trois contiennent deux primitives. Le calcul des bornes $\mathfrak{N}_{min}$ et $\mathfrak{N}_{max}$ pour chaque regroupement conduit à la construction de 150 systèmes de reconnaissance intermédiaires pour atteindre le troisième niveau perceptif. Les ensembles sélectionnés pour substituer les regroupements contiennent entre 2 et 16 primitives. Ils proviennent en majorité de l'espace de représentation *CCV-6z-apLDA* puisque 8 d'entre eux en sont issus. Pour le reste, deux sont construits à partir des espaces *DDD-BB-apLDA* et *DDD-4zH-apLDA*. Les deux derniers espaces de représentation ne sont représentés que par un ensemble de primitives.

L'obtention d'ensemble de primitives de petite taille à ce niveau perceptif est normale puisque la fréquence d'apparition des primitives substituant un regroupement sera forcément plus faible que celle des primitives de départ. Le choix majoritaire de l'espace de représentation *CCV-6z-apLDA* à ce niveau permet de conclure du bon fonctionnement de notre algorithme. En effet il est normal qu'à ce niveau le système essaye d'intégrer une source d'information qu'il n'a pas encore prise en compte.

Les performances globales atteintes au troisième niveau sont incluses dans le tableau XIII. Leur analyse montre que la descente de ce niveau perceptif n'est pas souhaitable, puisque le gain de performance n'est pas réellement significatif alors que 47 primitives sont ajoutées (soit une augmentation de 25%). L'évaluation du gain de performance ($\delta(3) = 0.001$) conduit à arrêter le déroulement de l'algorithme.

5.2.2.7 Conclusion sur la première expérience d'amélioration de la performance

La première conclusion de cette expérience est que notre algorithme conduit bien au résultat attendu, c'est-à-dire l'amélioration de la performance d'un système de reconnaissance par intégration de nouvelles sources d'information. Les performances obtenues sont supérieures à celles du système standard tout en utilisant moins de primitives.

Il semble que notre algorithme permette d'intégrer l'information, mise à disposition sous la forme de différents espaces de représentation, en une seule itération. En effet la descente au troisième niveau perceptif, malgré l'intégration de plusieurs dizaines de primitives n'apporte pas de gain conséquent de performances. Il est possible également que cette constatation soit le fait d'un phénomène de sur-apprentissage. En effet l'augmentation du nombre de primitives, sans accroître le nombre d'échantillons de la base de données, peut conduire à un tel problème. Dans le cas du système standard, la définition explicite des deux ensembles de primitives permet de l'éviter.

5.2.2.8 Modification du seuil de pouvoir discriminant au premier niveau perceptif

Lors de l'expérience décrite ci-dessus, le seuil de pouvoir discriminant associé au premier niveau perceptif τ_1 a été fixé arbitrairement à 25. Afin d'étudier l'influence de ce paramètre, nous avons décidé d'appliquer notre algorithme plusieurs fois en utilisant des valeurs décroissantes de ce seuil, de 25 à 0.

La décroissance de la valeur de seuil τ_1 conduit à une augmentation du nombre de primitives non-discriminantes. Pour une valeur de $\tau_1 = 0$, toutes les primitives de l'ensemble de départ seront considérées comme non-discriminantes. Nous présentons la décomposition en regroupements correspondante, dans le tableau XIV. Nous avons également indiqué la fréquence d'occurrence de chaque regroupement au premier niveau perceptif, ainsi que le résultat de la sélection réalisée par notre algorithme.

TABLEAU XIV

Regroupements identifiés lorsque le seuil de pouvoir discriminant est fixé à 0 – Résultat de la sélection des ensembles de primitives au premier niveau perceptif.

Regroupement	Fréquence	Espace choisi	Nombre de primitives	Performance
TbdD	3,66%	*DDD-BB*	8	76,4%
Zzg	0,50%	*CCV-6z*	4	75,4%
0T	0,76%	*CCV+HD-BB*	16	75,6%
hH	17,56%	*CCV+HD-BB*	64	82,3%
tf	0,77%	*CCV+HD-BB*	4	75,6%
By	1,58%	*CCV+HD-BB*	4	75,4%
-	51,90%	*DDD-4zH*	128	89,6%
L	3,05%	*DDD-4zH*	2	75,5%
F	0,22%	*CCV+HD-BB*	2	75,5%
P	0,19%	*CCV+HD-4zH*	4	75,5%
O	6,71%	*DDD-4zH*	8	77,0%
o	11,02%	*DDD-4zH*	32	78,4%
K	1,20%	*DDD-BB*	8	75,7%
Q	0,33%	*DDD-4zH*	2	75,4%
q	0,03%	*CCV-6z*	4	75,5%
p	0,05%	*CCV-6z*	4	75,5%
G	0,10%	*CCV-6z*	2	75,4%
I	0,01%	*CCV-6z*	4	75,5%

Une première remarque au sujet de ces données est qu'une corrélation peut toujours être constatée entre la fréquence d'occurrence d'un regroupement et le gain de performance obtenu lors de sa substitution. Nous pouvons également remarquer que tous les espaces de représentation disponibles pour la substitution des regroupements sont utilisés au moins une fois. Ceci signifie qu'ils permettent tous d'apporter une information pertinente pour la classification de graphèmes, même si lors de leurs évaluations individuelles ils ne conduisent pas à des taux de reconnaissance élevés.

Les performances obtenues par le système résultant de l'application de notre algorithme lors des différentes expériences réalisées sont présentées dans le tableau XV, en fonction de la valeur du seuil τ_1 utilisée. Le nombre de primitives non-discriminantes remplacées ainsi que celui de l'ensemble final ont également été inclus dans ce tableau. Les deux dernières lignes présentent les performances du système standard et celles du meilleur système construit lors de l'évaluation individuelle des différents espaces de représentation.

TABLEAU XV

Performances atteintes au second niveau perceptif lors de l'amélioration de l'ensemble de primitives perceptuelles.

τ_1	Card ($\bar{D}_1$)	Nb. primitives final	Taux de reconnaissance				
			10	100	1 000	5 000	10 000
25	7	188	99,3%	97,1%	92,5%	86,2%	82,7%
20	11	192	99,3%	96,9%	92,8%	87,3%	83,4%
10	18	247	99,4%	97,7%	93,8%	89,0%	85,9%
5	24	267	99,3%	97,6%	94,0%	89,1%	85,9%
0	27	300	99,4%	97,5%	94,3%	89,2%	86,2%
Standard		378	98,9%	95,7%	89,5%	82,2%	77,3%
DDD-4zH-apLDA		256	99,4%	97,7%	93,6%	88,6%	85,5%

La diminution de la valeur du seuil τ_1 conduit à une augmentation non négligeable de la performance du système final. Il est à noter que les performances présentées sont celles obtenues au second niveau perceptif. De la même manière que pour la première expérience ($\tau_1 = 25$), la descente au troisième niveau ne conduit pas à un gain significatif de perfor-

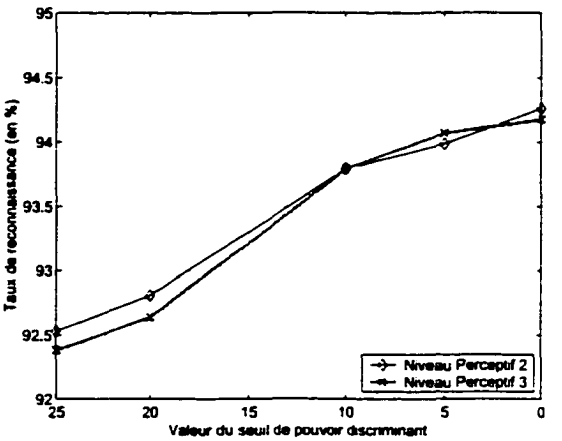


FIGURE 61 Performance globale aux niveaux perceptifs 2 et 3 en fonction de la variation du seuil de pouvoir discriminant τ_1.

mance, malgré l'intégration de plusieurs primitives. Ce phénomène peut être observé à la figure 61. La diminution du seuil de pouvoir discriminant entraîne la substitution de plus de primitives perceptuelles et donc rend possible l'amélioration de la classification d'un plus grand nombre de graphèmes, ce qui se traduit par un gain de performance.

La supériorité de la performance atteinte lors de l'application de notre algorithme avec un seuil de perplexité égal à 0 sur le meilleur ensemble de primitives (*DDD-4zH-apLDA*) traduit encore une fois l'intérêt de cette technique. En effet, même si toutes les primitives perceptuelles sont substituées, l'information qu'elles caractérisent est directement intégrée dans les primitives des niveaux suivants. De plus notre algorithme permet de choisir le meilleur espace de représentation pour un regroupement donné, ce qui est un point fort de cette approche.

Finalement nous présentons sur la figure 62 l'évolution des taux de reconnaissance en fonction du type d'écriture, pour les différents systèmes de reconnaissance résultants de l'application de l'algorithme d'intégration d'information. Les taux de reconnaissance atteints, pour un lexique de taille 1 000, sont de 97,3% pour les échantillons bâtons et 91,3% pour les cursifs. Lors de l'évaluation des différents espaces de représentation individuelle-

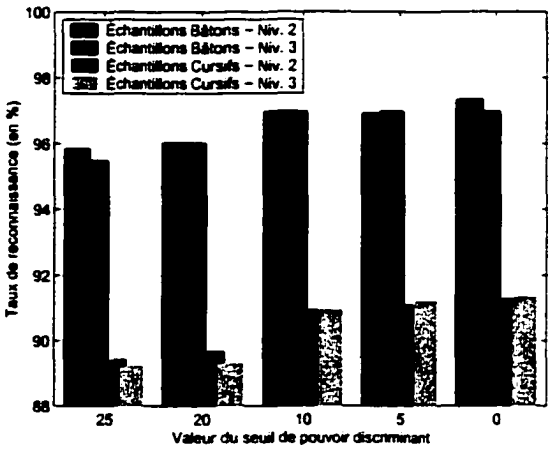


FIGURE 62 Performances pour les échantillons bâtons et cursifs aux niveaux perceptifs 2 et 3 en fonction de la variation du seuil de pouvoir discriminant τ_1.

ment ils étaient respectivement de 96,6% et 91,3% pour les échantillons bâtons et cursifs. Nous pouvons remarquer un gain uniquement pour les échantillons bâtons. Cependant il est à noter que lors de l'évaluation individuelle des espaces, deux systèmes distincts conduisent à la meilleure performance pour les deux styles d'écriture. En effet, le système *DDD-4zH-apLDA-256* conduisant au meilleur taux de reconnaissance global, ainsi que cursif, n'obtient qu'un taux de reconnaissance de 95,9% pour les échantillons bâtons. Cette remarque confirme encore une fois l'intérêt de l'approche proposée. La définition de regroupements de primitives permet en fait de diviser la tâche globale de reconnaissance en plusieurs sous-problèmes. La sélection de primitives réalisée indépendamment pour chaque regroupement permet de les spécialiser au sous-problème identifié et donc de combiner les différentes sources d'information disponibles d'une manière constructive.

5.2.3 Amélioration du meilleur ensemble de primitives

Afin de tester notre algorithme dans d'autres conditions, nous avons décidé de l'appliquer sur le meilleur ensemble de primitives obtenus lors de l'évaluation individuelle des différents espaces de représentation développés. Il s'agit de l'ensemble utilisant l'espace de

représentation DDD, la prise en compte de quatre zones avec division verticale de la zone médiane, l'application de l'algorithme LDA et contenant 256 primitives.

La valeur du seuil de pouvoir discriminant calculée à partir de l'équation 4.16 ($\tau_1 = 24,75$) permet d'isoler 22 primitives dans $\bar{D}_1$. L'étape d'analyse de ce sous-ensemble conduit à identifier 16 regroupements, six étant composés de deux primitives et dix d'une seule. Les espaces de représentation choisis pour la définition des nouvelles primitives sont ceux utilisés pour l'amélioration de l'ensemble perceptuel, moins *DDD-4zH-apLDA* qui caractérise le premier niveau perceptif.

De même que pour l'amélioration de l'ensemble perceptuel, la substitution individuelle des regroupements conduit toujours à un gain de performance. Il est parfois peu significatif, mais notre protocole expérimental étant rigoureusement le même pour chaque construction de système, il traduit tout de même un apport.

Les performances obtenues au premier et second niveau perceptif sont présentées dans le tableau XVI, globalement ainsi qu'en fonction du type d'écriture des échantillons.

TABLEAU XVI

Performances atteintes aux différents niveaux perceptifs lors de l'amélioration du meilleur ensemble de primitives.

	Niveau perceptif	Nb. primitives final	Taux de reconnaissance					
			Valid	10	100	1 000	5 000	10 000
Global	1	256	92,7%	99,4%	97,7%	93,6%	88,6%	85,5%
	2	308	93,0%	99,4%	97,2%	93,6%	88,7%	85,2%
Bâton	1	-	-	99,9%	99,0%	95,9%	92,2%	89,9%
	2	-	-	99,9%	98,8%	96,3%	92,8%	90,0%
Cursif	1	-	-	98,9%	96,5%	91,3%	84,7%	81,1%
	2	-	-	98,9%	95,6%	91,0%	84,6%	80,4%
Standard		378	88,4%	98,9%	95,7%	89,5%	82,2%	77,3%

Une constatation s'impose, l'application de notre algorithme ne permet pas une amélioration des performances dans ce cas. En effet, les taux de reconnaissance au second niveau

sont similaires à ceux obtenus avec l'ensemble de primitives de départ. L'analyse en fonction du type d'écriture montre qu'en fait un gain est présent pour les échantillons bâtons. Cependant il s'accompagne d'une perte pour les cursifs.

Cette remarque permet de mieux comprendre la non-amélioration des performances globales de l'ensemble de départ. En effet, l'analyse des performances des différents systèmes intermédiaires, ceux qui ont permis la sélection des nouvelles sources d'information, montre que la substitution de la grande majorité des regroupements conduit à une amélioration de la performance associée aux échantillons bâtons et parfois à une diminution de celle obtenue pour les échantillons cursifs. Une remarque importante à prendre en considération est que l'ensemble de départ est celui qui conduit au plus haut taux de reconnaissance sur les échantillons cursifs. Ces différentes constatations signifient que les différents espaces de représentation disponibles pour l'intégration ne permettent pas d'apporter une information complémentaire concernant les échantillons cursifs. La sélection des nouveaux ensembles de primitives se fait alors uniquement sur l'apport au niveau des échantillons bâtons. Le regroupement des différentes sources d'information au second niveau perceptif ne permet alors pas une augmentation des performances globales.

En conclusion, cette expérience met en évidence la limite des espaces de représentation développés. Étant donnée l'approche utilisée par notre algorithme, nous pouvons également énoncer que la représentation de l'information utilisée ici est trop précise. En effet, il vaut mieux utiliser, au premier niveau perceptif, un espace de représentation caractérisant une information moins précise, permettant ainsi de diviser le problème global de reconnaissance. Les niveaux suivants doivent alors permettre d'intégrer une information plus locale et/ou contextuelle. Les performances obtenues lors de l'amélioration de l'ensemble perceptuel permettent de confirmer cette proposition.

5.3 Conclusions et discussions

5.3.1 Conclusions sur les expériences réalisées

Dans ce chapitre, nous avons présenté les différentes expériences réalisées au cours de notre projet. La première section est consacrée à l'évaluation des différents espaces de représentation proposés dans la section 3.2, ainsi que des différentes techniques permettant d'accroître le pouvoir discriminant des primitives.

L'analyse des performances obtenues permet de constater l'intérêt d'utiliser l'algorithme LDA en particulier pour augmenter le pouvoir discriminant d'ensembles de primitives de petites tailles. Nous avons également constaté que la prise en compte des zones, au moment de l'extraction des caractéristiques, permet d'intégrer une information contextuelle qui facilite la reconnaissance des échantillons cursifs. L'évaluation de la stratégie de pondération proposée dans la section 3.3.4 a également été réalisée. Son application conduit bien au but recherché, à savoir la réduction de l'influence de la technique de *zoning* sur les échantillons bâtons. Par contre un effet de bord important, conduisant à une détérioration des performances sur les exemples cursifs est remarquable. De ce fait, nous conseillons de ne pas utiliser cette technique si le type de l'échantillon traité n'est pas connu *a priori*.

Cette phase expérimentale a également permis d'évaluer l'indicateur de pouvoir discriminant proposé dans la section 4.3.4.2.3. L'analyse des différentes valeurs obtenues montre qu'il peut être utilisé afin de sélectionner parmi plusieurs systèmes le meilleur, lorsque le seul paramètre variant est le nombre de primitives. Par contre la comparaison de systèmes construit à partir d'espaces de représentation différents manque de fiabilité dans l'état actuel de cet indicateur.

Dans la seconde partie de ce chapitre, nous avons exposé l'évaluation de notre algorithme d'intégration d'information. Son application sur l'ensemble de primitives perceptuelles a permis de montrer son efficacité. En effet le système résultant permet d'obtenir des taux

de reconnaissance nettement supérieurs à ceux du système standard, tout en utilisant moitié moins de primitives. Le gain est donc double puisque la diminution du nombre de paramètres conduit à un système plus fiable. De plus les performances atteintes sont également supérieures à celles du meilleur système obtenues lors de l'évaluation individuelle des espaces de représentation.

Finalement nous avons essayé d'améliorer la représentation de l'information fournie par ce dernier. L'expérience n'a pas été concluante puisque le système résultant ne permet que d'égaler les taux de reconnaissance atteints par le système de départ. Notre explication de ce phénomène est que l'ensemble de départ fourni une information trop précise au système et qu'il est ensuite difficile de l'améliorer par l'intermédiaire des espaces de représentation disponibles.

5.3.2 Discussions

Les différentes expériences réalisées ont permis de mettre en évidence l'intérêt de l'approche proposée dans cette thèse de doctorat. Après la détermination d'un sous-ensemble de primitives non-discriminantes, un point fort de notre approche est l'identification de regroupements de primitives. En effet cette étape permet de diviser la tâche de reconnaissance en plusieurs sous-problèmes. Cependant notre dernière expérience permet de préciser qu'il faut utiliser au premier niveau de représentation un ensemble de primitives apportant une représentation globale. La définition et la sélection automatique de primitives sont alors réalisées en fonction de chaque sous-problème identifié par les différents regroupements. Cette approche revient à utiliser une représentation hiérarchisée de l'information où celle caractérisée par les primitives des niveaux supérieurs est incluse implicitement dans les primitives des niveaux inférieurs.

Un point fort de notre approche est son fonctionnement automatique. En effet les étapes importantes d'identification des regroupements et de sélection des nouvelles primitives sont effectuées de manière autonome par l'algorithme. De ce fait l'intégration d'une nou-

velle source d'information ou l'utilisation de nouvelles bases de données ne nécessitent que la mise en œuvre d'un apprentissage. Ce dernier permettra d'obtenir la hiérarchie de caractéristiques pour les différents espaces de représentation disponibles et la base de données fournie. De plus la structure d'information construite permet d'optimiser l'extraction de caractéristiques lors du fonctionnement du système final en mode production.

Il est certain que différents points de notre algorithme sont perfectibles. La détermination du seuil de pouvoir discriminant permettant d'identifier les primitives non-discriminantes peut être améliorée. À notre avis, au premier niveau, il faut conserver deux stratégies suivant la nature de l'espace de représentation. Lors de la sélection des nouvelles primitives, nous construisons un grand nombre d'ensembles de primitives par quantification vectorielle. Étant donné l'algorithme utilisé, il est fort peu probable que l'ensemble sélectionné contienne le nombre optimal de primitives pour le sous-problème visé. Le développement ou l'utilisation d'un algorithme de quantification vectorielle permettant la détermination de ce nombre permettrait certainement une amélioration des performances et un gain en temps de calcul également.

Tout au long de cette thèse, nous avons bien mis en évidence la différence de comportement de notre système vis-à-vis du style d'écriture des échantillons : bâton ou cursif. En effet le premier est globalement plus facile à reconnaître que le second, comme le prouvent les taux de reconnaissance obtenus. Les différentes stratégies proposées pour améliorer un des deux styles conduisent généralement à une détérioration de la performance associée à l'autre. Il serait donc très intéressant de développer un ensemble de primitives permettant d'identifier le style d'origine des graphèmes. Ce dernier serait alors utilisé au premier niveau perceptif et ainsi permettrait d'outrepasser le problème associé au style d'écriture.

Nous pensons qu'il est encore possible d'augmenter les performances du système de reconnaissance, en intégrant d'autres sources d'information. En particulier, avant d'utiliser des primitives CCV ou DDD, il est certainement possible d'obtenir une information in-

termédiaire de manière à mieux identifier les sous-problèmes de reconnaissance. En particulier, nous n'avons pas envisagé la recherche d'information structurelle au sein des graphèmes. L'extraction de particularités locales ou l'intersection des graphèmes avec des droites devrait permettre d'apporter une information complémentaire.

Finalement notre modélisation prend en compte un ensemble de cinq primitives afin de caractériser les points de segmentation. Il est envisageable également d'améliorer ce dernier. En effet l'intégration d'une information plus précise au niveau des points de ligature entre graphèmes permettrait d'améliorer l'apport d'information contextuelle et certainement les performances de notre système de reconnaissance.

CONCLUSION

Le but de ce travail était l'amélioration d'un système de reconnaissance industriel de l'écriture manuscrite. L'analyse de l'existant nous a permis de mettre en évidence sa principale faiblesse : la représentation de l'information dont dispose le système pour effectuer sa tâche de reconnaissance. Elle est caractérisée par deux ensembles de primitives, complémentaires mais de faible pouvoir discriminant.

Notre première proposition pour pallier ce problème a été la définition de nouveaux espaces de représentation afin d'extraire de nouvelles sources d'information. Différentes stratégies ont également été développées dans le but d'accroître le pouvoir discriminant des primitives résultantes. Plusieurs systèmes de reconnaissance ont été construits à partir de différents ensembles de primitives. L'évaluation de leurs performances a permis de mettre en évidence l'adéquation des espaces de représentation développés, pour la reconnaissance de l'écriture manuscrite. Les différentes expériences réalisées ont également prouvé la pertinence des techniques permettant l'amélioration du pouvoir discriminant des primitives.

Afin d'intégrer ces nouvelles sources d'information dans le système de reconnaissance de base, nous avons développé un nouvel algorithme. Ce dernier permet la définition et la sélection dynamique d'ensembles de primitives pour remplacer les moins discriminantes d'un ensemble de départ. Son application a permis d'intégrer de manière pertinente plusieurs sources d'information et ainsi améliorer la représentation de l'information mise à la disposition du système de reconnaissance. Cette action s'est bien sûr traduite par une augmentation significative des performances par rapport au système de base, accompagnée d'une réduction du nombre de primitives.

Nos contributions

Au niveau de l'extraction de caractéristiques, plusieurs contributions peuvent être recensées. Premièrement nous avons appliqué la technique d'extraction de caractéristiques DDD à la reconnaissance de mots manuscrits. Jusqu'alors elle n'avait été appliquée qu'a la reconnaissance de caractères isolés. Nous avons montré qu'elle a un bon pouvoir discriminant mais uniquement suite à l'application d'une analyse discriminante linéaire. De plus, cet espace de représentation, associé à une stratégie de *zoning*, conduit à la construction du système de reconnaissance obtenant la meilleure performance sur l'ensemble des échantillons de notre corpus de test, avant application de notre algorithme de sélection.

Notre seconde contribution est une stratégie permettant la mise en œuvre de l'algorithme LDA dans un système de reconnaissance utilisant les modèles de Markov cachés. Elle consiste à utiliser un premier système de reconnaissance de manière à réaliser l'étiquetage des graphèmes par leur classe. Cette action est possible par l'intermédiaire de la procédure de *backtracking* de l'algorithme de Viterbi. L'étiquette permet alors d'effectuer une analyse discriminante linéaire de l'espace de représentation des données.

L'utilisation de la technique de *zoning* lors de l'extraction de caractéristiques permet de prendre en compte une information contextuelle locale. Ceci est particulièrement bénéfique pour la reconnaissance des échantillons cursifs. Par contre cette technique conduit à une perte de performance pour les exemples bâtons. Nous avons alors proposé une nouvelle stratégie de pondération des zones afin d'atténuer cette diminution de performance. Elle permet de prendre en compte d'une certaine manière le style d'écriture de l'échantillon au niveau de la définition des primitives.

La principale contribution de notre travail est bien sûr le développement d'un nouvel algorithme permettant la sélection dynamique de caractéristiques. L'approche proposée est simple et innovante. Elle consiste à remplacer les primitives les moins discriminantes par de nouvelles, plus pertinentes pour la tâche de reconnaissance visée.

Le concept de regroupements de primitives est le point clé de cette innovation. Nous avons défini un regroupement de primitives comme un sous-ensemble de primitives d'un ensemble donné dont tous les éléments partagent certaines propriétés. Pour la mise en œuvre de notre algorithme, la propriété évaluée pour regrouper deux primitives est la distance entre leurs distributions de probabilités conditionnelles sur les classes de modélisation. Si les distributions sont similaires pour deux primitives, cela signifie qu'elles sont utilisées pour caractériser les mêmes classes. Leur regroupement permet d'identifier un sous-problème de reconnaissance, puisque le nombre de classes concernées pour la classification des graphèmes, associés aux primitives concernées, est réduit par rapport au nombre total de classes. La définition et la sélection du meilleur ensemble de primitives permettant la substitution d'un regroupement conduit à une spécialisation de cet ensemble au sous-problème identifié. La prise en compte de ces regroupements permet une intégration pertinente des sources d'information disponibles.

Une autre contribution est la définition d'un indicateur du pouvoir de généralisation d'un système de reconnaissance utilisant les modèles de Markov cachés. Il permet d'obtenir cette information à la fin de la phase d'apprentissage sans avoir recours à l'évaluation de la performance du système. De plus son utilisation rend possible la comparaison directe de systèmes utilisant des ensembles de primitives de taille différente.

Perspectives

Le développement de cette thèse de doctorat a permis d'atteindre l'objectif visé : améliorer les performances du système de reconnaissance de l'écriture manuscrite du SRTP. Cependant il est certain que la solution proposée n'est pas optimale et donc que certaines améliorations sont possibles :

- *Multiplication des sources d'information disponibles* : la technique de sélection proposée fonctionne correctement. La disponibilité d'un plus grand nombre de sources d'information complémentaires à celles déjà disponibles doit permettre une aug-

mentation des performances. En particulier la définition d'un ensemble de primitives permettant de déterminer localement le style des échantillons (bâton/cursif) sera très bénéfique.

- *Amélioration de l'ensemble de primitives associé aux points de segmentation* : ce dernier permet d'intégrer une information sur les ligatures entre graphèmes, c'est une information contextuelle importante. Nous pouvons envisager l'amélioration de ce dernier de la même manière que ceux caractérisant les graphèmes.
- *Prise en compte de la corrélation entre classes* : différentes observations prouvent que certaines classes de la modélisation sont liées ou corrélées. L'obtention d'ensembles de primitives très discriminants ne contenant que 64 primitives en est une. Un travail d'analyse doit permettre de quantifier ce phénomène. Sa prise en compte pourrait intervenir lors de la définition des regroupements de primitives sous la forme d'un facteur de pondération diminuant l'influence des distances associées à des classes corrélées. Cette stratégie doit permettre une meilleure définition des regroupements.
- *Amélioration du processus de quantification vectorielle* : l'algorithme mis en œuvre au cours de notre travail n'est pas optimal. L'utilisation d'un algorithme de quantification vectorielle permettant de déterminer le nombre optimal de primitives permettrait d'accélérer le processus de sélection de primitives. Le travail effectué sur la corrélation entre les classes pourrait être pris en compte dans un tel algorithme puisque la définition d'un ensemble de primitives optimal est dépendant de la modélisation utilisée.
- *Combinaison d'ensembles de primitives améliorés* : l'amélioration de plusieurs ensembles de primitives peut être réalisée de manière indépendante. En considérant les deux ensembles statistiquement indépendants, une autre approche de combinaison peut être réalisée directement au niveau des modèles, en effectuant le produit des probabilités associés à chaque ensemble. Une application de cette stratégie consiste à effectuer l'amélioration des ensembles bâton et perceptuel de manière indépen-

dante et en utilisant uniquement les échantillons bâtons et cursifs respectivement. Cela devrait permettre d'obtenir un ensemble spécialisé par type d'écriture. Leur combinaison dans le modèle final devrait conduire à de bonnes performances.

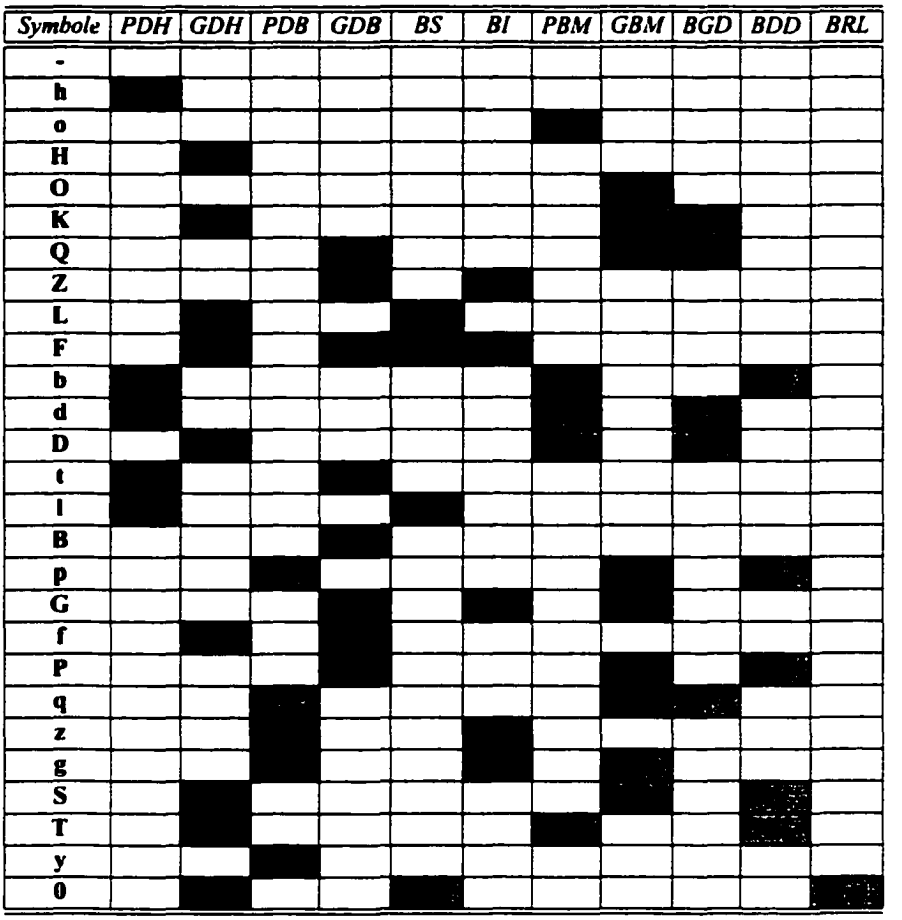
ANNEXE 1

DESCRIPTION DES ENSEMBLES DE PRIMITIVES

Les différents ensembles de primitives utilisés par le système de base ont été définis par A. El-Yacoubi au cours de son travail doctoral [38]. Deux ensembles permettent de caractériser les graphèmes de l'écriture. Le troisième, ne contenant que cinq symboles caractérise les points de ligature entre deux graphèmes consécutifs.

1.1 Les primitives perceptuelles

Les colonnes du tableau correspondent aux caractéristiques de base recherchées dans l'image. Les différentes lignes correspondent aux 27 primitives perceptuelles.



PDH : petit dépassement haut
GDH : grand dépassement haut
PDB : petit dépassement bas
GDB : grand dépassement bas
BS : boucle supérieure
BI : boucle inférieure
PBM : petite boucle médiane
GBM : grande boucle médiane
BGD : boucle à gauche du dépassement
BDD : boucle à droite du dépassement
BRL : boucle reposant sur la ligne de base

1.2 Les primitives bâtons

Les primitives bâtons sont extraites à partir des histogrammes de transitions horizontal et vertical.

Symbole	*(h,v)*	*Correspond aux formes semblables aux caractères*
- |	2,2	I L 1
r	2,4	r n
m	2,6	m
> F C	4,2	C F J 7
O A P	4,4	A O D P 6 9
W	4,6	oi or
E	6,2	E Z S 3 5
B	6,4	B Q 8
Q	6,6	Q

h : nombre de transitions dominant pour l'histogramme horizontal
v : nombre de transitions dominant pour l'histogramme vertical

1.3 Les primitives codant les points de segmentation

Symbole	*Nature du point de segmentation*
s	Point de segmentation engendré par l'algorithme et situé à proximité de la ligne de base
u	Point de segmentation engendré par l'algorithme et situé un peu loin de la ligne de base
n	Point de segmentation naturel non suivi d'un espace
@	Point de segmentation naturel suivi d'un petit espace
#	Point de segmentation naturel suivi d'un grand espace

ANNEXE 2

DÉTAIL DE LA SEGMENTATION DES CARACTÈRES

Dans cette annexe nous présentons le nombre d'occurrences des différents caractères modélisés, présents dans le corpus d'apprentissage (voir section 2.3.8). Pour chacun nous avons également indiqué les pourcentages associés aux différentes segmentation en graphèmes possibles. Ces données ont été récoltées lors de la vérification par un opérateur humain des alignements caractères/graphèmes (voir section 2.4.3) pour les 10 000 premiers échantillons de notre corpus d'apprentissage [32].

Caractère	Nombre d'occurrences	Pourcentage de segmentation en n morceaux					
		$n=0$	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$
A	3 974	0,33	86,59	12,48	0,58	0,03	0
B	1 645	0,06	81,95	15,50	2,31	0,18	0
C	3 835	0,57	90,35	8,40	0,65	0,03	0
D	1 273	1,41	78,87	18,07	1,57	0,08	0
E	6 264	2,99	81,32	14,81	0,83	0,05	0
F	454	3,74	75,33	19,38	1,54	0	0
G	982	0,20	29,53	67,31	2,65	0,31	0
H	770	0,26	4,16	91,04	4,29	0,26	0
I	2 587	7,58	91,30	1,04	0,08	0	0
J	213	0,47	74,65	24,88	0	0	0
K	12	0	16,67	83,33	0	0	0
L	3 764	0,80	79,94	18,57	0,69	0	0
M	1 817	0,11	24,93	63,51	10,35	1,05	0,06
N	3 807	0,18	8,56	86,81	4,33	0,11	0
O	2 728	2,64	88,38	8,94	0,04	0	0
P	999	1,90	87,09	10,71	0,30	0	0
Q	123	0	69,92	18,70	10,57	0,81	0
R	3 710	0,40	84,69	14,07	0,78	0,05	0
S	3 998	1,98	86,24	11,63	0,13	0,03	0
T	2 098	8,63	79,93	10,58	0,86	0	0
U	1 790	0,34	12,96	85,81	0,89	0	0
V	1 153	0,17	23,76	75,63	0,43	0	0
W	29	0	0	24,14	65,52	10,34	0

Suite du tableau page suivante...

Caractère	Nombre d'occurrences	Pourcentage de segmentation en n morceaux					
		$n=0$	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$
X	799	0,75	27,28	71,34	0,63	0	0
Y	680	0,44	25,29	73,38	0,88	0	0
Z	221	0,90	91,86	7,24	0	0	0
a	3 746	1,71	75,92	21,46	0,91	0	0
b	528	0,19	46,78	51,52	1,52	0	0
c	1 252	1,36	94,17	4,31	0,16	0	0
d	2 111	0,28	45,29	51,35	2,98	0,09	0
e	9 740	4,48	90,97	4,33	0,21	0,02	0
f	136	2,21	92,65	5,15	0	0	0
g	739	0,14	70,37	28,28	1,22	0	0
h	670	0,75	34,03	62,54	2,69	0	0
i	3 249	9,33	88,09	2,59	0	0	0
j	76	1,32	93,42	5,26	0	0	0
k	15	0	60,00	40,00	0	0	0
l	3 074	5,07	91,57	3,16	0,20	0	0
m	613	0,33	4,40	26,92	66,23	1,79	0,33
n	3 971	0,96	27,30	68,32	3,40	0,03	0
o	3 097	3,42	85,28	11,14	0,16	0	0
p	297	0	62,63	36,36	1,01	0	0
q	79	0	63,29	36,71	0	0	0
r	3 515	3,70	83,27	12,57	0,46	0	0
s	2 909	3,27	92,30	4,40	0,03	0	0
t	2 064	12,40	68,94	18,12	0,48	0	0
u	2 367	0,46	8,45	87,45	3,63	0,04	0
v	531	0,19	24,86	74,01	0,94	0	0
w	6	0	0	16,67	66,67	16,67	0
x	1 786	0,84	33,82	61,31	3,92	0,11	0
y	702	0,43	14,81	82,19	2,56	0	0
z	190	2,11	82,11	15,79	0	0	0
espace	7571	7,71	92,29	0	0	0	0

Suite du tableau page suivante...

Caractère	Nombre d'occurrences	Pourcentage de segmentation en n morceaux					
		$n = 0$	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$
0	120	4,17	86,67	9,17	0	0	0
1	137	3,65	92,70	3,65	0	0	0
2	99	1,01	61,62	35,35	2,02	0	0
3	26	0	84,62	15,38	0	0	0
4	5	0	20,00	80,00	0	0	0
5	9	0	77,78	22,22	0	0	0
6	15	0	66,67	33,33	0	0	0
7	11	0	63,64	36,36	0	0	0
8	11	0	100,00	0	0	0	0
9	70	0	87,14	12,86	0	0	0
-	5	60,00	40,00	0	0	0	0
\	135	94,81	4,44	0,74	0	0	0
/	255	2,75	95,69	1,57	0	0	0
.	17	94,12	5,88	0	0	0	0
,	0	0	0	0	0	0	0
'	0	0	0	0	0	0	0

Fin du tableau

ANNEXE 3

PERFORMANCES DÉTAILLÉES DU SYSTÈME STANDARD

Dans cette annexe, nous présentons les performances du système de reconnaissance standard du SRTP présenté dans le chapitre 2. L'évaluation a été réalisée en utilisant d'abord chaque ensemble de primitives individuellement puis les deux conjointement. Pour les trois systèmes construits, les corpus de données décrit dans la section 2.3.8 ont été utilisés. Lors des différents tests, le lexique global présenté dans la section 2.3.9 a servi au tirage aléatoire des différents noms de commune.

Les taux de reconnaissance ont été mesurés globalement pour l'ensemble du corpus de test ainsi que pour chaque type d'écriture. Dans les différents tableaux de cette annexe, TR(x) signifie que l'intitulé exact de l'exemple testé fait partie des x premières solutions proposées par le système de reconnaissance.

3.1 Évaluation de l'ensemble de primitives perceptuelles

Les primitives perceptuelles (voir section 2.3.3.1) sont basées sur la détection des dépassements et des boucles. L'ensemble contient 27 primitives distinctes. Elles servent plus particulièrement à caractériser l'écriture cursive.

		TR(1)	TR(2)	TR(3)	TR(4)	TR(5)	TR(10)
Lexique 10	Global	96,90	99,04	99,55	99,81	99,94	100
	Bâton	97,03	99,35	99,82	99,88	100	100
	Cursif	96,52	98,77	99,30	99,75	99,88	100
	Mixte	98,18	99,27	99,82	99,82	100	100
Lexique 100	Global	89,60	94,27	96,17	96,92	97,39	98,65
	Bâton	88,12	94,30	96,20	97,09	97,68	98,93
	Cursif	89,67	93,65	95,66	96,48	96,93	98,28
	Mixte	93,82	96,91	98,36	98,36	98,55	99,45
Lexique 1000	Global	74,99	82,97	86,22	88,13	89,28	92,79
	Bâton	70,90	80,17	84,09	86,64	87,95	92,52
	Cursif	75,70	83,32	86,19	87,79	88,93	92,21
	Mixte	84,36	90,00	92,91	94,18	94,91	96,18
Lexique 5000	Global	62,22	71,18	75,22	77,81	79,53	84,68
	Bâton	54,99	65,74	70,61	73,46	75,65	82,24
	Cursif	64,92	72,83	76,52	79,06	80,45	84,80
	Mixte	72,36	80,55	83,64	85,64	87,27	91,64
Lexique 10 000	Global	56,85	65,94	69,92	72,74	74,60	80,15
	Bâton	49,70	58,91	63,48	66,92	69,66	76,01
	Cursif	59,43	68,28	71,80	74,43	75,86	81,15
	Mixte	67,27	77,09	81,27	83,09	84,18	88,36

3.2 Évaluation de l'ensemble de primitives bâtons

Les primitives bâtons sont basées sur l'analyse des histogrammes de transitions horizontaux et verticaux des graphèmes. L'ensemble contient 14 symboles. Comme leur nom l'indique, elles servent à caractériser l'écriture bâton.

		TR(1)	TR(2)	TR(3)	TR(4)	TR(5)	TR(10)
Lexique 10	Global	97,67	99,23	99,59	99,79	99,94	100
	Bâton	99,35	99,82	99,88	100	100	100
	Cursif	96,23	98,73	99,34	99,59	99,88	100
	Mixte	98,91	99,64	99,82	100	100	100
Lexique 100	Global	91,51	94,97	96,13	96,92	97,63	98,82
	Bâton	97,33	98,52	98,99	99,29	99,58	99,76
	Cursif	86,23	91,72	93,61	94,88	95,98	98,07
	Mixte	97,09	98,55	98,55	98,73	98,91	99,27
Lexique 1000	Global	81,07	86,74	89,09	90,24	91,23	93,95
	Bâton	91,15	94,89	96,32	96,56	97,03	97,98
	Cursif	72,13	79,51	82,70	84,59	86,07	90,20
	Mixte	89,82	93,82	95,27	96,00	96,36	98,18
Lexique 5000	Global	71,78	78,46	81,73	83,40	84,25	87,89
	Bâton	84,92	90,62	92,46	93,29	93,94	95,84
	Cursif	60,25	68,11	72,38	74,63	75,70	80,66
	Mixte	82,73	87,09	90,36	92,00	92,55	95,64
Lexique 10 000	Global	66,28	74,28	77,26	79,50	80,89	84,94
	Bâton	79,57	87,41	89,49	90,74	91,63	94,30
	Cursif	54,06	62,66	66,48	69,59	71,27	76,64
	Mixte	79,82	85,64	87,64	89,09	90,73	93,09

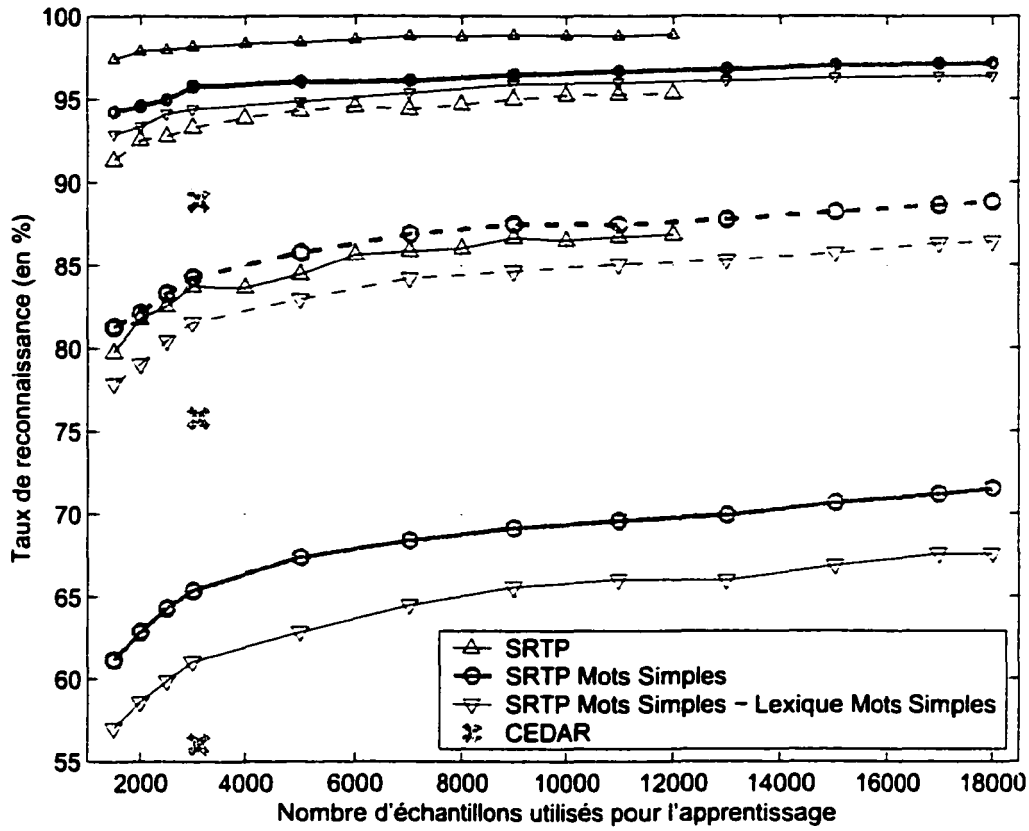
3.3 Évaluation des deux ensembles de primitives conjointement

Le système standard SRTP utilise les ensembles de primitives perceptuelles et bâtons conjointement. Cette technique revient à considérer toutes les combinaisons possibles de deux primitives, une de chaque ensemble. Cela équivaut à utiliser un seul ensemble de primitives de cardinalité égale au produit du nombre de primitives composant chaque ensemble, en l'occurrence $27 \times 14 = 378$.

		TR(1)	TR(2)	TR(3)	TR(4)	TR(5)	TR(10)
Lexique 10	Global	98,87	99,61	99,85	99,96	99,96	100
	Bâton	99,47	99,82	100	100	100	100
	Cursif	98,28	99,43	99,71	99,92	99,92	100
	Mixte	99,64	99,82	100	100	100	100
Lexique 100	Global	95,66	97,67	98,35	98,72	98,91	99,36
	Bâton	97,92	98,93	99,29	99,41	99,53	99,82
	Cursif	93,57	96,52	97,54	98,11	98,40	99,02
	Mixte	98,00	98,91	99,09	99,27	99,27	99,45
Lexique 1000	Global	89,48	93,33	94,97	95,64	96,26	97,30
	Bâton	93,35	96,08	97,09	97,69	98,10	98,52
	Cursif	85,41	90,49	92,87	93,69	94,55	96,15
	Mixte	95,64	97,45	97,82	98,00	98,18	98,73
Lexique 5000	Global	82,18	87,55	89,67	91,04	91,91	94,20
	Bâton	86,94	91,87	93,65	94,66	95,43	96,62
	Cursif	77,09	83,07	85,70	87,46	88,44	91,68
	Mixte	90,18	94,18	95,09	95,82	96,55	98,00
Lexique 10 000	Global	77,30	84,19	86,55	88,00	89,37	92,02
	Bâton	81,96	89,08	91,10	92,05	93,18	95,37
	Cursif	72,34	79,39	82,01	83,93	85,57	88,77
	Mixte	85,09	90,55	92,73	93,64	94,55	96,18

3.4 Influence de différents paramètres sur la performance

Dans la section 2.4.2, nous avons évalué l'influence de différents paramètres sur les performances du système. La figure suivante, similaire à la figure 25, présente les taux de reconnaissance obtenus pour des lexiques de taille 10, 100 et 1 000. Les taux de reconnaissance correspondant aux expériences avec un lexique de taille 10 sont représentés par des indicateurs de plus petites tailles que les autres. Des segments discontinus sont utilisés pour relier les points des expériences avec un lexique de taille 100.



BIBLIOGRAPHIE

[1] D.W. Aha and R.L. Bankert. A comparative evaluation of sequential feature selection algorithms. In *Proc. of the 5th International Workshop on Artificial Intelligence and Statistics*, pages 1–7, Ft. Lauderdale, USA, 1995.

[2] L.A. Alexandre, A.C. Campilho, and M. Kamel. On combining classifiers using sum and product rules. *Pattern Recognition Letter*, 22 :1283–1289, 2001.

[3] H. Almuallim and T.G. Dietterich. Learning with many irrelevant features. In *Proc. of the 9th National Conference on Artificial Intelligence*, volume 2, pages 547–552, Anaheim, USA, 1991. AAAI Press.

[4] H. Almuallim and T.G. Dietterich. Efficient algorithms for identifying relevant features. In *Proc. of the 9th Canadian Conference on Artificial Intelligence*, pages 38–45, Vancouver, Canada, 1992. Morgan Kaufmann.

[5] H.C. Andrews. Multidimensional rotations in feature selection. *IEEE Trans. on Computers*, 20 :1045–1051, September 1971.

[6] Emmanuel Augustin. *Reconnaissance de mots manuscrits par systèmes hybrides Réseaux de Neurones et Modèles de Markov Cachés*. PhD thesis, Université René Descartes - Paris V, 2001. 188 pages.

[7] L.R. Bahl, F. Jelinek, and R.L. Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 5(2) :179–190, 1983.

[8] R.R. Bailey and M. Srinath. Orthogonal moment features for use with parametric and non-parametric classifiers. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 18(4) :389–398, April 1996.

[9] A. Baraldi and P. Blonda. A survey of fuzzy clustering algorithms for pattern recognition. ii. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 29(6) :786–801, Dec. 1999.

[10] Gérard Battail. *Théorie de l'information - Application aux techniques de communication*. Masson, Paris, France, 1997.

[11] L.E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, 37 :1554–1563, 1966.

[12] S.O. Belkasim, M. Shridhar, and M. Ahmadi. Pattern recognition with moment invariants : A comparative study and new results. *Pattern Recognition*, 24(12) :1117–1138, 1991.

[13] R.M. Bozinovic and S.N. Srihari. Off-line cursive script word recognition. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 11(1) :68–83, 1989.

[14] Gilles Brassard and Paul Bratley. *Fundamentals of Algorithmics*. Prentice Hall, New Jersey, USA, 1996.

[15] C. Bron and J. Kerbosh. Finding all cliques of an undirected graph. *Communications of the ACM*, 16(9) :575–577, 1973.

[16] H. Bunke, M. Roth, and E.G. Schukat-Talamazzini. Off-line cursive handwriting recognition using hidden Markov models. *Pattern Recognition*, 28(9) :1399–1413, 1995.

[17] R. Buse, Z.Q. Liu, and T. Caelli. A structural and relational approach to handwritten word recognition. *IEEE Trans. on Systems, Man and Cybernetics - Part B*, 27(5) :847–861, 1997.

[18] C. Cardie. Using decision trees to improve case-based learning. In *Proc. of the 10th International Conference on Machine Learning*, pages 25–32, Amherst, USA, 1993. Morgan Kaufmann.

[19] O. Castillo, R. Cardona, and P. Melin. A hybrid approach for automated quality control combining learning vector quantization neural networks and fuzzy logic. In *Proc. of the 2002 International Joint Conference on Neural Networks*, volume 3, pages 2081–2085, Honolulu, Hawaii, May 2002.

[20] G. Chen and T.D. Bui. Invariant Fourier-wavelet descriptor for pattern recognition. *Pattern Recognition*, 32 :1083–1088, 1999.

[21] K. Chen and H. Liu. Towards an evolutionary algorithm : A comparison of two feature selection algorithms. In *Proc. of the 1999 Congress on Evolutionary Computation*, pages 1309–1313, Piscataway, USA, 1999.

[22] M.Y. Chen, A. Kundu, and J. Sargur. Off-line handwritten word recognition using a hidden Markov model type stochastic network. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 16(5) :481–496, 1994.

[23] M.Y. Chen, A. Kundu, and J. Zhou. Variable duration hidden Markov model and morphological segmentation for handwritten word. *IEEE Trans. on Image Processing*, 4(12) :1675–1687, 1995.

[24] Y.-C. Chim, A.A. Kassim, and Y. Ibrahim. Dual classifier system for handprinted alphanumeric character recognition. *Pattern Analysis and Application*, 1 :155–162, 1998.

[25] E. Cohen, J.J. Hull, and S.N. Srihari. Control structure for interpreting handwritten addresses. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 16(10) :1049–1055, 1994.

[26] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, New York, USA, 1991.

[27] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1 :131–156, 1997.

[28] J.C.W. Debuse and V.J. Rayward-Smith. Feature subset selection within a simulated annealing data mining algorithm. *Journal of Intelligent Information Systems*, 9(1) :57–81, 1997.

[29] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B (Methodological)*, 39(1) :1–39, 1977.

[30] G. Dimauro, S. Impedovo, G. Pirlo, and A. Salzo. An advanced segmentation technique for cursive word recognition. In *Proc. of the 6th International Workshop on Frontiers in Handwriting Recognition*, pages 99–111, Taejon, Korea, 1998.

[31] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. John Wiley & Sons, New York, USA, 2001. 2nd edition.

[32] Hugues Duplantier. Interfaces de visualisation pour la reconnaissance d'écriture. Technical report, Ecole Polytechnique, 1998. Rapport de stage d'Option Scientifique.

[33] G. Dzuba, A. Filatov, D. Gershuny, and I. Kil. Handwritten word recognition - the approach proved by practice. In *Proc. of the 6th International Workshop on Frontiers in Handwriting Recognition*, pages 99–111, 1998.

[34] A. El-Yacoubi, J.-M. Bertille, and M. Gilloux. Towards a more effective handwritten word recognition system. In *Proc. of the 4th International Workshop on Frontiers in Handwriting Recognition*, pages 378–385, Taipei, Taiwan, December 7-9 1994.

[35] A. El-Yacoubi, M. Gilloux, R. Sabourin, and C.Y. Suen. Objective evaluation of the discriminant power of features in an HMM-based word recognition system. In *Proc. of the 1st Brazilian Symposium of Document Image Analysis*, pages 60–73, Curritiba, Brazil, November 1997.

[36] A. El-Yacoubi, M. Gilloux, R. Sabourin, and C.Y. Suen. An HMM-based approach for off-line unconstrained handwritten word modeling and recognition. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 21(8) :752–760, 1999.

[37] A. El-Yacoubi, R. Sabourin, M. Gilloux, and C.Y. Suen. Improved model architecture and training phase in an off-line HMM-based word recognition system. In *Proc. of the 13th International Conference on Pattern Recognition*, pages 1521–1525, Brisbane, Australia, August 16-20 1998.

[38] Abdenaïm El-Yacoubi. *Modélisation Markovienne de l'écriture manuscrite - Application à la reconnaissance des adresses postales*. PhD thesis, Université de Rennes I, 1996. 307 pages.

[39] Charky Farouz. *Reconnaissance de mots manuscrits hors-ligne dans un vocabulaire ouvert par modélisation markovienne*. PhD thesis, Université de Nantes, 1999. 218 pages.

[40] J.T. Favata, G. Srikantan, and S.N. Srihari. Handprinted character/digit recognition using a multiple feature/resolution philosophy. In *Proc. of the 4th International Workshop on Frontiers in Handwriting Recognition*, pages 57–66, Taipei, Taiwan, December 7-9 1994.

[41] A. Filatov, N. Nikitin, A. Volgunin, and P. Zelinsky. The *Address*Script™ recognition system for handwritten envelopes. In *International Association for Pattern Recognition Workshop on Document Analysis Systems (DAS'98)*, pages 157–171, Nagano, Japan, November 4-6 1998.

[42] G.D. Forney. The Viterbi algorithm. *Proc. of the IEEE*, 61(3) :268–278, March 1973.

[43] I. Foroutan and J. Sklansky. Feature selection for automatic classification of non-Gaussian data. *IEEE Trans. on Systems, Man and Cybernetics*, 17(2) :187–198, 1987.

[44] K. Fukunaga. *Introduction to Statistical Pattern Recognition, Second Edition*. Academic Press, New York, 1990.

[45] P. Gader, B. Forester, M. Ganzberger, A. Gillies, B. Mitchell, M. Whalen, and T. Yocum. Recognition of handwritten digits using template and model matching. *Pattern Recognition*, 24(5) :421–431, 1991.

[46] P. Gader, M. Whalem, M. Ganzberger, and D. Hepp. Hand-printed word recognition on a NIST data set. *Machine Vision and Applications*, 8 :31–40, 1995.

[47] P.D. Gader, M. Mohamed, and J.-H. Chiang. Handwritten word recognition with character and inter-character neural networks. *IEEE Trans. on Systems, Man and Cybernetics - Part B*, 27(1) :158–164, 1997.

[48] P.D. Gader, M.A. Mohamed, and J.M. Keller. Fusion of handwritten word classifiers. *Pattern Recognition Letter*, 17 :577–584, 1996.

[49] G. Gaillat and M. Berthod. Panorama des techniques d'extraction de traits caractéristiques en lecture optique des caractères. *Revue Technique THOMSON-CSF*, 11(2) :943–959, 1979.

[50] R.G. Gallager. Claude E. Shannon : A retrospective on his life, work, and impact. *IEEE Trans. on Information Theory*, 47(7) :2681–2695, November 2001.

[51] A.M. Gillies. Cursive word recognition using hidden Markov models. In *Proc. of the 5th USPS Advanced Technology Conference*, pages 557–562, Washington, D.C., 1992.

[52] M. Gilloux and M. Leroux. Recognition of cursive script amounts on postal cheques. In *Proc. of the 5th USPS Advance Technology Conference*, pages 545–556, 1992.

[53] M. Gilloux, M. Leroux, and J.-M. Bertille. Strategies for cursive script recognition using hidden Markov models. *Machine Vision and Applications*, 8 :197–205, 1995.

[54] S. Günter and H. Bunke. Creation of classifier ensembles for handwritten word recognition using feature selection algorithms. In *Proc. of the 8th International Workshop on Frontiers in Handwriting Recognition*, pages 183–188, Niagara-on-the Lake, Canada, August 6-8 2002.

[55] I.J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3, 4) :237–264, 1953.

[56] V.K. Goyal, J.A. Kelner, and J. Kovacevic. Multiple description vector quantization with a coarse lattice. *IEEE Transactions on Information Theory*, 48(3) :781–788, March 2002.

[57] F. Grandidier. Rapport sur l'évaluation du système de reconnaissance de l'écriture du SRTP. Technical report, Ecole de Technologie Supérieure, Août 1998.

[58] F. Grandidier, R. Sabourin, A. El-Yacoubi, C.Y. Suen, and M. Gilloux. Influence of word length on handwriting recognition. In *Proc. of the 5rd International Conference on Document Analysis and Recognition*, pages 777–780, Bangalore, India, September 20-22, 1999.

[59] F. Grandidier, R. Sabourin, M. Gilloux, and C.Y. Suen. An *a priori* indicator of the discrimination power of discrete hidden Markov models. In *Proc. of the 6rd International Conference on Document Analysis and Recognition*, pages 350–354, Seattle, USA, September 10-13, 2001.

[60] F. Grandidier, R. Sabourin, C.Y. Suen, and M. Gilloux. A new strategy for improving feature sets in a discrete HMM-based handwriting recognition system. In *Proc. of the 7th International Workshop on Frontiers in Handwriting Recognition*, pages 113–122, Amsterdam, Netherlands, September 11-13, 2000.

[61] G.H. Granlund. The complexity of vision. *Signal Processing*, 74 :101–126, 1999.

[62] D. Guillevic and C.Y. Suen. HMM-KNN word recognition engine for bank cheque processing. In *Proc. of the 13th International Conference on Pattern Recognition*, pages 1526–1529, Brisbane, Australia, August 16-20 1998.

[63] K. Han and I.K. Sethi. An off-line cursive handwritten word recognition system and its application to legal amount interpretation. *International Journal of Pattern Recognition and Artificial Intelligence*, 11(5) :757–770, 1997.

[64] T. Heskes. Self-organizing maps, vector quantization, and mixture modeling. *IEEE Transactions on Neural Networks*, 12(6) :1299–1305, Nov. 2001.

[65] L. Heutte, T. Paquet, J.V Moreau, Y. Lecourtier, and C. Olivier. A structural/statistical feature based vector for handwritten character recognition. *Pattern Recognition Letter*, 19 :629–641, 1998.

[66] Laurent Heutte. *Reconnaissance de caractères manuscrits : application à la lecture automatique des chèques et des enveloppes postales*. PhD thesis, Université de Rouen, 1994.

[67] M.K. Hu. Visual pattern recognition by moments invariants. *IRE Trans. on Information Theory*, 8 :179–187, 1962.

[68] C.-M. Huang and R.W. Harris. A comparison of several vector quantization codebook generation approaches. *IEEE Transactions on Image Processing*, 2(1) :108–112, Jan. 1993.

[69] X.D. Huang, Y. Ariki, and M.A. Jack. *Hidden Markov Models for Speech Recognition*, volume 7 of *Information Technology Series*. Edinburgh University Press, 1990.

[70] X.D. Huang and M.A. Jack. Semi-continuous hidden Markov models for speech signals. *Computer Speech and Language*, 3 :239–251, 1989.

[71] J.J. Hull. A database for handwritten text recognition research. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 16(5) :550–554, May 1992.

[72] A. Hyvärinen. Survey on independent component analysis. *Neural Computing Survey*, 2 :94–128, 1999. http ://www.cis.hut.fi/aapo/papers/NCS99web/.

[73] S. Impedovo, L Ottaviano, and S. Occhinegro. A new method for automatic reading of typed/handwritten numerals. In *Proc. of the 2nd Workshop on Frontiers in Handwriting Recognition*, pages 427–433, Chateau de Bonas, France, 1991.

[74] A. Jain and D. Zongker. Feature selection : Evaluation, application and small sample performance. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 19(2) :153–158, February 1997.

[75] A.K. Jain, R.P.W. Duin, and J. Mao. Statistical pattern recognition : A review. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 22(1) :4–37, January 2000.

[76] F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, Cambridge Massachusetts, 1998.

[77] G.H. John, R. Kohavi, and K. Pfleger. Irrelevant features and subset selection problem. In *Proc. of the 11th International Conference on Machine Learning*, pages 121–129, 1994.

[78] S.M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Trans. on Acoustics, Speech and Signal Processing*, 35(3) :400–401, March 1987.

[79] A. Khotanzad and Y.H. Hong. Invariant iamge recognition by Zernike moments. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 12(5), May 1990.

[80] G. Kim and V. Govindaraju. Handwritten word recognition for real-time applications. In *Proc. of the 3rd International Conference on Document Analysis and Recognition*, pages 24–27, Montreal, Canada, August 14-16 1995.

[81] G. Kim and V. Govindaraju. Lexicon driven approach to handwritten word recognition for real-time applications. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 19(4) :366–379, 1997.

[82] G. Kim and V. Govindaraju. Handwritten phrase recognition as applied to street name images. *Pattern Recognition*, 31(1) :41–51, 1998.

[83] F. Kimura and M. Sridhar. Handwritten numerical recognition based on multiple algorithms. *Pattern Recognition*, 24(10) :969–983, 1991.

[84] K. Kira and L.A. Rendell. The feature selection problem : Traditional methods and a new algorithm. In *Proc. of the 10th National Conference on Artificial Intelligence (AAAI)*, pages 129–134, San Jose, USA, July 12-16 1992.

[85] J. Kittler. Feature set search algorithms. In C. H. Chen, editor, *Pattern Recognition and Signal Processing*, pages 41–60, 1978.

[86] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas. On combining classifiers. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 20(3) :226–239, March 1998.

[87] S. Knerr and E. Augustin. A neural network-hidden Markov model hybrid for cursive word recognition. In *Proc. of the 13th International Conference on Pattern Recognition*, pages 1518–1520, Brisbane, Australia, August 16-20 1998.

[88] S. Knerr, O. Baret, D. Price, J.-C. Simon, V. Anissimov, and N. Gorski. The A2iA recognition system for handwritten checks. In *Proc. of the 2nd Workshop on Document Analysis Systems*, pages 431–494, Philadelphia, 1996.

[89] A.L. Knoll. Experiments with characteristic loci for recognition of handprinted characters. *IEEE Trans. on Computers*, 18 :366–372, 1969.

[90] A. L. Koerich, R. Sabourin, and C.Y. Suen. Large vocabulary off-line handwriting recognition : A survey. *Pattern Analysis and Application*, 2002. Accepter pour publication.

[91] Alessandro L. Koerich. *Large Vocabulary Off-Line Handwritten Word Recognition*. PhD thesis, Ecole de technologie supérieure - Université du Québec, Août 2002.

[92] D. Koller and M. Sahami. Toward optimal feature selection. In *Proc. of the 13th International Conference on Machine Learning*, pages 284–292, Bari, Italy, July 1996. Morgan Kaufmann.

[93] I. Kononenko. Estimating attributes : Analysis and extensions of RELIEF. In *Proc. of the European Conference on Machine Learning*, volume 784 of *Lecture Notes in Computer Science*, pages 171–182, Catania, Italy, April 6-8 1994. Springer.

[94] M. Kudo and J. Sklansky. A comparative evaluation of medium- and large-scale feature selectors for pattern classifiers. In *Proc of the 1st International Workshop on Statistical Techniques in Pattern Recognition*, pages 91–96, Prague, Czech Republic, 1997.

[95] M. Kudo and J. Sklansky. Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition*, 33 :25–41, 2000.

[96] A. Kundu. *Handbook of Character Recognition and Document Image Analysis*, chapter Handwritten Word Recognition Using Hidden Markov Model. World Scientific Publishing Company, 1997.

[97] A. Kundu, H. He, and M.Y. Chen. Alternative to variable duration HMM in handwriting recognition. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 20(11) :1275–1281, 1998.

[98] A. Kundu, Y. He, and P. Bahl. Recognition handwritten word : first and second order hidden Markov model based approach. *Pattern Recognition*, 22(3) :283–297, 1989.

[99] J. Laaksonen and E. Oja. Learning subspace classifiers and error-corrective feature extraction. *International Journal of Pattern Recognition and Artificial Intelligence*, 12(4) :423–436, 1998.

[100] Eric Lecolinet. *Segmentation d'images de mots manuscrits : Application à la lecture de chaîne de caractères majuscules alphanumériques et à la lecture de l'écriture manuscrites*. PhD thesis, Université Paris 6, Mars 1990. 283 pages.

[101] Philippe Leray. *Apprentissage et Diagnostic de Systèmes Complexe : Réseaux de Neurones et Réseaux Bayésiens*. PhD thesis, Université Paris 6, 1998.

[102] M. Leroux, E. Lethelier, M. Gilloux, and B. Lemarié. Automatic reading of handwritten amounts on French checks. *International Journal of Pattern Recognition and Artificial Intelligence*, 11(4) :619–638, 1997.

[103] Manuel Leroux. *Reconnaissance de textes manuscrits à vocabulaire limité avec application à la lecture automatique des chèques*. PhD thesis, Université de Rouen, 1991.

[104] Y. Li. Reforming the theory of invariant moments for pattern recognition. *Pattern Recognition*, 25 :723–730, 1992.

[105] Y. Linde, A. Buzo, and R.M. Gray. An algorithm for vector quantizer design. *IEEE Trans. on Communication*, 28(1) :84–95, 1980.

[106] H. Liu and R. Setiono. Feature selection and classification - a probabilistic wrapper approach. In *Proc. of the 9th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, pages 419–424, Fukuoka, Japan, June 4-7 1996.

[107] H. Liu and R. Setiono. A probabilistic approach to feature selection - a filter solution. In *Proc. of the 13th International Conference on Machine Learning*, pages 319–327, Bari, Italy, July 1996. Morgan Kaufmann.

[108] H. Liu and R. Setiono. Scalable feature selection for large sized databases. In *Proc. of the 4th World Congress on Expert Systems*, Mexico City, Mexico, March 1998.

[109] S. Loncaric. A survey of shape analysis techniques. *Pattern Recognition*, 31(8) :983–1001, 1998.

[110] S. Madhvanath and V. Govindaraju. Perceptual features for off-line handwritten word recognition : A framework for heuristic prediction and matching. In *Proc. of the IAPR Workshop Matching and Representation, Syntactic and Statistical Pattern Recognition*, pages 524–531, Sydney, Australia, August 1998.

[111] S. Madhvanath and V. Govindaraju. Local references lines for handwritten phrase recognition. *Pattern Recognition*, 32 :2021–2028, 1999.

[112] S.A. Mahmoud. Arabic character recognition using Fourier descriptors and character contour encoding. *Pattern Recognition*, 27(6) :815–824, 1994.

[113] Y. Mallet, D. Coomans, J. Kautsky, and O. De Vel. Classification using adaptive wavelets for feature extraction. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 19(10) :1058–1066, 1997.

[114] D. Mangalagiu and M. Weinfeld. Optimisation de l'algorithme des k plus proches voisins dans un système coopératif de reconnaissance de l'écriture. In *Proc. of the 1er Colloque International Francophone sur l'Ecrit et le Document*, pages 151–160, Québec, Canada, 11-13 mai 1998.

[115] J. McNames. A fast nearest-neighbor algorithm based on a principal axis search tree. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 23(9) :964–976, Sep. 2001.

[116] M. Mitchell, J.H. Holland, and S. Forrest. When will a genetic algorithm outperform hill climbing. In *Advances in Neural Information Processing Systems*, volume 6, pages 51–58. Morgan Kaufmann Publishers, Inc., 1994.

[117] M. Modrzejewski. Feature selection using rough sets theory. In *Proc. of the European Conference on Machine Learning*, volume 667 of *Lecture Notes in Computer Science*, pages 213–226, Vienna, Austria,, April 5-7, 1993. Springer-Verlag.

[118] M. Mohamed and P. Gader. Handwritten word recognition using segmentation-free hidden Markov modeling and segmentation-based dynamic programming techniques. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 18(5) :548–554, 1996.

[119] J. Mohorko, P. Planinsic, and C. Zarko. Fast algorithm for pyramid vector quantization. *IEEE Signal Processing Letters*, 8(4) :102–105, 2001.

[120] A.N. Mucciardi and E.E. Gose. A comparison of seven techniques for choosing subsets of pattern recognition properties. *IEEE Trans. on Computers*, 20(9) :1023–1031, September 1971.

[121] P.M. Narendra and K. Fukunaga. A branch and bound algorithm for feature subsets selection. *IEEE Trans. on Computers*, 26(9) :917–922, September 1977.

[122] I.S. Oh and C.Y. Suen. Distance features for neural network-based recognition of handwritten characters. *International Journal on Document Analysis and Recognition (IJDAR)*, 1(2) :73–88, 1998.

[123] L.S. Oliveira, R. Sabourin, F. Bortolozzi, and C.Y. Suen. Automatic reognition of handwritten numerical strings : A recognition and verification strategy. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 24(11) :1438–1454, November 2002.

[124] A. Orlitsky. Scalar versus vector quantization : worst case analysis. *IEEE Transactions on Information Theory*, 48(6) :1393–1409, June 2002.

[125] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. on Systems, Man, and Cybernetics*, SMC-9(1) :62–66, Jan. 1979.

[126] T. Paquet and Y. Lecourtier. Automatic reading of the literal amount of bank checks. *Machine Vision and Applications*, 6 :151–162, 1993.

[127] T. Paquet and Y. Lecourtier. Recognition of handwritten sentences using a restricted lexicon. *Pattern Recognition*, 26(3) :391–407, 1993.

[128] H.-S. Park and S.-W. Lee. Off-line recognition of large-set handwritten characters with multiple hidden Markov models. *Pattern Recognition*, 29(2) :231–244, 1996.

[129] R. Plamondon and S.N. Srihari. On-line and off-line handwriting recognition : A comprehensive survey. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 22(1) :63–84, January 2000.

[130] A.B. Poritz. Hidden Markov models : A guided tour. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7–13, New York, USA, April 1988.

[131] Groupe La Poste. Les échanges - L'avenir - La Poste, Rapport Annuel 2001. http ://www.laposte.fr/decouvre/ra2001.pdf.

[132] P. Pudil, J. Novovičová, and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Letter*, 15 :1119–1125, November 1994.

[133] W.F. Punch, E.D. Punch, M. Pei, L. Chia-Shun, P. Hovland, and R. Enbody. Further research on feature selection and classification using genetic algorithms. In *Proc of the 5th International Conference on Genetic Algorithms*, pages 557–564, Urbana-Champaign, USA, July 1993.

[134] J.R. Quinlan. *C4.5 : Programs for Machine Learning*. Morgan Kaufmann, San Mateo, California, USA, 1993.

[135] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2) :257–285, 1989.

[136] M.L. Raymer, W.F. Punch, E.D. Goodman, L.A.Huhn, and A.K. Jain. Dimensionality reduction using genetic algorithms. *IEEE Trans. on Evolutionary Computation*, 4(2) :164–171, July 2000.

[137] K.M. Sayre. Machine recognition of handwritten words : A project report. *Pattern Recognition*, 5(3) :213–228, 1973.

[138] C. Scagliola. Search algorithms for the recognition of cursive phrases without word segmentation. In *Proc. of the 6th International Workshop on Frontiers in Handwriting Recognition*, pages 123–132, Taejon, Korea, 1998.

[139] L. Schomaker and E. Segers. Finding features used in the human reading of cursive handwriting. *International Journal on Document Analysis and Recognition*, 2 :13–18, 1999.

[140] A.W. Senior and A.J. Robinson. An off-line cursive handwriting recognition system. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 20(3) :309–321, 1998.

[141] C.E. Shannon. A mathematical theory of communication (part 1). *The Bell System Technical Journal*, 27 :379–423, July 1948.

[142] C.E. Shannon. A mathematical theory of communication (part 2). *The Bell System Technical Journal*, 27 :623–656, October 1948.

[143] D. Shen and H.H.S. Ip. Discriminative wavelet shape descriptors for recognition of 2-D patterns. *Pattern Recognition*, 32 :151–165, 1999.

[144] J.-C. Simon. Off-line cursive word recognition. *Proceedings of the IEEE*, 80(7) :1150–1161, July 1992.

[145] D.B. Skalak. Prototype and feature selection by sampling and random mutation hill climbing algorithms. In *Proc. of the 11th International Conference on Machine Learning*, pages 293–301, New Brunswick, USA, 1994. Morgan Kaufmann.

[146] T. Starner, J. Makhoul, R. Schwartz, and G. Chou. On-line cursive handwriting recognition using speech recognition methods. In *Proc. of the International Conference on Acoustics, Speech and Signal Processing*, volume 5, pages 125–128, 1994.

[147] T. Steinherz, E. Rivlin, and N. Intrator. Offline cursive script word recognition - a survey. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(2/3) :90–110, 1999.

[148] C.Y. Suen, J. Guo, and Z.C. Li. Analysis and recognition of alphanumeric handprints by parts. *IEEE Trans. on Systems, Man, and Cybernetics*, 24(4) :614–631, April 1994.

[149] C.C. Tappert, C.Y. Suen, and T. Wakara. The state of the art in on-line handwriting recognition. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 12(8) :787–808, 1990.

[150] Ø.D. Trier, A.K. Jain, and T. Taxt. Feature extraction methods for character recognition - a survey. *Pattern Recognition*, 29(4) :641–662, 1996.

[151] H. Vafaie and I. Imam. Feature selection methods : Genetic algorithms vs. greedy-like search. In *Proc. of the 9th International Conference on Fuzzy and Intelligent Control Systems*, Louisville, USA, 1994.

[152] B. Verma, P. Gader, and W. Chen. Fusion of multiple handwritten word recognition techniques. *Pattern Recognition Letter*, 22 :991–998, 2001.

[153] A. Vinciarelli. A survey on off-line cursive word recognition. *Pattern Recognition*, 35(7) :1433–1446, July 2002.

[154] Å. Wallin and O. Kübler. Complete sets of Zernike moment invariants and the role of the pseudoinvariants. *IEEE Trans. on Pattern Analysis and Machine Recognition*, 17(11), November 1995.

[155] J. Wood. Invariant pattern recognition : A review. *Pattern Recognition*, 29(1) :1–17, 1996.

[156] L. Xu, A. Krzyżak, and C.Y. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. on Systems, Man and Cybernetics*, 22(3) :418–435, May/June 1992.