Laboratoire d'Informatique,
de Traitement de l'Information et des Systèmes
Université de Rouen
UFR de sciences et techniques

Thèse en vue de l'obtention du diplôme de
Docteur de l'Université de Rouen

Discipline: informatique

Extraction de séquences numériques dans des documents manuscrits quelconques

Clément Chatelain

Présentée le 5 décembre 2006 devant le jury composé de :

Laurent Heutte	Université de Rouen, Encadrant
Guy Lorette	Université de Rennes, Président du jury
Jean-Marc Ogier	Université de la Rochelle, Rapporteur
Thierry Paquet	Université de Rouen, Encadrant
Franck Signorile	EMC-Captiva, Invité
Christian Viard-Gaudin	Université de Nantes, Rapporteur

Résumé

Dans le cadre du traitement automatique de courriers entrants, nous présentons dans cette thèse l'étude, la conception et la mise en œuvre d'un système d'extraction de champs numériques dans des documents manuscrits quelconques. En effet, si la reconnaissance d'entités manuscrites isolées peut être considérée comme un problème en partie résolu, l'extraction d'information dans des images de documents aussi complexes et peu contraints que les courriers manuscrits libres reste à ce jour un réel défi. Ce problème nécessite aussi bien la mise en œuvre de méthodes classiques de reconnaissance d'entités manuscrites que de méthodes issues du domaine de l'extraction d'information dans des documents électroniques. Notre contribution repose sur le développement de deux stratégies différentes : la première réalise l'extraction des champs numériques en se basant sur les techniques classiques de reconnaissance de l'écriture, alors que la seconde, plus proche des méthodes utilisées pour l'extraction d'information, réalise indépendamment la localisation et la reconnaissance des champs. Les résultats obtenus sur une base réelle de courriers manuscrits montrent que les choix plus originaux de la seconde approche se révèlent également plus pertinents. Il en résulte un système complet, générique et industrialisable répondant à l'une des perspectives émergentes dans le domaine de la lecture automatique de documents manuscrits : l'extraction d'informations complexes dans des images de documents quelconques.

Abstract

Within the framework of the automatic treatment of incoming mail documents, we present in this thesis the study, the conception and the development of a numerical field extraction system in handwritten documents. Indeed, the recognition of isolated handwritten entities can be considered as a partially resolved problem, but the extraction of information in images of complex and structure-free documents is still a real challenge. This problem requires the implementation of both handwriting recognition and information extraction methods. Our contribution consists in the development of two different strategies : the first one is based on the extension of classical handwriting recognition methods, while the second is inspired from methods used within the domain of information extraction in electronic documents. The results obtained on a real handwritten mail database show that the original choices of the second approach are more relevant. Finally, a complete, generic and industrialisable system is produced. Hence, this answering one of the emergent perspectives

in the field of the automatic reading of handwritten documents : the extraction of complex information in images of some documents.

Table des matières

Introduction générale

Avec l'apparition de l'écriture, l'homme s'est pourvu d'une alternative à la parole lui permettant d'externaliser et de structurer sa pensée. L'écriture manuscrite demeure aujourd'hui l'un des moyens de communication les plus simples et les plus expressifs, permettant d'exprimer l'identité et la culture d'un individu. L'écriture manuscrite a ainsi su s'imposer comme un fondement de nombreuses civilisations. Aujourd'hui, malgré l'avènement des nouvelles technologies, elle reste un moyen de communication incontournable.

Par ailleurs, chaque individu émet et reçoit une quantité d'information toujours croissante, face à laquelle notre société doit mettre en œuvre un traitement de masse. De plus, les contraintes économiques imposent un traitement rapide de cette information. Dans ce contexte, il est possible de s'interroger sur la place de l'écriture manuscrite, puisque l'interprétation de celle-ci requiert une intervention humaine lente et coûteuse. Dès lors, l'idée d'une automatisation s'est naturellement imposée. Si l'imprimerie puis l'informatique ont permis d'automatiser le processus d'écriture, celle-ci n'a pas pour autant disparue de notre société. Certains travaux plus récents ont donc cherché à adapter les machines afin d'automatiser la lecture des documents écrits. Cette adaptation est toutefois délicate, à cause de la difficulté pour une machine de prendre en compte la richesse et la variabilité de l'écriture humaine.

On a ainsi vu apparaître dans les années 50 les premiers systèmes de reconnaissance de caractères imprimés, puis de caractères manuscrits, avec des performances modestes. Ces travaux ont progressivement évolué vers des systèmes de lecture de plus en plus fiables d'entités manuscrites de plus en plus complexes telles que les mots cursifs ou les séquences de chiffres. À partir des années 90, ces travaux ont été intégrés dans des systèmes industriels de lecture automatique de documents remportant un franc succès. Il s'agit en particulier des applications phares de lecture automatique de chèques bancaires, d'adresses postales ou de formulaires. Ces applications industrielles sont désormais parfaitement opérationnelles et traitent plusieurs millions de documents par jour.

Hormis ces applications très spécifiques, la reconnaissance de l'écriture manuscrite reste toujours un problème délicat en l'absence de connaissances *a priori* sur les documents traités. Depuis quelques années, un nouveau tournant a été amorcé avec une orientation des recherches vers la lecture automatique de documents aux contenus moins contraints tels que les textes libres. Initialement dénués de motivations applicatives industrielles, ces travaux ont cherché à effectuer une lecture

intégrale de textes. Plus récemment, les besoins industriels se sont précisés, dans l'optique d'effectuer un traitement automatique des masses de courriers manuscrits reçus quotidiennement en très grand nombre par les grandes entreprises ou administrations. Afin de traiter cette masse de documents appelée «courrier entrant», l'idée d'en dégager l'information pertinente plutôt que d'en effectuer une lecture complète a émergé. Il s'agit ainsi d'une problématique d'extraction d'information, visant à résumer un document par un ensemble de champs pertinents tels que l'objet du courrier, le nom de l'expéditeur, la date de l'envoi du courrier, etc.

Le sujet traité dans cette thèse concerne l'extraction de séquences numériques dans des documents manuscrits quelconques, et se situe donc pleinement dans cette nouvelle problématique. Il s'agit d'extraire des séquences numériques qui constituent une information pertinente pour la tâche de traitement automatique du courrier. Les numéros de téléphone, les codes postaux ou les numéros de contrat permettent par exemple d'effectuer un tri du courrier vers le service compétent dans l'entreprise. La problématique se situe donc au croisement de deux domaines de recherches : la *reconnaissance de l'écriture manuscrite* et l'*extraction d'information*. Si ces deux disciplines ont été largement étudiées indépendamment, les travaux concernant l'extraction d'information dans les documents manuscrits sont beaucoup plus rares. Notre première contribution concerne ainsi la réflexion autour des stratégies envisageables pour l'extraction d'informations dans des images de documents. À partir de cette réflexion, nous dégagons deux stratégies pertinentes et opposées, que nous proposons de mettre en œuvre. La conception et la mise place de deux chaînes de traitement issues de ces stratégies constitue notre seconde contribution. Il en résulte des systèmes complets, génériques et industrialisables permettant d'effectuer l'extraction de séquences numériques dans des documents manuscrits faiblement contraints.

Afin de présenter la méthodologie relative à la reconnaissance de l'écriture manuscrite, nous proposons dans le chapitre 1 un panorama des méthodes existantes pour la reconnaissance d'entités manuscrites isolées : caractères, mots et séquences numériques. Nous montrons que ces systèmes atteignent désormais des performances satisfaisantes, notamment grâce aux progrès réalisés ces dernières années dans le domaine de la classification statistique et la modélisation de l'écriture.

Le chapitre 2 est consacré à l'étude des systèmes complets de lecture de documents. Ces systèmes reposent à la fois sur une intégration des méthodes de reconnaissance d'entités manuscrites isolées décrites précédemment, et sur un processus de localisation de l'information. Nous passons en revue les différentes méthodes de localisation des informations dans les documents plus ou moins contraints, et montrons que lorsque des documents trop peu contraints sont traités, la localisation des informations peut être vue comme un réel problème d'extraction d'information. Nous donnons ainsi un aperçu des travaux menés dans le domaine de l'extraction d'informations dans les documents textuels électroniques, et en présentons brièvement les différentes étapes de traitement. Ces systèmes d'extraction d'information n'étant pas directement applicables aux documents manuscrits, nous envisageons dans une dernière partie plusieurs stratégies pour l'extraction de séquences numériques dans

les documents manuscrits. Deux stratégies se dégagent : la première, la plus intuitive, cherche à localiser et à reconnaitre les chiffres dans le document pour ensuite localiser les champs à l'aide des connaissances *a priori* sur les champs numériques recherchés (nombre de chiffres, position des séparateurs). La seconde approche, plus originale, cherche à localiser directement les composantes des champs recherchés en exploitant le plus tôt possible les connaissances *a priori*, sans utiliser la reconnaissance chiffre. Celle-ci est appliquée dans un second temps sur les séquences localisées. Les deux chapitres suivants sont consacrés à la mise en œuvre de ces deux stratégies.

Ainsi, nous présentons dans le chapitre 3 la réalisation d'une première chaîne de traitement complète pour l'extraction des champs numériques dans des documents quelconques, basée sur la stratégie la plus «évidente» évoquée dans le chapitre précédent. La mise en œuvre repose sur une localisation et une reconnaissance des chiffres dans le document afin d'extraire les champs recherchés. Les résultats montrent que le système permet d'obtenir des performances satisfaisantes en rappel, même si la précision du système est relativement faible.

Dans le chapitre 4, une seconde chaîne de traitement plus originale est présentée, inspirée des méthodes d'extraction d'information dans les documents électroniques. En exploitant uniquement les connaissances *a priori* relatives aux champs recherchés, la localisation des champs est effectuée sans procéder à la reconnaissance des entités. Les hypothèses de localisation sont ensuite soumises à un module de reconnaissance de champs numériques spécifique chargé de déterminer la valeur numérique des champs. Afin de fiabiliser les résultats, nous proposons une étape de vérification des hypothèses de champs fondée sur l'analyse des résultats de la reconnaisance pour accepter ou rejeter les hypothèses de localisation/reconnaissance de champs. Les résultats montrent que cette seconde méthode semble être le meilleur moyen d'aborder le problème puisque ses performances dépassent celles de la première.

Afin d'améliorer les performances des deux systèmes, nous revenons dans le chapitre 5 sur l'un des points clefs commun aux deux stratégies qui concerne le rejet des composantes non numériques. Après avoir décrit les différents types de rejets et passé en revue les techniques de la littérature permettant de les prendre en compte, nous proposons une approche séquentielle en deux étapes pour le rejet des formes non numériques. La première étape filtre les rejets dits «évidents», alors que la seconde traite les formes plus ambigües. Nous nous concentrons plus particulièrement sur le développement de la première étape qui soulève un problème de classification où les coûts de mauvaise classification sont à la fois déséquilibrés et inconnus. Pour résoudre ce problème, nous présentons un algorithme d'apprentissage multi-objectif appliqué à un classifieur SVM. Une comparaison des résultats obtenus avec une autre méthode d'apprentissage de la littérature montre que notre approche est efficace. Nous présentons enfin l'intégration de ce module de filtrage des rejets dans les deux chaînes de traitement et évaluons ses performances.

Chapitre 1

Systèmes de reconnaissance de l'écriture manuscrite

1.1 Introduction

La reconnaissance de l'écriture manuscrite a connu ces dernières années de grands progrès, et les succès des travaux de recherches ont donné lieu à de nombreuses applications industrielles, notamment dans le domaine de la lecture automatique de formulaires [Ramdane 03, Cracknell 98, Milewski 06b], de chèques [Impedovo 97b, Lethelier 96] ou d'adresses postales [Cohen 94, El-Yacoubi 02, Kim 98], ainsi que les applications de reconnaissance de l'écriture dites «en ligne» [Seni 96, Connell 02] à travers les PDA, tablet-PC ou stylo caméra.

Dans ce chapitre, nous nous focalisons sur la reconnaissance d'entités manusrites dèjà localisées. Nous abordons donc le problème de la reconnaissance de caractères isolés, de mots, de chiffres liés ou de séquences numériques en considérant qu'ils ont été localisés au préalable. Pourtant, la localisation ne devrait pas être dissociée de la reconnaissance puisque, d'après le paradoxe énoncé par Sayre «pour reconnaître une entité, il faut savoir la localiser, mais pour la localiser, il faut tout d'abord la reconnaître» [Sayre 73]. Il n'empèche que dans la littérature, la grande majorité des travaux en reconnaissance de l'écriture concerne la seule reconnaissance des entités, une fois les entités segmentées. Cette tendance générale s'explique certainement par le fait que jusqu'à présent, la plupart des recherches ont eu lieu dans les cadres applicatifs du traitement automatique des chèques et du courrier postal, pour lesquels un certain nombre de connaissances *a priori* sur les documents facilitent grandement la localisation des entités. Nous proposons donc dans ce chapitre un état de l'art des méthodes de reconnaissance d'entités déjà localisées, ce qui pose les briques élémentaires nécessaires à la compréhension du chapitre suivant consacré au problème plus général de localisation/reconnaissance d'entités manuscrites dans les documents.

D'une manière générale, la complexité de la reconnaissance d'information manuscrite dépend de plusieurs critères [Crettez 98, Lorette 92, Koerich 03b] :

- Le conditionnement de l'information : l'écriture reconnue peut être plus ou moins conditionnée par la présence de précasé (cas des formulaires, code postal d'une adresse), des cadres (montants de chèques) ou lignes (lignes d'un bloc adresse, montant littéral d'un chèque), ou bien non conditionnée (documents libres).
- Le style d'écriture (voir figure 1.1) : selon [Tappert 84], la difficulté à reconnaître l'écriture augmente avec les 5 styles d'écriture suivants : écriture scripte précasée, écriture scripte avec caractères espacés, écriture scripte libre, écriture cursive, écriture mixte cursive et scripte. Notons également la présence d'alphabets non latins plus délicats à reconnaître car possédant un grand nombre de symboles : caractères chinois [Park 96, Liu 00], Kanji [Omachi 00], arabes [Amin 98, El Hajj 05], japonais [Srihari 97b], Devanâgarî [Keeni 96], coréen [Jung 00], etc.
- Le nombre de scripteurs : la réduction du nombre de scripteurs potentiels permet éventuellement de réduire la variabilité et d'apprendre les différents styles d'écriture [Nosary 02]. La difficulté s'accroit en contexte omni-scripteur en raison des styles d'écriture très différents de chacun (voir figure 1.2).
- La taille du vocabulaire : les systèmes de reconnaissance de textes sont souvent basés sur un lexique qui facilite grandement la lecture [Kimura 94, Kim 97b], surtout si celui ci possède un faible nombre de mots (cas des montants littéraux de chèques qui contiennent une trentaine de mots). La reconnaissance de mots est d'autant plus aisée que le nombre de mots dans le lexique est faible. Notons que dans le cas de la reconnaissance de séquences numériques, la présence d'un lexique est plus rare (cas de la reconnaissance de codes postaux ou de numéros INSEE).

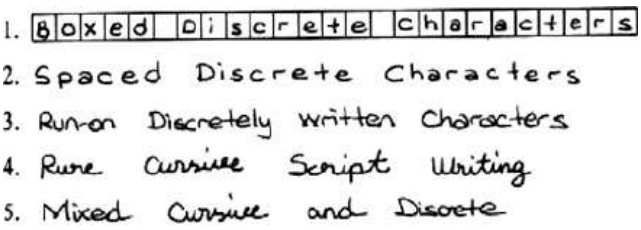


FIG. 1.1 – Classification des 5 styles d'écriture du plus facile (1) au plus difficile (5) à reconnaître selon [Tappert 84].

1.2 Reconnaissance de caractères isolés

Au cœur des systèmes de reconnaissance de l'écriture manuscrite, ce sont les moteurs de reconnaissance de caractères isolés qui ont le plus bénéficié des recherches [Plamondon 00]. Apparus dans les années 50, les premiers moteurs de reconnais-

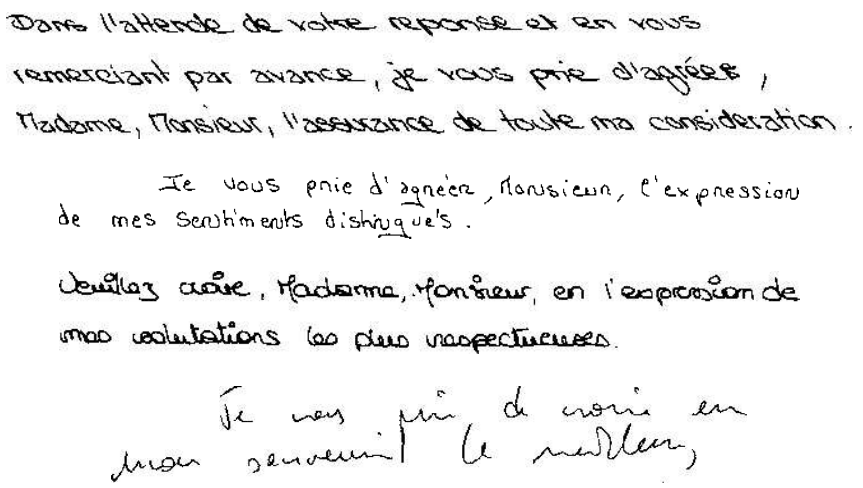


FIG. 1.2 – Différences d'écriture suivant les scripteurs.

sance de caractères dactylographiés étaient basés sur des algorithmes de «template matching» cherchant à faire correspondre la forme inconnue à une bibliothèque de modèles de référence aux patrons (template). Puis on vit apparaître les premiers moteurs de reconnaissance de caractères manuscrits, basés sur l'extraction de vecteurs de caractéristiques de bas niveau sur des images binarisées soumises à des classifieurs statistiques [Arica 01]. Durant cette période, la puissance limitée des machines et la mauvaise qualité des systèmes d'acquisition de données ont toutefois bridé les travaux. À partir des années 80, l'apparition des tablettes graphiques pouvant capturer les coordonnées du mouvement du tracé a permis aux chercheurs de s'intéresser à la reconnaissance de l'éciture en-ligne [Suen 90]. Simultanément, avec l'explosion des technologies de l'information, la puissance des machines a augmenté et les méthodologies developpées auparavant ont pu être mises en œuvre. On a alors pu voir de nombreuses applications utilisant la reconnaissance de caractères manuscrits [Bozinovic 89, Govindan 90]. Depuis les années 90, les progrès faits en traitement d'image, reconnaissance de formes et classification ont amorcé une nouvelle évolution dans les systèmes de reconnaissance d'écriture. Les techniques statistiques modernes telles que les réseaux de neurones, les machines à vecteurs de support ou les modèles de Markov cachés, couplées à une nouvelle augmentation de la puissance des machines ainsi qu'à une amélioration des scanners ou des tablettes graphiques ont permis d'obtenir les premiers résultats satisfaisants pour la reconnaissance de l'écriture. En particulier, on obtient désormais des résultats acceptables pour la reconnaissance de caractères manuscrits isolés [Arica 01] ou pour la reconnaissance de mots en contexte mono-scripteur avec un lexique limité [Plamondon 00, Nosary 02].

Le problème de la reconnaissance de caractères isolés est le suivant : étant donnée une image de caractère isolé (chiffres, lettres minuscules ou majuscule, symbole :

ponctuation, symbole monétaire, etc.), la reconnaissance de caractère vise à lui attribuer sa classe d'appartenance à l'aide d'un algorithme de reconnaissance de formes. La difficulté du problème vient de la variabilité des formes de caractères lorsqu'on se situe en contexte omni-scripteur. On peut constater ces différences sur la figure 1.3 en considérant les dix classes de chiffres : remarquons les variations de taille, de structure, d'inclinaison et de trait au sein d'une même classe.

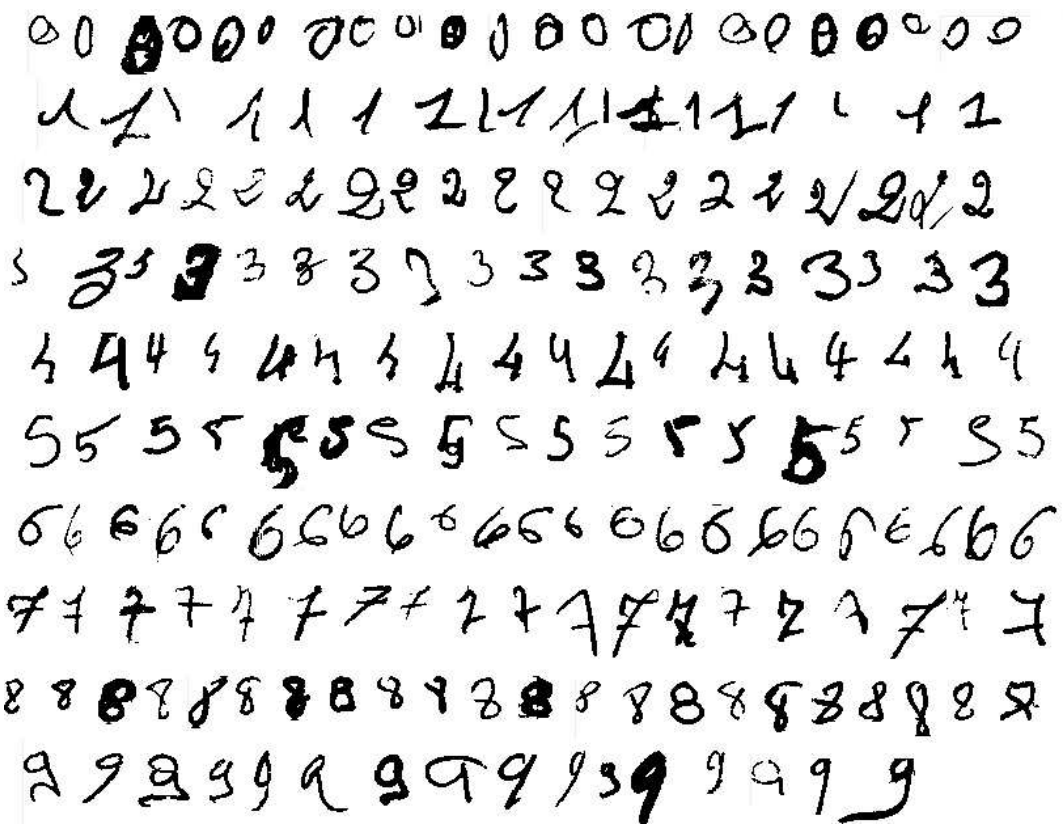


FIG. 1.3 – Variabilité des caractères manuscrits : exemple des 10 classes de chiffre.

L'architecture d'un moteur de reconnaissance de caractères manuscrits isolés est, comme pour tout problème de reconnaissance de formes, composé de trois étapes (hormis l'acquisition des données) : les prétraitements, la représentation des données et la prise de décision [Jain 00] (voir figure 1.4). Les prétraitements visent à transformer l'image en vue de faciliter les traitement ultérieurs : lissage ou redressement des caractères, homogénéisation de l'épaisseur du trait, etc. En reconnaissance de caractères, la représentation des données se traduit par une phase d'extraction de caractéristiques donnant une description synthétique de la forme à reconnaître dans un espace à plusieurs dimensions. La prise de décision s'effectue à l'aide d'un classifieur qui se prononce sur l'appartenance de la forme à une ou plusieurs classes de

caractère. Les moteurs de reconnaissance de caractères manuscrits isolés donnent désormais de bons résultats, grâce aux travaux concernant les extracteurs de caractéristiques [Trier 96] et les classifieurs [Jain 00].

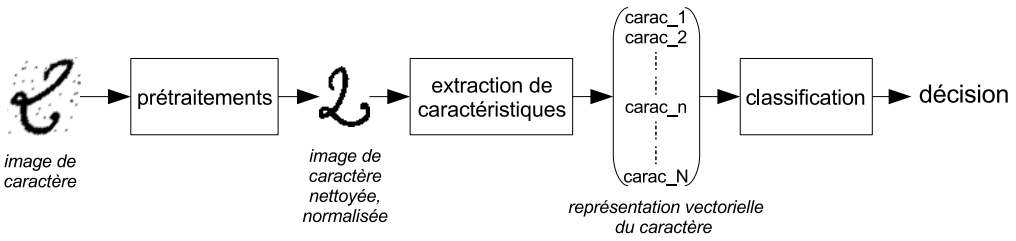


FIG. 1.4 – Chaine de traitement pour la reconnaissance de caractères manuscrits.

La conception d'un moteur de reconnaissance de caractères dépend donc des choix effectués pour les trois étapes : prétraitements, extraction de caractéristiques, classifieur. Ceux-ci doivent être développés simultanément car les choix faits lors d'une étape peuvent influer sur les deux autres. Il est par exemple inutile d'effectuer un redressement de l'image si l'on utilise des caractéristiques insensibles à la rotation. De la même manière, certains classifieurs sont incompatibles avec un vecteur de caractéristiques à grande dimension. D'où la nécessité de prendre en compte au mieux les contraintes applicatives relatives aux temps de traitement et d'apprentissage, aux capacités de rejet éventuelles requises, au nombre d'exemples disponibles pour l'apprentissage, etc.

Nous présentons maintenant les trois étapes d'un moteur de reconnaissance de caractères manuscrits.

1.2.1 Prétraitements

Les quatre étapes de prétraitement classiques sont la binarisation, le débruitage, la normalisation de la taille des caractères et la correction d'inclinaison. Toutes ces étapes ne sont pas systématiquement mises en œuvre.

- Binarisation / seuillage : il s'agit d'assigner aux pixels les types de valeurs adaptées à l'extraction de caractéristiques : noir & blanc ou plus rarement niveaux de gris. L'opération de binarisation vise à séparer l'information manuscrite du fond de l'image à l'aide d'une méthode de seuillage par exemple [Trier 95, Trier 96]. Cette opération permet de réduire la quantité d'information à traiter, tout en conservant le signal à traiter dans sa quasi-intégralité.
- Débruitage : cette étape corrige dans la mesure du possible les imperfections de l'image liées à la capture de l'image, à l'aide d'algorithmes de traitement d'images.
- Normalisation de la taille des caractères : afin de rendre la suite des traitements insensible à la taille des caractères, une étape de normalisation de la taille des caractères est parfois effectuée.

– Correction d'inclinaison : rarement utilisée sur les lettres isolées, une correction d'inclinaison est parfois effectuée sur les chiffres manuscrits. Les méthodes sont basées sur une analyse de l'inclinaison de la forme, et une correction à l'aide d'une transformation par cisaillement [Slavik 01] souvent suivie d'une étape de lissage.

1.2.2 Espace de représentation

Cette étape est une des clefs d'un système efficace de reconnaissance de caractères. En effet, l'utilisation d'un classifieur très performant ne peut compenser une représentation mal adaptée ou peu discriminante. La difficulté de cette étape provient du fait que la qualité d'une représentation ne peut se juger que sur un problème particulier (reconnaissance de chiffres, lettres, symboles), et qu'il n'existe pas de représentation multi-caractères.

On peut définir l'extraction de caractéristiques comme le problème d'*extraction à partir de l'image de l'information la plus pertinente pour un problème de classification donné, c'est à dire celle qui minimise la variabilité intra-classe et qui maximise la variabilité inter-classe*[Devijver 82]. Cette information pertinente prend souvent la forme d'un vecteur de valeurs numériques.

S'il est difficile de sélectionner *a priori* un extracteur de caractéristiques pour un problème donné, on ne peut pas pour autant choisir d'extraire toutes les caractéristiques possibles. Il y a plusieurs raisons à cela : d'une part, l'utilisation d'un grand nombre de caractéristiques avec une méthode de classification statistique implique pour la phase d'apprentissage un nombre d'exemples exponentiel avec la dimension de la représentation. D'autre part, suivant les classifieurs, il est possible que la présence de caractéristiques non discriminantes pour le problème nuise aux performances du classifieur. Enfin soulignons qu'en fonction du classifieur utilisé, l'utilisation d'un grand nombre de caractéristiques peut entraîner des temps de traitement trop importants. Ces observations ont entrainées le développement de méthodes de sélection de caractéristiques [Guyon 03, Oliveira 06] visant à limiter la représentation aux descripteurs les plus pertinents pour le problème.

Concernant la variabilité intra-classe, elle est bien sûr due aux différents styles d'écritures (voir figure 1.3), mais elle est également due aux différentes inclinaisons d'un même caractère. Il est donc possible de réduire une partie de la variabilité intra-classe soit en appliquant des algorithmes de redressement de l'écriture, soit en utilisant des extracteurs de caractéristiques insensibles à l'inclinaison des caractères.

Il existe de nombreux extracteurs de caractéristiques adaptés à la discrimination des caractères manuscrits [Trier 96]. Nous présentons les familles de caractéristiques les plus utilisées en reconnaissance de caractères manuscrits.

– Les caractéristiques les plus simples sont les valeurs même des pixels. L'avantage d'utiliser les valeurs des pixels comme caractéristiques est de ne nécessiter aucun traitement, mis à part une étape de normalisation des caractères. Ces caractéristiques, utilisées pour les méthodes d'appariement de forme («template matching»), sont également employées par les classifieurs de type réseaux de

neurones à convolution [LeCun 98, Poisson 05, Bengio 95] (voir section 1.5.2). Dérivées des pixels, les densités de pixel de l'image sont également utilisées. Aussi appelée «zoning», cette technique consiste à découper l'image selon une grille $n * m$ et à calculer la densité de pixels dans chaque case de la grille [Favata 96, Kim 04].

– Une autre famille de caractéristiques concerne les histogrammes des projections de l'image de caractère. Les histogrammes sont obtenus par projections horizontale et verticale des pixels noirs de l'image (voir figure 1.5). Les caractéristiques utilisées peuvent être directement les valeurs des histogrammes éventuellement normalisés, ou bien extraites de ces histogrammes en cherchant par exemple à détecter les pics [Heutte 94].

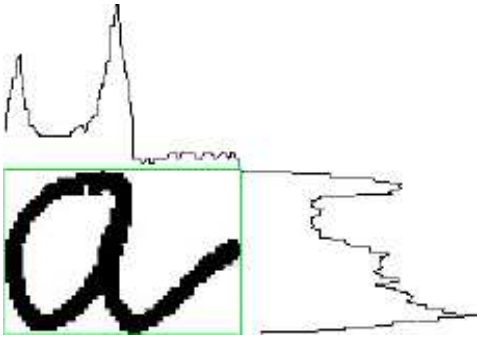


FIG. 1.5 – Histogrammes des projections horizontales et verticales (image provenant de [Koerich 03a]).

– Les quatre profils (haut, bas, droite, gauche) [Shridhar 84] sont obtenus par l'intermédiaire de sondes appliquées sur le caractère. Pour le profil gauche d'un caractère, on lance des sondes depuis le bord gauche de l'image qui s'arrêtent lorsqu'elles rencontrent le premier pixel noir. Les abscisses des sondes constituent le profil gauche du caractère (voir figure 1.6).

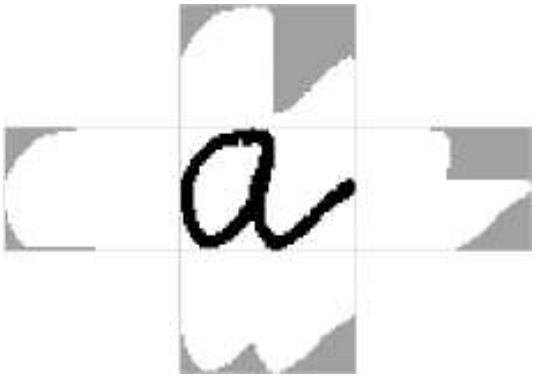


FIG. 1.6 – Les 4 profils d'un caractère (image provenant de [Koerich 03a]).

- Les moments invariants sont des caractéristiques intéressantes car elles sont invariantes en translation, taille et rotation. Ce sont des mesures statistiques de la distribution des pixels autour du centre de gravité du caractère. Citons par exemple les moments invariants de Hu [Hu 62].
- Les caractéristiques extraites des contours sont également très utilisées en reconnaissance de caractères manuscrits [Pal 01, Kimura 94, Taxt 90]. Pour une image de caractère binarisée, les contours contiennent toute l'information de l'imagette, il semble donc naturel d'en extraire des caractéristiques. Les contours sont définis comme l'ensemble des pixels du caractère ayant au moins un pixel en commun avec le fond (en 4 ou 8 connexité). À partir de ce contour, plusieurs caractéristiques peuvent être extraites, telles que les caractéristiques du «chaincode» [Kimura 94]. Après avoir effectué un pavage de l'imagette, l'histogramme des directions de Freeman des pixels est extrait dans chaque zone de l'image. Les histogrammes constituent les caractéristiques du vecteur (voir figure 1.7).

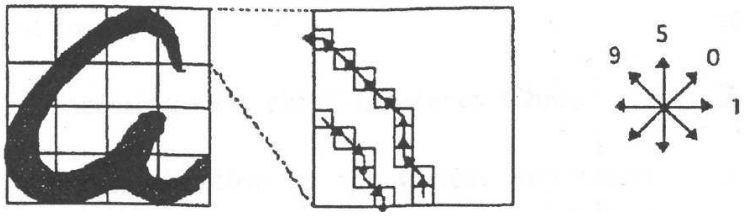


FIG. 1.7 – Image pavée, extraction du coutour et histogramme des directions du contour en 8-connexité sur un des pavés [Kimura 94].

  L'approximation du contour par des fonctions paramètriques telles que les descripteurs de Fourier permet également de générer des caractéristiques [Sekita 88, Taxt 90].
- Nécessitant une étape de squelettisation, les caractéristiques structurelles permettent une description alternative de la forme [CoxIII 82, Heutte 98]. Le nombre et la position des occlusions, des fins de traits, des double et triple jonctions, des extrema, d'intersection avec des sondes horizontales ou verticales constituent autant de caractéristiques pertinentes pour la discrimination des caractères manuscrits. En approximant le squelette d'une forme par des segments de droites et des points de jonction, on peut également extraire des arcs de concavité du caractère. On retrouve aussi les caractéristiques issues des descripteurs de Fourier utilisés sur le squelette.

Une combinaison de différentes familles de caractérisiques est souvent mise en œuvre afin d'obtenir plusieurs représentations d'un même forme et d'améliorer la discrimination [Xue 06, Heutte 98, Foggia 99]. Lorsque le nombre de caractéristiques devient trop élevé, des méthodes de sélection de caractéristiques peuvent être mises en œuvre [Guyon 03, Oliveira 06].

Sélection de caractéristiques

Le but de la sélection de caractéristiques est d'éliminer les caractéristiques non discriminantes ou redondantes. La réduction du nombre de caractéristiques a de nombreux avantages [Guyon 03] : elle permet d'améliorer la visualisation et la compréhension des données, de réduire les temps d'apprentissage et de classification des systèmes, d'améliorer les performances en classification, et permet de réduire la taille des bases d'apprentissage.

Selon [Guyon 03], il existe trois catégories de méthodes pour la sélection de caractéristiques. Les «wrappers» [Kohavi 97, Oliveira 02a] sont les méthodes les plus simples. Elles utilisent le processus de classification comme une boîte noire pour évaluer le pouvoir discriminant des caractéristiques. Leur inconvénient est d'être assez lourdes à mettre en œuvre puisqu'elles nécessitent l'évaluation de toutes les combinaisons possibles de caractéristiques. Les approches «filtres» sélectionnent les caractéristiques indépendemment du comportement du classifieur [Oh 99, Koller 96]. Elles sont beaucoup plus légères à mettre en place et sont génériques car non dépendantes d'un classifieur donné. Enfin les «méthodes embarquées» sélectionnent les caractéristiques pendant le processus d'apprentissage du classifieur [Breiman 84]. Si cette dernière classe de méthodes semble actuellement la plus efficace, elle est peu générique puisqu'elle est liée à un classifieur donné et nécessite le développement d'un algorithme d'apprentissage spécifique.

1.2.3 Classifieurs

Le rôle du classifieur est de se prononcer sur l'appartenance d'une forme à chacune des classes de caractère à partir du vecteur de caractéristiques. Il existe de nombreux classifieurs possédant des caractéristiques de performances et de vitesse différentes [Jain 00, Liu 02b, Liu 02a, Bottou 94]. Selon [Jain 00], il existe quatre grandes familles de classifieurs : le *pattern matching* (ou «appariement de formes» par une mesure de distance ou de corrélation), l'appariement structurel ou syntaxique, la classification statistique, et les réseaux de neurones[1].

Les approches par appariement de formes visent à comparer une forme à des représentants de chaque classe via une mesure de similarité. Elles sont peu adaptées à la reconnaissance de l'écrit car la très forte variabilité des caractères manuscrits implique un nombre très important de représentants pour chaque classe.

Les approches par appariement structurel ou syntaxique reposent sur une représentation hiérarchique des formes. Chaque forme est vue comme un ensemble de sous-formes qu'on appelle «patterns», elles mêmes composées de patterns plus petites. Les plus petites patterns sont des caractéristiques, par exemple une occlusion ou un trait pour les caractères manuscrits. Classiquement, on peut comparer la structure des formes et la syntaxe d'un langage [Jain 00] : les formes sont vues commes des phrases, les caractéristiques sont les lettres de l'alphabet, et les phrases sont obtenues par une grammaire. La grammaire de chaque classe

[1] ces derniers sont parfois classés comme des classifieurs statistiques

doit être inférée à partir des exemples disponibles dans la base d'apprentissage. En travaillant avec la structure des formes, on fait une certaine abstraction de la variabilité de l'écriture, ce qui a motivé de nombreux travaux dans ce domaine [Hu 98, Amin 97, Sadykhov 02, Tsang 98, Verschueren 84]. Bien que séduisantes, ces approches sont toutefois très sensibles aux problèmes de segmentation qui modifient la structure des caractères, ainsi qu'au bruit.

Les classifieurs les plus utilisées en reconnaissance de caractères manuscrits sont incontestablement la classification statistique et les réseaux de neurones. Nous donnons une présentation synthétique de ces deux approches.

1.2.3.1 Approches statistiques

Dans les approches statistiques, chaque forme est vue comme un point dans un espace à n dimensions, n étant le nombre de caractéristiques. Chaque forme x appartenant à la classe u_i est vue comme une observation générée aléatoirement par la distribution de probabilité de la classe u_i : $p(x/u_i)$. La phase d'apprentissage supervisé consiste à déterminer les règles de décision à partir des exemples de la base d'apprentissage. Pour un ensemble d'apprentissage donné, on peut constuire les frontières de décision de deux manières différentes. La première solution consiste à générer les frontières implicitement à partir des distributions de probabilité de chaque classe (approches modélisantes : fenètres de Parzen, mixture de gaussiennes, K plus proches voisins). Le deuxième type d'approche consiste à estimer explicitement les frontières de décision entre les classes (approches discriminantes).

Les approches modélisantes construisent les frontières de décision à partir des distributions de probabilités de chaque classe : $p(x/u_i)$. Lorsque ces densités de probabilités sont connues, on peut obtenir directement les probabilités *a posteriori* d'appartenance de la forme à chaque classe en appliquant la règle de Bayes :

$$p(u_i/x) = \frac{p(x/u_i).p(u_i)}{p(x)}$$

La décision se fait alors en choisissant pour x la classe qui minimise le risque conditionnel $R(u_i/x)$:

$$R(u_i/x) = \sum_{j=1}^{c} L(u_i, u_j).p(u_j/x)$$

où $L(u_i, u_j)$ désigne le coût de mauvaise classification, c'est-à-dire le coût engendré par la décision u_i à la place de la vraie classe u_j.

Cependant les densités de probabilités ne sont généralement pas connues, et elles doivent être estimées à partir de l'ensemble d'apprentissage. Dans ce cas, les densités de probabilité estimées peuvent être paramètriques ou non paramètriques.

Dans les méthodes paramètriques, on considère que la forme des distributions de probabilité est connue, des gaussiennes dans le cas classique. Lors de l'apprentissage, on cherche donc à estimer les paramètres inconnus des gaussiennes pour

chaque classe : moyennes et variances, ou éventuellement matrices de covariances pour chaque classe. Une fois ces paramètres estimés, la décision se fait naturellement par la règle de Bayes. L'inconvénient de ce type d'approche est qu'il introduit un grand nombre de paramètres pour avoir des distributions de probabilités précises, surtout en grande dimension. En particulier, l'estimation des matrices de covariances pour chaque classe demande un nombre d'exemple dans la base d'apprentissage très important, ce qui la rend peu adaptée aux problèmes à grande dimension.

Les méthodes non paramètriques sont mises en œuvre dans le cas où l'on ne dispose pas de connaissances *a priori* sur la distribution de probabilité des classes. Les deux approches non paramètriques les plus connues sont le k plus proche voisin (KPPV) et le classifieur de Parzen. La règle de décision du KPPV est une approche géométrique, alors que le classifieur de Parzen remplace les densités de probabilité par leurs estimées selon la méthode des fenêtres de Parzen. Ces deux approches nécessitent le calcul d'une distance du point à classer à tous les exemples de la base d'apprentissage et sont donc relativement lentes. On peut cependant réduire le nombre d'éléments dans la base d'apprentissage [Fukunaga 89a, Fukunaga 89b].

Les approches discriminantes visent à construire directement des frontières de décision par la minimisation d'un critère d'erreur entre les sorties réelles et escomptées du classifieur. Le critère d'erreur choisi est souvent le taux de classification ou l'erreur quadratique. Il existe plusieurs classifieurs discriminants tels que le classifieur linéaire discriminant de Fisher, le perceptron monocouche, les arbres de décision CART (Classification And Regression Trees) [Breiman 84] et C4.5 [Quinlan 93] ou, plus récemment, les machines à vecteurs de support [Vapnik 95].

Les machines à vecteurs de support ont connu de nombreuses applications en reconnaissance de caractères ces dernières années [Bellili 03, Ayat 02, Oliveira 04, Zhao 00, Cortes 95, Burges 97]. Considérées comme les classifieurs possédant les meilleures capacités de généralisation, elles méritent toute notre attention. Nous en décrivons maintenant le fonctionnement.

1.2.3.2 Les machines à vecteurs de support

Les machines à vecteurs de support (SVM) sont des classifieurs à deux classes introduits par Vapnik [Vapnik 95] et possédant une grande capacité de généralisation. Le principe de l'optimisation des SVM est de maximiser la marge entre les classes, c'est-à-dire l'espace sans exemple autour de la frontière de décision. Pour cela, l'algorithme d'apprentissage sélectionne judicieusement un certain nombre de «vecteurs de support» parmi les exemples de la base d'apprentissage, qui définissent la frontière de décision optimale.

Dans le cas d'un problème de classification à deux classes linéairement séparables, il existe une infinité d'hyperplans capables de séparer parfaitement les deux classes. Pour toutes les formes x_i de classe u_i de la base d'apprentissage, on a :

$$\begin{cases} w^t x_i + w_0 > 0 \text{ si } u_i = 1 \\ w^t x_i + w_0 < 0 \text{ si } u_i = -1 \end{cases}$$

Le principe des SVM est donc de choisir l'hyperplan (w, w_0) qui va maximiser la marge, c'est à dire fournir la plus grande distance possible entre la frontière de décision et les plus proches exemples (voir figure 1.8).

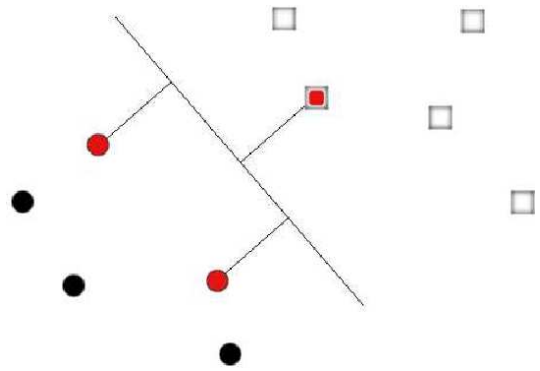


FIG. 1.8 – Hyperplan avec 3 vecteurs de support (en rouge). La position de la frontière maximise la distance entre ces points et leur projeté sur l'hyperplan.

Pour un hyperplan d'équation $h(x) = w^t x + w_0$, la distance d'un point x à l'hyperplan est $h(x)/\|w\|$. La plus gande marge possible entre l'hyperplan et les vecteurs de support est donc obtenue par minimisation de $\frac{1}{2}\|w\|^2$, sous les contraintes d'un bon classement des points de la base d'apprentissage : $u_i(w^t x_i + w_0) > 1$ pour tout $i \in 1, \ldots, m$.

D'après la théorie de l'optimisation, et comme l'objectif (minimiser $\frac{1}{2}\|w\|^2$) et les contraintes $(u_i(w^t x_i + w_0) > 1)$ sont strictement convexes, ce problème peut se poser sous la forme d'un Lagrangien :

$$L(w, w_0, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i \left(u_i.(x_i.w + w_0) - 1\right)$$

dont il faut annuler les dérivées partielles par rapport à w et w_0, les α_i étant les multiplicateurs de Lagrange. Dans cette expression, appelée «expression duale», on constate que les contraintes de bon classement sont présentes sous la forme de pénalités sur le critère. Le théorème de Kuhn-Tucker prouve que ce problème de minimisation sous contraintes est équivalent aux solutions des équations annulant les dérivées du Lagrangien par rapport aux variables w , w_0 , α. L'annulation de ces dérivées partielles donne les α :

$$\begin{cases} Max_\alpha \left\{ \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j u_i u_j (x_i.x_j) \right\} \\ \sum_{i=1}^{m} \alpha_i u_i = 0 \end{cases}$$

ainsi que w_0 :

$$w_0 = \sum_{i=1}^{m} \alpha_i u_i x_i$$

La résolution de ces équations est un problème de programmation quadratique convexe, qui dans la pratique peut être traitée par une des implémentations disponibles [Joachims 99, Platt 99a, Chang 01, Collobert 02].

Les multiplicateurs de Lagrange α_i vérifiant l'annulation des dérivées partielles correspondent aux vecteurs de support. Tous les autres points ont un α_i nul. Finalement, l'équation de l'hyperplan séparateur est :

$$h(x) = (w^* x) + w_0^* = \sum_{i=1}^{m} \alpha_i^* u_i.(x.x_i) + w_0^*$$

où les α_i^* sont les α_i non nuls et w_0^* est trouvé en placant les coordonnées d'un vecteurs de support x_i de classe u_i dans $w^t x_i + w_0 > 0$ si $u_i = 1$ ou dans $w^t x_i + w_0 < 0$ si $u_i = -1$.

Remarquons que la complexité de la décision dépend du nombre de vecteurs de support. On aura donc une décision d'autant plus rapide que le nombre de vecteurs de support conservés à l'issue de l'apprentisage est faible.

Dans les problèmes réels, il est toutefois rare que les classes soient linéairement séparables. Dans le cas contraire, la contrainte de «bon classement» définie initialement par :

$$u_i(w^t x_i + w_0) > 1$$

doit être relachée par l'intermédiaire d'un paramètre ξ_i pour devenir :

$$u_i(w^t x_i + w_0) > 1 - \xi_i$$

Dans ce cas, la plus gande marge possible entre l'hyperplan et les vecteurs de support, initialement obtenue par minimisation de $\frac{1}{2}\|w\|^2$, doit désormais être obtenue par la minimisation de :

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} \xi_i$$

où C désigne un paramètre strictement positif à déterminer.

Nous venons de présenter une méthode de séparation linéaire, donc assez limitée. L'introduction des fonctions noyau va permettre de s'affranchir de cette limitation. On montre dans [Cornuéjols 02] que dans l'équation de l'hyperplan séparateur :

$$h(x) = (w^* x) + w_0^* = \sum_{i=1}^{m} \alpha_i^* u_i.(x.x_i) + w_0^*$$

le produit scalaire $(x.x_i)$ peut être remplacé par n'importe quelle fonction noyau $K(x, x_i)$ réalisant un produit scalaire. L'équation de l'hyperplan devient :

$$h(x) = (w^* x) + w_0^* = \sum_{i=1}^{m} \alpha_i^* u_i.K(x, x_i) + w_0^*$$

où les α_i^* sont les solutions de :

$$\begin{cases} Max_\alpha \left\{ \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j u_i u_j K(x_i, x_j) \right\} \\ \sum_{i=1}^m \alpha_i u_i = 0 \end{cases}$$

Les fonctions noyau couramment utilisées sont :
- le noyau linéaire $K(x, x_i) = x * x_i$
- le noyau fonction de base radiale (RBF) $K(x, x_i) = \exp(-\gamma \|x - x_i\|^2)$
- le noyau polynomial $K(x, x_i) = (x * x_i + c)^2$
- le noyau sigmoïde $K(x, x_i) = \tanh(x * x_i + c)$

La figure 1.9 montrent des exemples de frontières de décision obtenues avec différents types de noyaux.

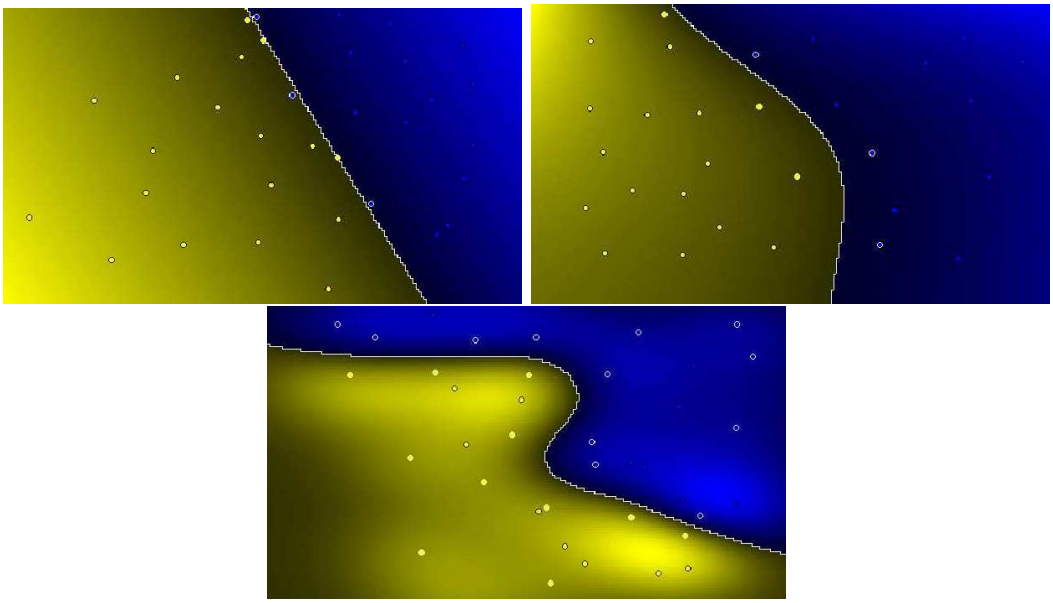


FIG. 1.9 – Frontières de décision obtenues par trois SVM à noyau linéaire, polynomial et RBF.

Remarquons qu'à l'exception du noyau linéaire, tous les noyaux possèdent un paramètre. Selon les comparatifs [Liu 02a, Liu 04] sur un problème de reconnaissance de chiffres manuscrits, les noyaux RBF donnent les meilleurs résultats.

Les SVM multiclasses

Rappelons qu'initialement, les SVM sont des classifieurs à deux classes. Bien que certains travaux cherchent à rendre le problème SVM multiclasses [Guermeur 00, Weston 99], le problème du multiclasses est généralement traité par combinaison de classifieurs binaires. Pour un problème à p classes, il existe deux catégories de méthodes de combinaison [Hsu 02a] :

- La première approche consiste à utiliser p classifieurs «un contre tous» permettant de faire la discrimination de chacune des classes contre toutes les autres. La règle de décision utilisée dans ce cas est généralement le maximum, ou l'on affecte au point inconnu la classe associée au SVM dont la sortie est la plus grande.
- L'autre approche consiste à mettre en œuvre $p(p-1)/2$ classifieurs «1 contre 1» pour chaque paire de classes possible. On attribue à un élément la classe majoritaire parmi les $p(p-1)/2$ fonctions de décision.

Selon [Hsu 02a], les deux approches offrent des performances similaires. Signalons les travaux de Platt [Platt 00] qui propose une méthode efficace pour la production de probabilité *a posteriori* avec une approche «1 contre 1» organisée en arbre de décision (DAG).

Conclusion sur les SVM

Plusieurs comparatifs [Liu 02a, LeCun 98] montrent que les SVM offrent des performances très intéressantes pour la reconnaissance de caractères manuscrits grâce à une grande capacité à généraliser. Les SVM n'offrant pas par défaut la possibilité de fournir des probabilités *a posteriori*, certains travaux ont proposé des approximations pour les obtenir à partir de la fonction non bornée $h(x)$. Dans [Platt 99b], les probabilités sur un problème à deux classes sont obtenues en appliquant une sigmoïde sur $h(x)$. Une extension au problème multiclasse a été proposé dans [Milgram 05] pour les deux types d'approches «un contre tous» et «un contre un».

L'apprentissage des SVM pose toutefois le problème du réglage d'au moins deux paramètres : le paramètre de pénalité C et le paramètre du noyau (γ dans le cas d'un noyau RBF). Le réglage de ces paramètres est souvent appelé «sélection de modèle» [Gold 03] et est généralement réalisé par essai/erreur sur une base de test ou par validation croisée.

L'autre inconvénient des SVM concerne leur comportement en haute dimension. Si théoriquement les SVM supporte bien les hautes dimensions, dans la pratique on voit leurs performances chuter. C'est la raison pour laquelle des méthodes de sélection de variables sont souvent mises en œuvre [Huang 06, Guyon 02, J.Weston 00, L.Hermes 00].

Enfin les SVM sont considérés comme lents en phase d'apprentissage comme en phase de décision [Liu 02b]. Leur rapidité en décision dépend du problème traité et du nombre de vecteurs de support conservés à l'issue de l'apprentissage. Afin de limiter ce nombre, il est possible d'entraîner les SVM avec un terme de régularisation pénalisant la conservation d'un nombre important de vecteurs de support [Guigue 05].

1.2.3.3 Les réseaux de neurones

Les réseaux de neurones ont connu un grand succès à partir des années 90, notamment grâce à la mise au point d'un algorithme d'apprentissage efficace et facile à mettre en œuvre : la rétropropagation du gradient [Bishop 95, Zhang 00,

Lecun 87]. Il existe de nombreux type de réseaux de neurones mais les deux types les plus utilisés en reconnaissance de caractères sont les perceptrons multicouches [Gader 96, Cao 97, Wong 02, LeCun 89] et les réseaux à fonctions de base radiales [Hwang 97, Gilloux 95, Lemarie 93].

Dans un modèle statistique, la connaissance, c'est-à-dire la distribution des classes, est représentée par un modèle mathématique (mélange de gaussiennes par exemple) dont les paramètres doivent être estimés. Ces modèles constituent une limitation puisqu'ils ne seront jamais qu'une approximation de la «forme» des classes. Selon Lecun [Lecun 87], le modèle connexionniste surmonte ce problème en représentant la connaissance sous la forme d'un réseau d'unités élémentaires reliées par des arcs pondérés. C'est dans ces connexions que réside la connaissance, et celle-ci peut prendre une forme plus variée qu'avec un modéle mathématique prédéfini. Le but est d'apprendre au réseau à fournir les sorties voulues pour un ensemble de valeurs d'entrée. Pour cela, on se base sur un très grand nombre d'exemples qui permettent d'ajuster les paramètres - les poids des connexions - de manière à obtenir les sorties désirées en fonction des entrées.

Il existe de nombreuses topologies de réseaux de neurones (voir figure 1.10) :

- Les réseaux multicouches : ils sont organisés en couches, chaque neurone prend généralement en entrée les sorties de tous les neurones de la couche inférieure. Ils ne possèdent pas de cycles ni de connexions intra-couche. On définit alors une «couche d'entrée», une «couche de sortie», et n «couches cachées». Ce type de réseau est très répandu, du fait de son apprentissage aisé réalisé par l'algorithme de rétropropagation du gradient.
- Les réseaux à connexions locales : on reprend la même structure en couche que précédemment, mais avec un nombre de connexions limité : un neurone n'est pas forcément connecté à tous les neurones de la couche précédente.
- Les réseaux à connexions récurrentes : on a toujours une structure en couches, mais avec des retours ou des connexions possibles entre la sortie et l'entrée des neurones d'une même couche.
- Enfin dans les réseaux à connexions complètes, tous les neurones sont interconnectés, comme par exemple dans le modèle de Hopfield et la machine de Boltzmann.

Signalons que suivant la topologie et l'algorithme d'apprentissage utilisés, les réseaux de neurones peuvent être rangés parmi les classifieurs statistiques. Les réseaux multicouches entrainés avec une erreur quadratique (RBF et MLP) peuvent être considérés comme des classifieurs statistiques discriminants [Jain 00].

Le neurone formel est l'unité élémentaire des réseaux. Il effectue la somme pondérée de ses entrées, et la soumet à une fonction non linéaire dérivable (voir figure 1.11). Pour un neurone formel possédant n entrées x_i, le neurone effectue la somme pondérée y puis «active» sa sortie z à l'aide d'une fonction non linéaire f :

$$y = \sum_{i=1}^{n} w_i x_i \qquad \text{et} \qquad z = f(y) = f(\sum_{i=1}^{n} w_i x_i)$$

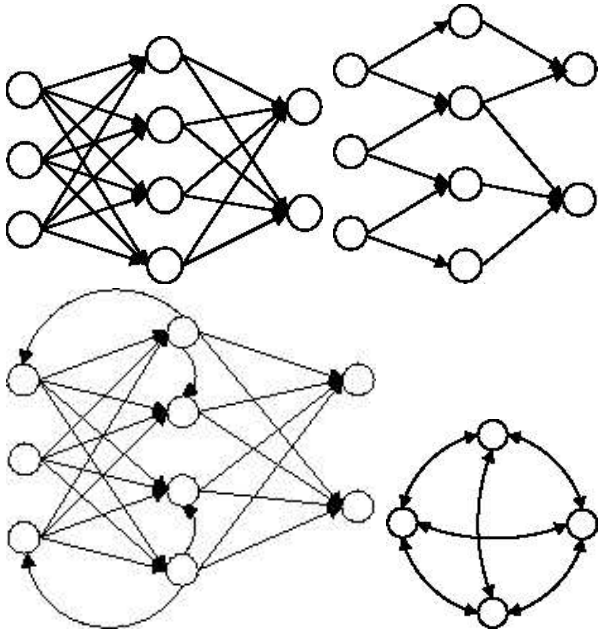


FIG. 1.10 – Différentes topologies de réseau de neurones : réseaux multicouches, à connexions locales, à connexions récurrentes et à connexions complètes.

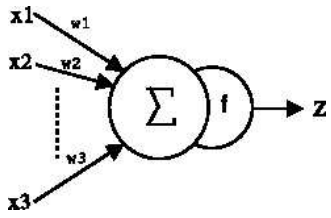


FIG. 1.11 – Neurone formel.

La fonction sigmoïde est la fonction linéaire la plus souvent utilisée, mais il existe beaucoup d'autres fonctions : tangente hyperbolique, fonction de heavyside, fonction gaussienne, etc.

La rétropropagation du gradient

Les réseaux multicouches «Feed-forward» (MLP, RBF) doivent en partie leur succès à l'existence d'un algorithme d'apprentissage efficace et facile à implémenter : la rétropropagation du gradient.

La rétropropagation du gradient consiste à propager «à l'envers» (de la couche de sortie vers la couche d'entrée) l'erreur obtenue sur les exemples de la base d'apprentissage. On utilise pour cela l'erreur quadratique, i.e. le carré de la différence

entre ce qu'on obtient et ce qu'on désire. Si on calcule la dérivée partielle de l'erreur quadratique par rapport aux poids des connexions (le «gradient»), il est possible de déterminer la contribution des poids à l'erreur générale, et de corriger ces poids de manière à se rapprocher du résultat souhaité. La correction se fait par itérations successives en corrigant plus ou moins fortement les poids par l'intermédiaire d'un coefficient η.

À l'issue d'un certain nombre d'itérations, lorsque qu'on est satisfait du classement des exemples de la base d'apprentissage, les poids obtenus définissent ainsi des frontières entre les classes.

Considérons le réseau à une couche cachée de la figure 1.12. Il est défini par :

- Une couche d'entrée à m cellules d'entrées $x_i = e_i$ (il ne s'agit pas de neurones, ces cellules présentent simplement les entrées e_i au réseau).
- Une couche cachée à n neurones d'activation y_j
- Une couche de sortie de p neurones d'activation z_k
- $n \times m$ connexions entre la couche d'entrée et la couche cachée, pondérées par les poids v_{ji}
- $m \times p$ connexions entre la couche cachée et la couche de sortie, chacune pondérée par w_{kj}

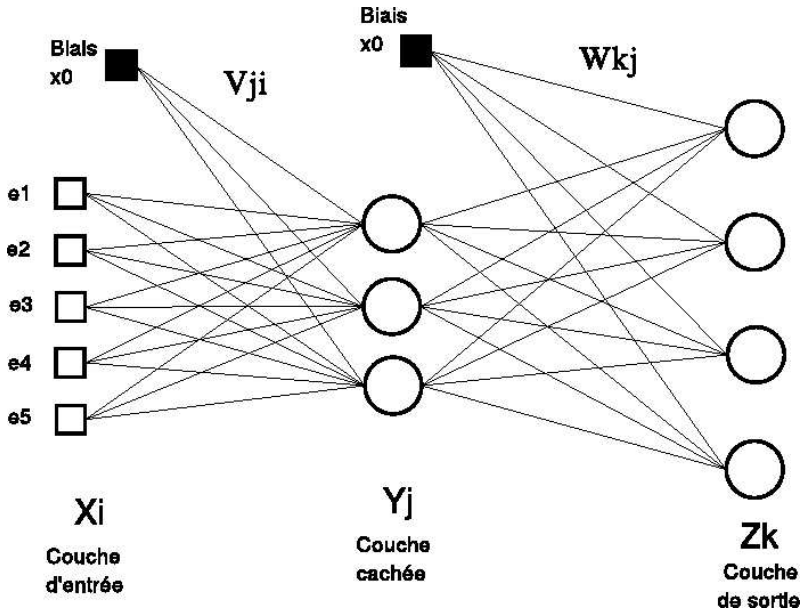


FIG. 1.12 – Exemple de réseau MLP à une couche cachée avec 5 entrées, 3 neurones dans la couche cachée, et 4 sorties.

La rétropropagation du gradient est alors effectuée à l'aide de l'algorithme 1.2.3.3.

Précisions concernant les réseaux de neurones multicouches

Algorithme 1 Algorithme de rétropropagation du gradient

ÉTAPE 1 : Initialisation des poids des connexions. Ces poids sont choisis aléatoirement.

ÉTAPE 2 : Propagation des entrées. Les e_i sont présentées à la couche d'entrée : $x_i = e_i$, puis propagées vers la couche cachée :

$$y_j = f(\sum_{i=1}^{m} v_{ij} x_i + x_0)$$

puis de la couche cachée vers la sortie :

$$z_k = f(\sum_{i=1}^{n} w_{kj} y_j + y_0)$$

Les valeurs x_0 et y_0 sont des biais : des scalaires et non des sorties de la couche précédente.

ÉTAPE 3 : Rétropropagation de l'erreur. Pour chaque exemple de la base d'apprentissage appliqué en entrée du réseau, on calcule son erreur sur les couches de sorties, c'est-à-dire la différence entre la sortie désirée s_k et la sortie réelle z_k :

$$E_k = z_k(1 - z_k)(s_k - z_k)$$

On propage cette erreur sur la couche cachée ; l'erreur de chaque neurone de la couche cachée est donnée par :

$$F_j = y_j(1 - y_j) \sum_{k=1}^{p} w_{kj} E_k$$

ÉTAPE 4 : Correction des poids des connexions. Il reste à modifier les poids des connexions. Entre la couche d'entrée et la couche cachée :

$$\begin{cases} \Delta w_{kj} = \eta y_j E_k \\ \Delta x_0 = \eta E_k \end{cases}$$

Entre la couche cachée et la couche de sortie :

$$\begin{cases} \Delta v_{ji} = \eta x_i F_j \\ \Delta y_0 = \eta F_j \end{cases}$$

η étant un paramètre à fixer.

BOUCLER à l'étape 2 jusqu'à un critère d'arrêt à définir.

Si l'algorithme de rétropropagation du gradient est efficace, il est difficile de parfaitement contrôler le comportement du réseau durant l'apprentissage [Lecun 87]. En effet, la configuration de départ (valeurs aléatoires des poids du réseau) et l'ordre de présentation des exemples influent sur la solution finale. De plus, il n'existe pas

de résultat théorique concernant le dimensionnement des couches cachées. Une autre incertitude concerne la possibilité de tomber dans un minimu local, en particulier si le réseau a été mal configuré. Enfin le paramètre η doit être correctement fixé pour obtenir un apprentissage à la fois rapide et précis.

Problème du dimensionnement : un inconvénient des MLP est qu'on ne peut pas connaitre *a priori* les dimensions du réseau pour un problème donné. L'expérience montre qu'il n'est pas nécessaire d'avoir plus d'une couche cachée : Liu [Liu 02c] montre par exemple qu'il obtient de meilleurs résultats avec un réseau à une couche cachée de 150 neurones plutôt qu'avec deux couches cachées de 65 puis 39 neurones sur un problème de reconnaissance de lettres segmentées manuscrites. En revanche, on ne peut pas déterminer *a priori* le nombre de neurones de la couche cachée nécessaire à un problème donné. Certaines heuristiques communément admises avancent les chiffres de (nb d'entrées + nb de sorties)/2 ou $\sqrt{\text{nb d'entrées * nb de sorties}}$, sans toutefois prendre en compte la difficulté du problème.

Le paramètre η : le paramètre η permettant d'ajuster les poids des connexions est également délicat à déterminer ; il est nécessaire de régler ce paramètre de manière empirique mais cela impose d'effectuer plusieurs apprentissages souvent longs. Il existe aussi des algorithmes permettant de régler dynamiquement la valeur de η [Bishop 95]. On peut par exemple faire décroître η au fur et à mesure de l'apprentissage, soit en fonction de la quantité d'erreur, soit en fonction du nombre d'itérations. L'algorithme *line search* [Bishop 95] propose également un η dynamique déterminant la valeur optimale à chaque itération.

Problème du sur-apprentissage : un autre paramètre doit être trouvé empiriquement : le nombre d'itérations lors de la phase d'apprentissage. Celui-ci est primordial puisqu'il apparaît au bout d'un certain nombre d'itérations le phénomène bien connu du «sur-apprentissage» durant lequel le MLP commence à apprendre par cœur les exemples de la base d'apprentissage et perd sa capacité à généraliser. En utilisant une base de validation, on peut calculer l'erreur de généralisation du réseau en fonction du nombre d'itérations pour choisir la configuration des poids qui minimise l'erreur.

Problème des minima locaux : comme pour toute méthode à gradient, l'algorithme de rétropropagation du gradient peut tomber dans un minimum local. L'algorithme n'étant initialement pas prévu pour sortir de ces *minima*, un terme d'inertie ou une composante aléatoire peuvent être ajoutés à la correction des poids afin d'explorer d'autres parties de l'espace des paramètres. Les algorithmes génétiques peuvent également être appliqués pour réaliser l'apprentissage des réseaux en évitant les *minima* locaux [Lee 96].

L'expérience montre toutefois que malgré tous ces inconvénients, les MLP ont permis d'obtenir des performances très intéressantes, en particulier pour la reconnaissance de caractères manuscrits [Morita 02, Leroux 97, Knerr 97, Gader 96]. Les MLP sont également largement utilisés pour leurs nombreuses qualités : ils sont

très rapides en phase de décision[2] et supportent très bien les hautes dimensions. Un autre avantage des réseaux multicouches est l'interprétation probabiliste de ses sorties. Selon Bridle [Bridle 90], la fonction *softmax* permet à un réseau multicouche entrainé avec un critère des moindres carré de générer des probabilités *a posteriori*. Pour un réseau à k sorties s, la fonction *softmax* redéfinit les k sorties corrigées $sc_j \in \{sc_1, \ldots, sc_k\}$ par la relation :

$$sc_j = \frac{\exp s_j}{\sum_{i=1}^{k} \exp s_i}$$

L'obtention des probabilités *a posteriori* est toutefois conditionnée par les contraintes suivantes :

- Le réseau doit être suffisamment bien dimensionné (la couche cachée doit posséder suffisamment de neurones, ce qui reste difficile à définir).
- La représentation des données doit être la même dans la base d'apprentissage et dans le problème réel.
- Le nombre d'exemples dans la base d'apprentissage doit ête infini. Dans la pratique cette contrainte n'est évidemment jamais vérifiée. Pour un problème à m caractéristiques, il est communément admis qu'on dispose de suffisamment d'exemples avec m^2 exemples par classe.

L'interprétation probabiliste des sorties permet ainsi un couplage intéressant avec les modèles de Markov cachés [Morita 02, Knerr 97]. Nous reviendrons sur ce type de méthodes en section 1.5.3.

1.2.4 Combinaison de classifieurs

Si les comparatifs [Liu 04, LeCun 95, Liu 02b, Liu 02a] semblent montrer que les perceptrons multicouches et les machines à vecteurs de support avec noyau gaussien donnent les meilleures performances, une idée intéressante apparue dans les années 80 consiste à combiner les classifieurs afin de bénéficier de leur éventuelle complémentarité [Zouari 02, Rahman 03]. La combinaison de classifieurs a été utilisée avec succès en reconnaissance de formes et en particulier de caractères manuscrits [Huang 95, Kittler 98, Ho 94, Xu 92, Rahman 97]. Il existe trois schémas de combinaison de classifieurs [Rahman 03] (voir figure 1.13) :

- La combinaison parallèle dans laquelle le caractère à reconnaître est présenté à plusieurs classifieurs indépendants dont les sorties sont combinées pour donner la décision finale.
- La combinaison séquentielle de classifieur où les classifieurs sont disposés en niveaux successifs de décision permettant de réduire progessivement le nombre de classes possibles. [Francesconi 01, Giusti 02].
- Les approches hybrides consistent à combiner les architectures séquentielles et parallèles. Ce type d'approches est généralement dédié à un problème précis et est difficilement généralisable [Kim 00, Vuurpijl 03].

[2] voir le comparatif de [Liu 02a] sur les temps de traitements de plusieurs classifieurs sur des chiffres manuscrits

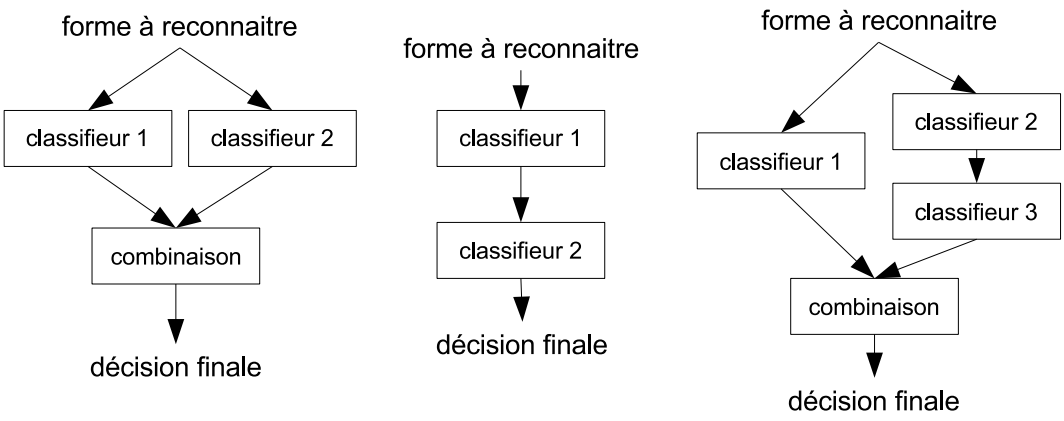


FIG. 1.13 – Les 3 schémas de combinaison de classifieurs : approche parallèle, approche séquentielle et approche hybride.

Parmi ces trois approches, la combinaison parallèle est la plus utilisée car contrairement aux deux autres approches, elle ne nécessite pas de connaître précisément le comportement des classifieurs. Les approches parallèles sont ainsi plus facilement généralisables et plus simples à mettre en œuvre puisqu'elles nécessitent simplement de développer une étape de combinaison des sorties. On peut distinguer plusieurs types de combinaisons des sorties suivant que l'on procède à une fusion ou à une sélection des sorties. Dans les méthodes de sélection, on cherche à sélectionner le meilleur sous-ensemble de classifieurs en fonction des résultats des classifieurs simples. La décision finale peut être prise soit par le meilleur classifieur uniquement [Giacinto 00], soit par plusieurs classifieurs [Jacobs 91, Lecce 00]. Dans les approches par fusion, un schéma de combinaison fixé prend en compte les décisions de tous les classifieurs. Dans ce cas, le schéma de combinaison peut être déterminé avec ou sans apprentissage.

Les méthodes de combinaison avec apprentissage déterminent via une base d'apprentissage supplémentaire les paramètres de la combinaison. Une des méthodes les plus répandues consiste à utiliser un réseau de neurones dont les entrées sont les sorties des classifieurs simples [Chi 96, Hao 97, Prevost 98].

Les méthodes de combinaison sans apprentissage, bien que sous-optimales [Duin 02], ne nécessitent aucune donnée supplémentaite, et se révèlent très simple à mettre en œuvre. Cette approche est la plus répandue, en particulier sur les problèmes de reconnaissance de caractères [Ho 94, Kimura 91].

Nous venons de présenter les méthodes de reconnaissance d'entités isolées. Nous décrivons maintenant les approches de la littérature pour la reconnaissance d'entités composées de plusieurs caractères : nous commençons par décrire sommairement les approches pour la reconnaissance de mots, puis nous décrivons plus précisément les

méthodes de reconnaissances de chiffres liés et de séquences numériques, ces dernières nous intéressant plus particulièrement.

1.3 Reconnaissance de mots

Il existe deux stratégies possibles pour la reconnaissance des mots manuscrits : les approches *globales* qui considèrent le mot dans son ensemble sans chercher à identifier chacune des lettres qui le compose ; qu'on oppose aux approches *analytiques* qui cherchent à découper le mot en lettres afin de le reconnaître.

Dans les approches globales, des caractéristiques sont extraites sur le mot entier afin de calculer une distance à des modèles de mots [Koerich 03b, Powalka 97]. Ces approches présentent l'inconvénient de subir la variabilité des mots, plus importante encore que celle observée sur les lettres. Ainsi, elles requièrent des bases de mots conséquentes. Elles sont, de plus, peu discriminantes pour des mots différents dont la forme est proche, ce qui les limite à des applications à lexique réduit (cas des montants numériques de chèques [Impedovo 97a, Knerr 97, Leroux 97]), ou à des étapes de pré ou post-traitement visant soit à filtrer une partie du lexique [Annick 94, Madhvanath 93], soit à vérifier les solutions d'une approche analytique [Powalka 94].

Les approches analytiques visent à reconnaître les mots en identifiant les lettres qui le composent. Une étape de segmentation est donc nécessaire afin de déterminer les limites entre les lettres. Cette tâche est particulièrement délicate du fait de l'absence de segmentation idéale : les limites entre caractères sont parfois difficiles à déterminer même pour un être humain. Il existe deux types d'approches analytiques suivant que l'on effectue une segmentation *explicite* ou *implicite*.

Les approches à segmentation explicite [Gader 97, Kimura 94, Knerr 97, Koch 04] utilisent des algorithmes de segmentation généralement basés sur les contours ou les profils [Bozinovic 89, El-Yacoubi 99, Gader 97, Kim 97b, Kimura 94] pour proposer des hypothèses de points de segmentation. Les différentes hypothèses sont généralement organisées en treillis à plusieurs niveaux (voir figure 1.14) et évaluées par le moteur de reconnaissance de caractères. On parle alors de stratégie de *segmentation-reconnaissance*.

Les approches à segmentation implicite [Mohammed 96, Senior 98, Cho 95, Morita 06] considèrent tous les points du tracé comme des points de segmentation potentiels à l'aide d'une fenêtre glissante successivement décalée de 1 à quelques pixels. Des caractéristiques de bas niveau sont extraites de chaque fenêtre et sont soumises à un classifieur dynamique (HMM, TDNN, réseaux récurrents) qui prend une décision globale de segmentation et de reconnaissance sur l'ensemble du mot. Si ce type d'approches solutionne en partie le choix difficile des points de segmentation, il semble qu'elles sont moins discriminantes que les approches analytiques mettant en œuvre des classifieurs statistiques (MLP, SVM, etc.).

Pour bénéficier des avantages des deux types d'approches, on a vu apparaître des stratégies dites *neuro-markoviennes* où les observations fournies par un classifieur de type réseau de neurones alimentent un modèle de Markov caché [Bengio 95,

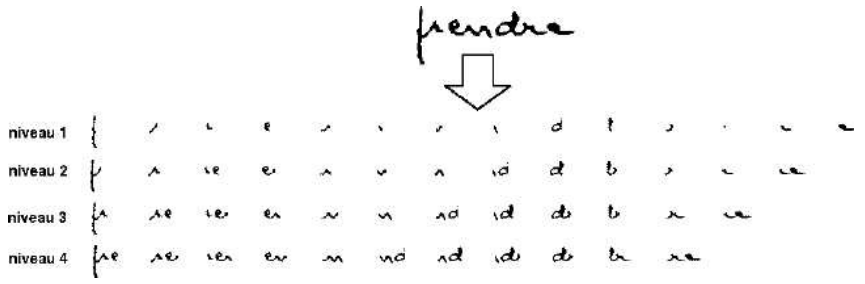


FIG. 1.14 – Représentation des hypothèses de segmentation par un treillis à 4 niveaux de regroupement.

Gilloux 95, Morita 06, Kim 00, Koerich 02].

Les systèmes de reconnaissance de mots bénéficient toujours d'un lexique plus ou moins important selon le contexte de l'application. Il existe deux stratégies permettant de prendre en compte ces connaissances lexicales. Le premier type d'approches, dites *dirigées par le lexique*, fait intervenir le lexique le plus tôt possible dans les traitements en mettant en concurrence les modèles de mot. Durant la reconnaissance, on va donc chercher à aligner les hypothèses de segmentation-reconnaissance sur chacun des modèles de mots du lexique pour ne proposer que des hypothèses de mots valides au sens du lexique. L'alignement des observations se fait par programmation dynamique à l'aide des algorithmes de Viterbi [Forney 73] ou Forward [Rabiner 90]. Le deuxième type d'approche, dites *non dirigées par le lexique*, n'exploite les connaissances lexicales qu'à l'issue de la phase de reconnaissance [Lopresti 00, Manke 96, Oommen 97]. Les hypothèses de segmentation-reconnaissance sont comparées aux mots du dictionnaire par le biais d'une distance d'édition.

1.4 Reconnaissance de chiffres liés

La reconnaissance de chiffres liés a fait l'objet de nombreuses recherches, dans le cadre du traitement automatique de chèques et des codes postaux. La reconnaissance de chiffres liés suppose d'avoir localisé au préalable une composante comme constituée de plusieurs chiffres, ce qui est une opération *a priori* très délicate en vertu du paradoxe de Sayre. L'estimation du nombre de chiffres d'une composante peut être effectuée par une analyse des contours [Pal 01] ou des dimensions de la boîte englobante de la composante [Britto 02]. L'identification de chiffres liés peut également être effectuée par une analyse de Fourier [Zhu 99] ou être fournie par le contexte de l'application : dans [Morita 06], les chaînes de chiffres du jour ou d'une année comportent toujours 2 ou 4 chiffres.

Du fait de l'absence de lexique, il existe peu d'approches strictement globales (ou holistiques) pour la reconnaissance de chiffres liés. Ceci s'explique par le très grand

nombre de classes qu'engendrerait une telle approche (100 classes pour les doubles chiffres, 1000 pour les triples, etc.), et la difficulté d'obtenir suffisamment d'exemples pour chacune de ces classes. Dans [Wang 00], une approche holistique pour la reconnaissance de doubles chiffres est présentée, reposant sur une extension d'un moteur de reconnaissance de chiffres isolés à un problème à 100 classes, à l'aide d'un classifieur KPPV. Les performances ne sont toutefois données que pour les classes dont l'effectif dans la base d'apprentissage est suffisant, ce qui biaise les résultats. Dans [Zhou 05], une autre approche basée sur une combinaison de classifieurs SVM à deux classes est présentée. Là aussi, seules les classes des paires de chiffres les plus représentées dans la base MNIST SD19 sont testées. Les approches globales peuvent aussi être utilisées en complément d'une approche analytique [Wang 99]. La majorité des approches pour la reconnaissance de chiffres liés est donc analytique. Les méthodes analytiques procèdent à une segmentation de la composante en chiffres.

Le plus souvent, la segmentation utilisée pour les chiffres liés est une segmentation explicite [Lu 99, Morita 06, Pal 01, Shi 97, Wang 00, E.Ashraf 03, Sadri 04, Chen 00, Oliveira 00, Kim 02b], même si la combinaison de segmentations explicite et implicite est parfois utilisée [Zhou 00]. Dans le cas d'une segmentation explicite, il s'agit de déterminer le ou les meilleurs chemins de segmentation en fonction du nombre de chiffre de la composante. Ceux-ci sont généralement trouvés à partir d'une analyse locale du tracé.

Deux stratégies permettent généralement de choisir la meilleure hypothèse de segmentation. La première stratégie consiste à évaluer la qualité des chemins de segmentation sans reconnaissance [Kim 02b, Pal 01, Lu 99]. À l'issue de l'évaluation, un classement des chemins de segmentation est fourni et seuls les deux chiffres résultant de la meilleure segmentation sont soumis au moteur de reconnaissance chiffre. L'inconvénient de ce type d'approche réside dans le choix délicat du meilleur chemin qui ne peut être remis en cause par la suite. Plus répandue, la deuxième stratégie consiste à appliquer une stratégie de segmentation-reconnaissance en soumettant toutes les hypothèses de segmentation au moteur de reconnaissance chiffres [Oliveira 00, Morita 06]. Une décision sur l'ensemble de la composante est prise à l'aide d'un algorithme de programmation dynamique.

1.5 Reconnaissance de séquences numériques

Portée par les applications industrielles de reconnaissance de montants numériques de chèques et de codes postaux dans les blocs adresses, la reconnaissance de séquences numériques est certainement l'un des domaines les plus abouti en reconnaissance d'écriture manuscrite. Elle consiste à reconnaître tous les chiffres et éventuellement identifier les entités non numériques (séparateur, symbole, virgule, etc.) d'une séquence numérique déjà localisée. Les deux applications les plus connues sont la reconnaissance de montants numériques de chèques [Dzuba 97a, Impedovo 97a, Kim 97a, Knerr 97, Zhang 02] et la reconnaissance de code postal dans les adresses [LeCun 89, Dzuba 97b, Liu 04, Cohen 94, Pfister 00].

Mais certains travaux ont également concerné la reconnaissance de dates sur les chèques [Morita 02, Morita 06, Xu 03]. Dans ce cas, les séquences traitées ne sont pas toujours exclusivement numériques puisque le champ «mois» peut être écrit en toutes lettres. Pour traiter ce cas de figure, ces approches sont couplées avec un moteur de reconnaissance de mots.

La reconnaissance de séquences numériques diffère de la reconnaissance des chiffres liés car le nombre de chiffres de la séquence est généralement inconnu. Notons toutefois le cas des codes postaux où le nombre de chiffre de la séquence est souvent fixe (5 chiffres en France, 5 ou 9 chiffres aux Etats Unis, etc.). Elle diffère également des méthodes pour la reconnaissance de mots dirigée par le lexique puisqu'aucun lexique n'est disponible. En revanche, on peut comparer les approches mises en œuvre pour le problème de reconnaissance de séquences numériques avec les approches de reconnaissance de mots non dirigée par le lexique. Nous avons vu dans la section 1.4 que les moteurs de reconnaissance de chiffres offraient désormais des performances intéressantes ; le véritable enjeu de la reconnaissance de séquences numériques se situe donc plutôt dans la capacité à localiser les chiffres, en particulier lorsque les séquences contiennent des chiffres liés.

Dans la mesure où la reconnaissance de séquences numériques ne peut bénéficier de l'apport d'un lexique, chaque chiffre doit être reconnu sans pouvoir bénéficier du résultat de la reconnaissance des chiffres contigüs. Le type d'approche utilisé est donc nécessairement «analytique», en opposition aux méthodes dites «globales» qui considèrent la séquence à reconnaître comme une seule entité. Toutes les approches de la littérature procèdent donc à une localisation des chiffres par un processus de segmentation. Comme pour la reconnaissance de chiffres liés, la segmentation peut être implicite si tous les points du tracé sont susceptibles d'être choisis comme point de segmentation [Cavalin 06, Britto 00, Britto 03], ou explicite si un algorithme de segmentation sélectionne des points candidats à la segmentation [Xu 03, Lei 04, Koga 01, Liu 04, Oliveira 02b].

1.5.1 Approches à segmentation explicite

Les segmentations dites «explicites» ou «discrètes» effectuent une sélection des points de segmentation les plus probables par une analyse des composantes. Pour la segmentation de chiffres, on parle le plus souvent de «chemin de segmentation», contrairement à la segmentation des lettres où les «points de segmentation» sont utilisés pour segmenter les entités. Ceci est dû à la plus grande complexité des liaisons entre chiffres (voir figure 1.15). Certaines liaisons multiples ou à contact prolongé imposent de séparer les composantes selon un chemin. Les chemins de segmentation sont généralement obtenus par des points caractéristiques issus d'une analyse des contours de la forme [Pal 01, Morita 06, E.Ashraf 03, Kim 02b], du squelette ou d'un amincissement du fond [Lu 99, Sadri 04], d'une analyse en deux dimensions du tracé [Koga 01], ou d'une combinaison analyse des contours/amincissement du fond [Chen 00, Oliveira 02b]. Nous renvoyons à l'état de l'art de Trier [Trier 96] pour une revue des algorithmes de segmentation de caractères existants.

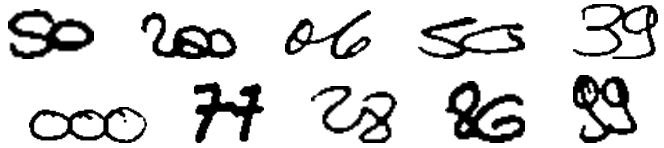


FIG. 1.15 – Types de liaisons complexes des chiffres liés : liaisons simples, doubles, ou à contact prolongé (non pontuel).

Une fois les points de segmentation potentiels identifiés, il existe deux méthodes permettant de choisir la segmentation finale de la séquence de chiffres : les méthodes dites de *segmentation puis reconnaissance* et les approches de *segmentation-reconnaissance*. Les approches de segmentation puis reconnaissance choisissent les meilleurs points de segmentation sans l'aide de la reconnaissance, alors que les méthodes de *segmentation-reconnaissance*, beaucoup plus répandues, sont basées sur l'utilisation du moteur de reconnaissance de chiffres pour valider et classer les hypothèses de segmentation.

Segmentation puis reconnaissance

Les approches de segmentation puis reconnaissance, appelées «segmentation-based» dans la littérature anglaise, visent à sélectionner les chemins de segmentation sans contrôle de la reconnaissance chiffre [Xu 03, Zhang 02, Palacios 97, Impedovo 97a]. Ce choix est réalisé soit par un tri des chemins de segmentation selon un critère évaluant la qualité de segmentation [Pal 01, Lu 99], soit par un module de vérification des hypothèses de segmentation générées [Zhang 02, Impedovo 97a]. Une fois le choix du chemin de segmentation effectué, les entités segmentées sont soumises au moteur de reconnaissance de chiffres pour fournir le résultat de reconnaissance final.

Ces approches sont assez peu utilisées en reconnaissance de séquences numériques à cause de la sélection difficile du meilleur chemin de segmentation sans reconnaissance et à l'impossibilité de remettre en cause ces choix dans la suite de la chaîne de traitement. Notons que cette stratégie entre en contradiction avec le paradoxe de Sayre. Il a été montré dans [Fujisawa 92] que les méthodes de segmentation sans reconnaissance ne pouvaient conduire à des résultats fiables. Pour pallier à ce problème, certaines approches mettent en œuvre une boucle de retour à l'issue de la reconnaissance chiffre afin d'explorer d'autres hypothèses de segmentation [Impedovo 97a, Palacios 97].

Segmentation/reconnaissance

Les approches de type segmentation/reconnaissance, également appelées approches «segmentation-free» ou «recognition-based», consistent à alterner les phases de segmentation et de reconnaissance de manière à valider les hypothèses de segmentation par la reconnaissance. Ce type d'approche est très répandu en reconnaissance de séquences numériques car il donne de bons résultats et est assez facile à mettre en œuvre [Liu 06, Liu 04, Kim 02a, Lei 04, Lethelier 95, Koga 01, Heutte 97,

Leroux 97, Oliveira 02b, Ha 98]. L'idée est de soumettre toutes les hypothèses de segmentation au moteur de reconnaissance chiffre afin de classer ces hypothèses au sens des scores de confiance du classifieur chiffre. Le postulat caché derrière cette idée est que les hypothèses de segmentation les plus proches de la réalité produisent les scores de reconnaissance les plus élevés.

Afin de ne pas rater un point de segmentation, les méthodes de segmentation-reconnaissance effectuent généralement une sur-segmentation des composantes en graphèmes, conduisant à une représentation des hypothèses de segmentation en treillis ou en graphe (voir figure 1.16). Le nombre de niveaux de regroupement des graphèmes varie suivant la méthode de segmentation utilisée.

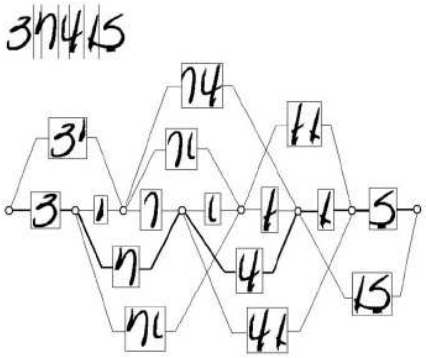


FIG. 1.16 – Hypothèses de segmentation d'une séquence numérique dans [Liu 04].

L'ensemble des hypothèses de segmentation du treillis est soumis à un classifieur chiffre pour former un treillis d'hypothèses de segmentation/reconnaissance. La reconnaissance globale de la séquence numérique est alors effectuée par la recherche du meilleur chemin au sens des scores de confiance du classifieur. Idéalement, le classifieur chiffre utilisé doit être capable de fournir une probabilité *a posteriori* $p(u_i/x)$ afin d'évaluer l'hypothèse de segmentation. C'est le cas des classifieurs de type MLP, RBF ou SVM, mais un certain nombre de classifieurs produisent une distance de la forme à la classe. Il existe alors des méthodes pour transformer ces distances en probabilités [Liu 04]. Les probabilités *a posteriori* sont utilisées pour calculer les vraisemblances de chaque chemin de segmentation/reconnaissance. Le chemin maximisant cette vraisemblance est choisi. La recherche du meilleur chemin est effectuée le plus souvent par programmation dynamique [Liu 04, Lei 04], ou par un formalisme statistique [Lethelier 95] ou une méthode de recherche de graphe [Filatov 95]. Dans certaines approches, la qualité de segmentation est évaluée et prise en compte dans le calcul du meilleur chemin [Leroux 97].

Dans les approches par segmentation-reconnaissance, le score de confiance produit par le classifieur chiffre joue donc un rôle important. En particulier, on suppose que ce score est élevé lorsqu'un chiffre bien segmenté lui est soumis, et plus faible dans le cas d'un caractère mal formé ou d'une manière générale d'une forme différente

d'un chiffre (chiffre lié, fragment de chiffre, etc.). Afin d'améliorer les capacités de rejet des classifieurs chiffre, il est possible d'entraîner le classifieur chiffre avec des formes à rejeter [Kim 02a].

Explorant tous les chemins de segmentation possibles, les méthodes de segmentation-reconnaissance sont beaucoup plus fiables que les approches de segmentation puis reconnaissance. Elles sont toutefois moins rapides du fait de l'utilisation intensive du moteur de reconnaissance de chiffres, et de la combinatoire élevée lorsque de longues séquences sont traitées. Dans [Ha 98], les auteurs présentent la combinaison d'une approche de segmentation puis reconnaissance avec une approche de segmentation/reconnaissance afin de fiabiliser les résultats.

1.5.2 Approches à segmentation implicite

Pour contourner le difficile problème du choix des points de segmentation, les approches à segmentation implicite (ou continues) considèrent tous les points du tracé comme points de segmentation potentiels. La segmentation et la reconnaissance sont réalisées conjointement, d'où le nom parfois employé de «segmentation-reconnaissance intégrée». Il s'agit de méthodes à fenêtres glissantes qui parcourent la séquence de chiffres à l'aide d'une fenêtre de taille fixe, en extrayant des caractéristiques de bas niveau. L'analyse des fenêtres est effectuée soit par un classifieur classique, soit par des modèles dynamiques tels que les modèles de Markov cachés ou les réseaux de neurones à convolution, qui déterminent la classe d'appartenance de chaque fenêtre en fonction des fenêtres voisines.

Méthodes à fenêtre glissante

Les méthodes à fenêtre glissante utilisent un classifieur «classique» qui se déplace sur la séquence et prend en entrée une fenêtre d'observation centrée pour classer l'élément courant. Pour une fenêtre de largeur $w = 2d + 1$ (d éléments précédents, 1 élément courant et d éléments suivants), il s'agit de déterminer $y_{i,t}$ avec la fenêtre $< x_{i,t-d}, \ldots, x_{i,t}, \ldots, x_{i,t+d} >$. Les méthodes à fenêtre glissante permettent ainsi de prendre en compte le contexte au niveau des observations.

Si ces méthodes prennent en compte le contexte au niveau des observations, elles ne permettent pas de prendre en compte les corrélations entre les étiquettes. D'où l'introduction des méthodes à fenêtre glissante récurrente.

Les méthodes à fenêtre glissante récurrente sont basées sur le même principe que les méthodes à fenêtre glissante simple, mais les sorties précédentes $y_{i,t-d} \ldots y_{i,t-1}$ sont utilisées par le classifieur en plus de la fenêtre $< x_{i,t-d}, \ldots, x_{i,t}, \ldots, x_{i,t+d} >$ pour déterminer $y_{i,t}$. La récurrence permet de prendre en compte le contexte au niveau des étiquettes.

Ces méthodes à fenêtre glissante récurrente ont le plus souvent été mises en oeuvre en utilisant des réseaux de neurones, en connectant les sorties du réseau aux entrées de la couche cachée (recurrent neural network)

Ce type de réseau a été utilisé dans de nombreux domaines tels que la reconnaissance de codes postaux manuscrits [LeCun 89], la reconnaissance de la parole

[Pérez-Ortiz 01] ou la catégorisation de textes [Wermter 99].

Approches basées sur les modèles de Markov cachés

Depuis les années 70, les modèles de Markov cachés (Hidden Markov Model : HMM) ont été utilisés avec succès, en particulier en reconnaissance de la parole [Rabiner 90, Morgan 93] et en reconnaissance de l'écriture pour la reconnaissance de mots [El-Yacoubi 02], mais aussi de séquences numériques [Britto 03, Cai 99, Procter 98, Ha 98, Cavalin 06]. Ils permettent une modélisation probabiliste efficace et possèdent des algorithmes d'apprentissage automatique performants. Étant prévus pour la modélisation de signaux à une dimension, ils peuvent être appliqués à l'écriture manuscrite (signal à deux dimensions) par le biais d'une fenêtre glissante décalée horizontalement sur la séquence à reconnaître.

Un modèle de Markov caché est un processus doublement stochastique, constitué d'un processus sous-jacent non observable, qui peut être déduit au travers d'un second processus stochastique qui produit des séquences d'observations. Dans les méthodes de reconnaissance de séquences numériques par HMM, on cherche à modéliser la séquence numérique par des modèles de Markov cachés. La couche cachée du modèle est illustrée par la séquence d'étiquettes de chiffres constituant la séquence, et la couche observable correspond à une séquence d'observations que l'extraction de caractéristiques fournira à partir de la fenêtre glissante.

Dans la modélisation par HMM, le processus caché est constitué d'un jeu d'états interconnectés par des transitions dotées chacune d'une distribution de probabilité. Le processus observable consiste en un jeu de sorties (observations), qui peuvent être émises par chaque état selon une fonction de densité de probabilité. On définit donc deux matrices pour décrire le modèle : une matrice de probabilités de transitions entre les états et une matrice des probabilités d'observation des symboles.

Les modèles de Markov cachés peuvent être discrets si les observations appartiennent à un alphabet fini de symboles, ou continus si les observations sont continues. En reprenant le formalisme de Rabiner [Rabiner 90], un modèle de Markov caché discret se définit donc par les éléments suivants :

- Un ensemble de N états $S_1, S_2, \ldots, S_N$.
- M, le nombre de symboles distincts par état. Soit V l'ensemble de ces symboles : $V = \{v_1, \ldots, v_M\}$
- La matrice des probabilités de transition entre les états $A = \{a_{ij}\}$. Si q_t désigne l'état courant au temps t, on a :

$$a_{ij} = P(q_{t+1} = S_j \mid q_t = S_i), \quad 1 \leq i, j \leq N$$

- La distribution de probabilité d'observation des symboles à l'état j, $b_j(k) = P(O_t \mid q_j)$, où

$$b_j(k) = P[v_k \text{ en } t | q_t = S_j], 1 \leq j \leq N, 1 \leq k \leq M$$

- La matrice des distributions des états initiaux π :

$$\pi_i = P(q_1 = S_i), \quad 1 \leq i \leq N$$

On définit donc complètement un HMM en spécifiant les deux paramètres du modèle : N et M, ainsi que les trois matrices de probabilité A, B et π. On note ce modèle $\Lambda = (A, B, \pi)$

Pour les modèles de Markov continus, les probabilités d'émission des symboles $b_j(k)$ sont modélisées soit par des mélanges de gaussiennes dont les paramètres sont estimés lors de l'apprentissage du modèle [Vinciarelli 04], soit obtenues à partir des probabilités *a posteriori* $P(q_j \mid O_t)$ fournies par un classifieur. C'est le cas des approches neuro-markoviennes (voir section 1.5.3).

Phase de décision : En décision, le problème est le suivant : étant donné la séquence d'observation $O = O_1, \ldots, O_T$ et le modèle λ, quelle est la séquence d'états $Q = q_1, \ldots, q_T$ la plus probable ? Le premier problème consiste donc à découvrir la partie cachée du modèle. La recherche de la meilleure séquence d'étiquettes au sens d'une séquence d'observations et d'un modèle est particulièrement gourmande en calculs. On utilise donc un algorithme d'optimisation appelé *algorithme de Viterbi* [Forney 73]. Cet algorithme, issu de la programmation dynamique, repose sur le principe d'optimalité suivant : le meilleur chemin pour aller de $t = 0$ à $t = N$ est composé du meilleur chemin pour aller de $t = 0$ à $t = N - 1$ et du meilleur chemin pour aller de $t = N - 1$ à $t = N$. L'algorithme de Viterbi consiste ainsi à calculer pour toutes les étiquettes et pour tous les instants t la probabilité du meilleur chemin amenant à l'état courant, compte tenu des premières observations.

Phase d'apprentissage : Lors de l'apprentissage du modèle, le problème est le suivant : comment ajuster les paramètres du modèle $\Lambda = (A, B, \pi)$ pour maximiser $P(O|\Lambda)$? Il n'existe pas de méthode analytique pour résoudre ce problème. En effet, pour un ensemble de séquences d'observations Ω constituant l'ensemble d'apprentissage, il n'existe pas de méthode optimale pour estimer directement les paramètres du modèle. On dispose cependant de méthodes itératives telles que la méthode de Baum-Welch [Rabiner 90] qui permettent d'affiner le modèle par réestimations successives jusqu'à obtention d'un modèle localement optimal. Celle-ci permet de déterminer un modèle $\Lambda = (A, B, \pi)$ qui maximise localement $P(\Omega|\Lambda)$.

Signalons que des pseudo HMM à deux dimensions pour la reconnaissance de montants numérique ont été proposés dans [Bippus 97]. Afin d'améliorer la modélisation des séquences numériques, des durées d'état différentes sont utilisées dans [Cai 99].

Grâce au cadre probabiliste qu'ils offrent ainsi qu'à l'existence d'un algorithme d'apprentissage efficace, les HMM sont un outil de modélisation de séquence performant pour la reconnaissance d'écriture. S'ils solutionnent en partie le problème de la segmentation des caractères, les HMM souffrent toutefois d'une capacité de discrimination plus faible que les méthodes de segmentation explicite mettant en œuvre un classifieur. Partant de ce constat, la combinaison des approches à segmentation explicite avec des HMM ont été exploré, soit par des approches neuro-markovienne, soit par une combinaison séquentielle des deux approches (voir section 1.5.3).

Approches basées sur les réseaux de neurones à convolution

Certaines architectures connexionnistes dites «à convolution» permettent l'analyse de séquences. Dans les réseaux de neurones multicouches «classiques» (MLP, RBF, voir section 1.2.3.3), chaque neurone est connecté à tous les neurones de la couche précédente. Dans les réseaux à convolution, un neurone est seulement connecté à un sous-ensemble de neurones de la couche précédente [Poisson 05]. Ainsi, selon Bengio [Bengio 95], on peut voir chaque neurone comme un «détecteur de caractéristique local dont la fonction est determinée par le processus d'apprentissage».

Selon [Poisson 05], il existe deux types de réseaux à convolutions : les TDNN (Time Delay Neural Network), et les SDNN (Space Displacement Neural Network). Le TDNN est un réseau à délai utilisé pour les données de nature séquentielle (une dimension), alors que l'architecture des SDNN est adaptée à des données à deux dimensions. Le SDNN est une généralisation du TDNN à une topologie 2D. Dans le cadre de la reconnaissance de l'écriture, le TDNN permet donc un décalage horizontal d'une fenêtre de hauteur la taille de la séquence, dont les pixels sont les entrées du réseau. Le SDNN permet un décalage horizontal et vertical de la fenêtre sur les caractères à reconnaître. L'apprentissage de ces réseaux à convolution est réalisé par une généralisation de l'algorithme de rétropropagation du gradient à des réseaux aux connexions locales.

Un classifieur de type TDNN est utilisé pour la reconnaissance de séquences dans [Martin 93]. Un réseau de neurones à 2 couches cachées et poids partagés est entrainé sur 11 classes (10 chiffres + non chiffre). En phase de décision, la fenêtre est déplacée exhaustivement sur l'image et une des sorties chiffre s'active lorsqu'un caractère centré lui est présenté.

Dans [Matan 92], un classifieur chiffre SDNN est utilisé pour la reconnaissance de séquences numériques. L'algorithme de Viterbi est couplé à la dernière couche du réseau pour décider de la meilleure interprétation des entrées. La segmentation est effectuée dans les couches de caractéristiques («features maps»). Un SDNN est également utilisé par Lecun [LeCun 98] pour la reconnaissance de codes postaux. Le SDNN est composé de 5 couches dont la première extrait 20 primitives différentes à plusieurs localisations différentes sur l'image d'entrée.

Les réseaux de neurones à convolution semblent très séduisants : apprentissage automatique des extracteurs de caractéristiques, abstraction de la position des caractères, résistance aux rejets. Ils restent cependant peu utilisés en reconnaissance de l'écriture manuscrite, certainement à cause de la difficulté à paramétrer de tels classifieurs. En effet, un certain nombre de paramètres doivent être réglés en plus des traditionnels paramètres des réseaux de neurones (voir section 1.2.3.3) : dimensionnement de la fenêtre et des couches de convolutions, délai.

1.5.3 Combinaison des approches

Approches neuro-markoviennes

Les approches neuro-markoviennes, aussi qualifiées d'«hybrides», visent à bénéficier des avantages des HMM et des approches à segmentation explicite. En

effet, les HMM proposent une capacité de modélisation supérieure aux méthodes à segmentation explicite, mais leur capacité de dicrimination est inférieure. D'où l'idée de coupler la modélisation efficace des HMM avec un classifieur discriminant : les réseaux de neurones.

L'idée est d'utiliser les sorties d'un classifieur neuronal comme observations continues d'un modèle de Markov caché. Dans ce cas, les probabilités *a posteriori* fournies en sortie du réseau $P(q_j|O_t)$ sont transformées par la règle de Bayes en vraisemblances normalisées $P(O_t|q_j)/P(O_t)$. Une procédure d'apprentissage itérative du système hybride est généralement mise en œuvre, où les sorties désirées du réseau de neurones sont fournies par le HMM. La rétropropagation du gradient est alors appliquée pour la mise à jour des poids du réseau.

Ces méthodes ont rencontré un franc succès pour la reconnaissance de mots [Morgan 93, Bengio 95, Gilloux 95, Knerr 98], mais aussi pour la reconnaissance de séquence numériques avec un MLP dans [Morita 06], avec un SDNN dans [Matan 92]. Ce type d'approche reste à ce jour un des moyens les plus efficaces d'allier le pouvoir discriminant des réseaux de neurones et la capacité de modélisation des séquences des modèles de Markov cachés.

Combinaison de segmentation implicite et explicite

Afin de bénéficier des avantages des méthodes implicites et explicites, il est possible d'effectuer une combinaison des deux approches. Dans les combinaisons parallèles, les deux approches sont effectuées simultanément afin de fiabiliser la reconnaissance [Ha 98]. Dans [Britto 03], une combinaison séquentielle des approches est proposée. La reconnaissance débute avec une approche à segmentation implicite qui sélectionne un certain nombre d'hypothèses de reconnaissance. Une méthode de segmentation-reconnaissance est ensuite appliquée pour lever les ambiguïtés.

1.6 Conclusion

Nous avons dressé dans cette étude bibliographique un panorama des systèmes de reconnaissance d'entités manuscrites isolées. Nous avons pu constater que grâce aux progrès dans le domaine de la classification statistique et la modélisation de séquences, les systèmes actuels offrent désormais des performances intéressantes pour la reconnaissance de caractères, de chiffres liés ou de séquences numériques. Concernant la reconnaissance de mots, les performances varient beaucoup suivant la taille du lexique, et les performances sont acceptables pour les lexiques de taille raisonnable.

Si la reconnaissance «hors contexte» d'entités manuscrites propose désormais des performances acceptables, la localisation des entités ont moins été traitées dans la littérature. Le chapitre suivant est ainsi consacré à la localisation des entités manuscrites dans les documents.

Chapitre 2

Systèmes de lecture de documents et extraction d'information

2.1 Introduction

Nous avons vu dans le chapitre précédent que les progrès faits dans le domaine de la classification statistique et la modélisation de l'écriture permettent désormais de reconnaître correctement des entités déjà localisées. Une problématique moins traitée dans la littérature est la localisation des informations manuscrites dans les documents. Pourtant, tout système réel de lecture de documents suppose une localisation de l'information à reconnaître. La localisation consiste à isoler toutes les composantes et seulement les composantes d'une entité que l'on cherche à identifier dans le cas d'un mot, d'une séquence de mots, d'une phrase ; ou la valeur dans le cas d'une séquence de chiffres ou d'un champ numérique particulier (numéro de téléphone, date, etc). La localisation se traduit donc par une étape de segmentation du document en entités distinctes.

La localisation des informations manuscrites dans les documents est un problème difficile dans la mesure où il est directement confronté au paradoxe de Sayre qui stipule que dans un problème de reconnaissance de formes, la localisation et la reconnaissance des entités ne peuvent être dissociées pour être menées correctement [Sayre 73]. Il existe plusieurs manières de contourner ce problème, en fonction des connaissances *a priori* que possède le système au sujet des documents traités. Dans le cas de documents contraints (chèques, formulaires, etc.), la connaissance d'un certain nombre d'informations *a priori* sur la structure du document permet de considérer un modèle de document suffisamment contraint pour effectuer une localisation des informations sans reconnaissance. Dans ce cas, la localisation et la reconnaissance des entités sont dissociées. En revanche, dans le cas de documents moins contraints (textes libres), les connaissances *a priori* disponibles sont trop faibles pour obtenir un modèle de document suffisamment contraint. La localisation et la reconnaissance

doivent alors être menées conjointement.

Ce second cas de figure, plus complexe, suppose la mise en place de stratégies de localisation/reconnaissance appliquées à l'ensemble du document. Il a cependant été montré que de telles stratégies posaient encore de sérieux problèmes puisqu'en l'état actuel des recherches, la lecture intégrale d'un document ne peut être effectuée de manière fiable sans connaissances *a priori* [Plamondon 00, Lorette 99]. À partir de ce constat, une stratégie alternative visant à extraire l'information des documents commence à émerger. Il ne s'agit plus de considérer une lecture intégrale du document mais plutôt d'effectuer une reconnaissance partielle visant à extraire l'information pertinente. Dans le cadre de cette thèse qui vise à localiser et reconnaître des «champs numériques» dans des documents non contraints (les «courriers entrants»), nous nous situons pleinement dans cette problématique d'extraction d'information dans des documents manuscrits non contraints.

Dans ce chapitre, nous considérons donc le problème général de la localisation des informations manuscrites dans les documents, et nous nous focalisons plus particulièrement sur le problème de la localisation et de la reconnaissance de champs numériques dans des documents manuscrits de type courrier entrant. Dans une première partie, nous définissons ce problème précis en évoquant la nature du courrier entrant et l'enjeu du traitement automatique de tels documents dans l'industrie, ainsi que la nature de l'information recherchée : les champs numériques. Afin de positionner notre problème, nous étudierons dans une seconde partie la localisation des informations manuscrites dans les systèmes existants de lecture de documents plus ou moins contraints. Lorsque les documents sont très peu contraints (cas des textes libres), nous montrerons qu'il s'agit d'un réel problème d'extraction d'information. Nous explorerons ainsi dans une troisième partie les idées générales et les méthodes de ce domaine de recherche, puis nous envisagerons la possibilité d'une adaptation des méthodes d'extraction d'information à notre problème. À partir de ces enseignements, nous envisagerons plusieurs stratégies pour le problème de localisation et de reconnaissance de champs numériques dans des documents manuscrits.

2.2 Contexte de l'étude

2.2.1 Les courriers entrants manuscrits

Les documents traités dans cette étude sont les *courriers entrants* manuscrits. Le courrier entrant désigne les documents reçus quotidiennement en grand nombre par les entreprises. Aujourd'hui, la gestion du courrier entrant dans les entreprises pose de nombreux problèmes : réception du courrier, ouverture des enveloppes, reconnaissance du type de document (formulaire ou manuscrit), identification de l'objet du courrier (changement d'adresse, réclamation, résiliation, etc.), acheminement de l'envoi vers le service compétent et enfin, prise en compte du courrier. Tout ceci représente bien évidemment un coût, tant du point de vue financier que du point de vue du temps de traitement. Dans certains cas, le nombre de documents traités dépasse le million par jour. Pour traiter cette masse de courriers, les entreprises

cherchent à automatiser le plus possible les différentes étapes du traitement : la réception et l'ouverture des enveloppes peuvent se faire de façon entièrement automatisée grâce à du matériel spécialisé ; pour éviter le flux physique des documents, tout le courrier est numérisé, facilitant ainsi l'acheminement et le traitement. Mais la dernière étape de lecture automatique du document se limite actuellement à certains types de courrier : essentiellement les formulaires, chèques, factures, etc. Les courriers manuscrits dits libres (voir figure 2.1) restent à ce jour extrêmement difficiles à traiter.

Par définition, il n'existe pas de modèle de document fixe pour ces courriers : le contenu, la mise en page et la localisation des informations sont inconnus du système de traitement automatique. Remarquons sur la figure 2.1 la diversité des mises en page et l'instabilité de la structure des courriers. Par exemple, les entêtes des courriers peuvent être placés dans la partie haute, basse, à gauche ou à droite du document, ou même être absents. L'information qui y figure fluctue : nom, prénom, adresse, numéro de client, date, numéro de téléphone. En ce qui concerne les styles d'écriture, ils diffèrent également en fonction des scripteurs : écriture cursive, scripte ou mixte ; espacement des mots plus ou moins important. Enfin les contenus des courriers varient : communication d'une pièce administrative, arrêt d'un service, résiliation de contrat, etc.

Les seules connaissances *a priori* disponibles sur ces documents sont le fait qu'ils sont écrits en langue française, et l'orientation approximativement horizontale des lignes du document. Remarquons toutefois que les lignes de texte ne sont pas parfaitement horizontales ni parallèles, et qu'elles peuvent se chevaucher.

2.2.2 Les champs numériques

Dans le cadre du traitement automatique du courrier entrant, nous proposons d'extraire un certain nombre d'informations des documents manuscrits afin d'effectuer un éventuel tri et d'automatiser au maximum leur prise en charge. Le contenu des documents étant très variable, on peut se demander quelles sont les informations utiles, susceptibles d'être extraites des courriers. Une première tâche possible est l'extraction de l'objet du courrier. En effet, les courriers possèdent toujours un objet qui, s'il n'est pas toujours clairement identifié par le mot «objet :», peut toutefois se déduire de la présence d'un certain nombre de mots clefs ("résiliation", "contrat", "changement", "adresse", etc.). Nous renvoyons à la thèse de Guillaume Koch [Koch 06] pour les travaux concernant la catégorisation des courriers entrants manuscrits à partir de l'extraction de mots clefs. L'autre type d'information pertinente présente dans les courriers manuscrits est l'information contenue dans les «champs numériques» : numéro de téléphone, code postal, numéro de fichier, code client, etc. (voir figure 2.2).

On peut définir les champs numériques comme une sous-catégorie plus contrainte des séquences numériques. Si toutes les séquences numériques possèdent une syntaxe particulière (nombre de chiffres, présence et position des séparateurs), certaines sont plus contraintes que d'autres. Par exemple, la syntaxe régissant un numéro de

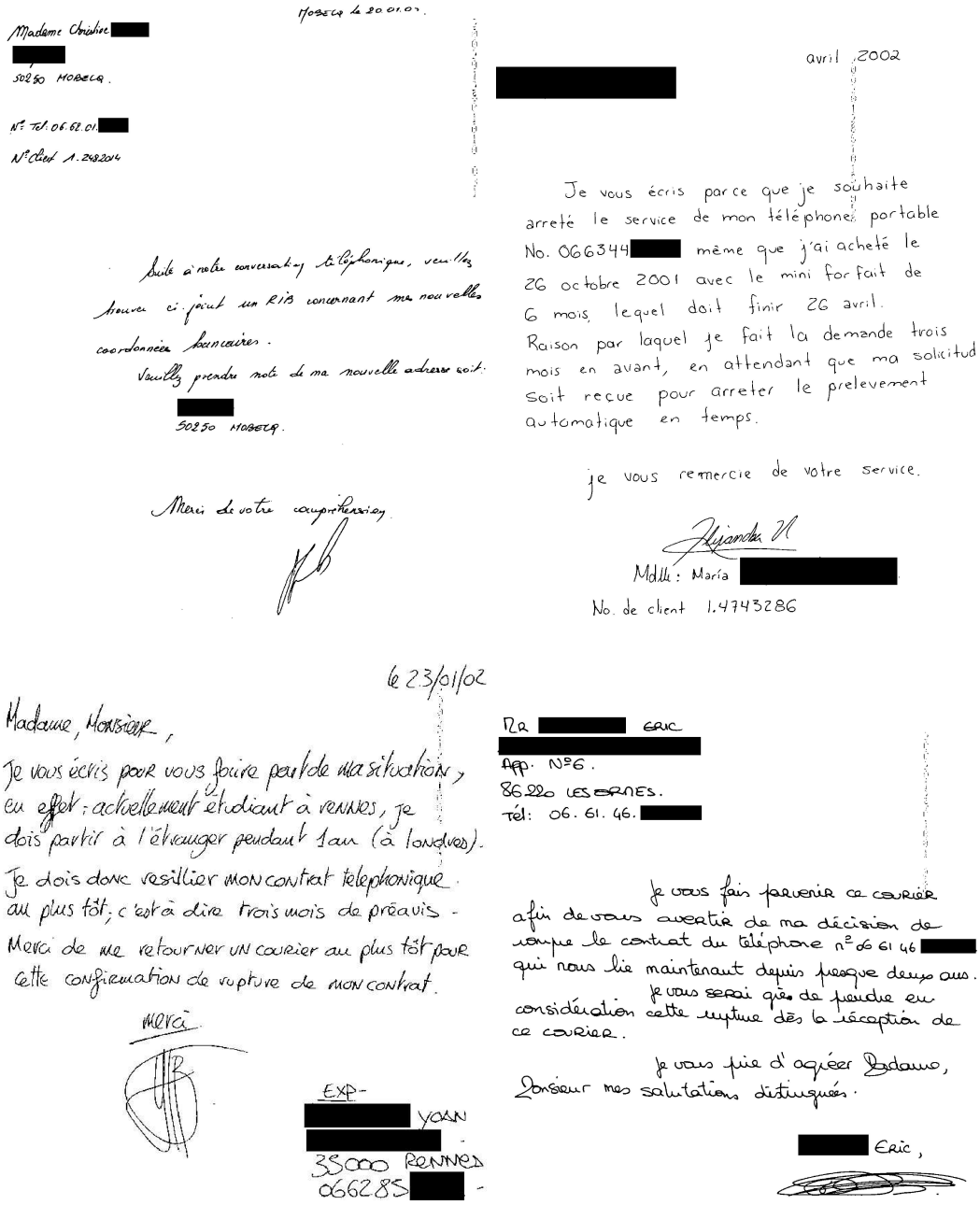


FIG. 2.1 – Exemples de courriers entrants manuscrits.

téléphone est beaucoup plus contrainte que celle d'un montant numérique. Dans le cas du numéro de téléphone, la séquence comporte 10 chiffres et des séparateurs (tiret, point) peuvent séparer chaque paire de chiffres. Dans le cas du montant

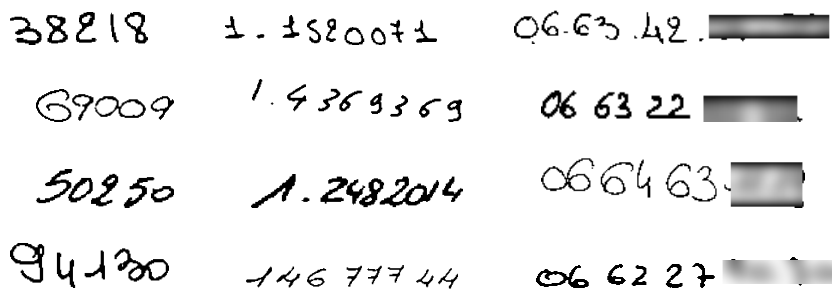


FIG. 2.2 – Exemples de champs numériques : codes postaux, numéros de client, numéros de téléphones.

numérique, le nombre de chiffres est quelconque et la position du séparateur (virgule, point) peut varier. Si les séquences numériques désignent n'importe quelle succession de chiffres (nombre, montant, etc.), on appelle un champ numérique les séquences numériques qui respectent une syntaxe particulière et suffisamment contraignante : le nombre de chiffres et la présence de séparateurs doivent être connus. Les montants (voir figure 2.3) ne sont donc pas considérés comme des champs numériques. On peut constater sur la figure 2.2 que les champs numériques respectent une syntaxe, même si elle n'est pas fixe : les numéros de téléphone contiennent toujours 10 chiffres, et la position des séparateurs est fixe même si leur présence n'est pas systématique.

FIG. 2.3 – Exemples de montants numériques.

Signalons également le cas particulier des dates qui, si elles repectent une syntaxe particulière et contraignante, peuvent être écrites soit dans une représentation strictement numérique, soit dans une représentation mixte numérique/textuelle avec le mois mentionné en toutes lettres (voir figure 2.4). Du fait de ces informations textuelles, les dates ne seront pas considérées comme champs numériques dans cette étude.

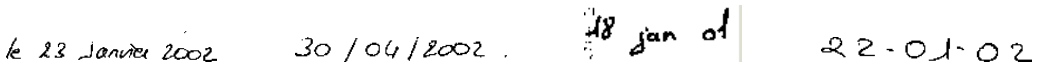


FIG. 2.4 – Les dates respectent une syntaxe particulière, mais contiennent souvent des informations textuelles (mois écrit en toutes lettres).

Ces champs numériques constituent une information pertinente dans la me-

sure où ils contiennent généralement des informations sur le client permettant de déterminer via une base de données clients son identification (numéro de téléphone ou code client), son type de contrat (code client) ou sa localisation géographique (code postal). La localisation et la reconnaissance de ces champs dans les courriers entrants constituent donc un réel besoin pour les entreprises recevant ce type de courrier.

2.2.3 Base de courriers annotés

Nous disposons pour nos expérimentations d'une base de courriers entrants contenant 293 courriers en apprentissage et autant en test. Les bases sont annotées au niveau champs, c'est-à-dire qu'on dispose de la position et de la valeur numérique de chaque champ, mais pas de l'étiquetage au niveau composante. Trois types de champs d'intérêt sont annotés : codes postaux, numéros de téléphone et codes client. Les effectifs des deux bases sont rapportés dans le tableau 2.1

Nombre de champs	codes postaux	téléphones	codes clients	total
Apprentissage	313	241	123	677
Test	328	250	150	718

TAB. 2.1 – Types de champs et effectifs dans les bases de courriers annotés.

2.3 Localisation de l'information manuscrite dans les systèmes de lecture de documents

Nous avons pu constater dans le chapitre précédent que la reconnaissance d'entités «hors contexte», déjà localisées offrait désormais des performances acceptables : caractères, mots et séquences numériques isolés peuvent être reconnus avec des taux de lecture intéressants. À partir des années 80, ces méthodes de reconnaissance d'entités manuscrites ont été intégrées dans des systèmes de lecture complets permettant le traitement automatique de documents. La reconnaissance des entités est alors précédée de l'étape délicate de localisation des informations, basée sur une exploitation du contexte de l'application. Les systèmes de lecture de document exploitent ainsi le contexte de l'application pour (i) localiser l'information d'intérêt grâce aux connaissances *a priori* sur la disposition plus ou moins fixe des éléments du document (ii) fiabiliser la reconnaissance de cette information en exploitant la présence d'éventuelles contraintes telles que la redondance de l'information, la présence d'un lexique, la connaissance du scripteur, etc. Une étape de post-traitement exploitant également le contexte de l'application est souvent mise en œuvre afin de vérifier les hypothèses de localisation et de reconnaissance des entités.

Nous nous focalisons dans cette partie sur la phase de localisation des informations dans les documents plus ou moins contraints. Les méthodes de localisation

de l'information reposent en grande partie sur l'exploitation des connaissances *a priori* disponibles concernant le document. À partir des connaissances a priori, il est possible de constituer un *modèle de document* plus ou moins figé définissant l'organisation des informations à l'intérieur de ce document : structure physique, nature et position des informations, présence de repères ou symboles connus à des emplacements précis, connaissances syntaxiques régissant tout ou partie de l'information recherchée. Toutes les méthodes de localisation reposent sur l'exploitation d'un modèle de document. On peut distinguer deux cas de figure suivant le niveau de contraintes qu'apportent les connaissances a priori.

- Lorsque l'on dispose de connaissances *a priori* en quantité suffisante, le modèle de document est suffisamment contraint pour réaliser une localisation des informations en se basant sur le modèle. C'est le cas des applications de lecture automatique de chèques bancaires, de formulaires ou d'adresses postales, où les différentes entités recherchées sont facilement localisées, généralement sans faire appel à la reconnaissance.
- Lorsque les connaissances *a priori* sont trop faibles, le modèle de document n'est pas suffisamment contraint pour effectuer une localisation directe des informations. C'est le cas des textes libres qui ne possèdent pas de structure physique stable. Dans ce cas, la localisation des informations pose de nouveaux problèmes : une segmentation des entités manuscrites est nécessaire afin d'identifier les mots du texte. On connait la difficulté d'une telle opération, et puisque le paradoxe de Sayre devient dans ce cas incontournable, la phase de reconnaissance doit être liée à la phase de localisation pour fournir des résultats fiables.

Nous présentons maintenant les systèmes de localisation de l'information manuscrite proposés dans la littérature dans ces deux cas de figure.

Dans le cas où l'on dispose d'un modèle de document suffisamment contraint, il est possible d'effectuer une localisation fiable de l'information sans reconnaissance en se basant sur la structure physique connue du document. Il s'agit pricipalement, par ordre de contraintes, des documents suivants : formulaires, chèques bancaires, adresses postales ou textes contraints.

2.3.1 Localisation de champs d'intérêt dans les formulaires

Les formulaires contenant des informations manuscrites possèdent généralement une structure totalement statique, autorisant une localisation immédiate des zones d'intérêt. À partir de ces zones d'intérêt, des délimiteurs matérialisant la zone dans laquelle le scripteur doit écrire permettent d'identifier facilement les composantes appartenant au champ recherché : cases prédéfinies dans le cas de précasé, zone identifiant la région contenant l'information, ligne de base sur laquelle le scripteur doit remplir le champ. L'application d'un simple calque peut ainsi parfois suffire à extraire les informations recherchées.

Dans [Madhvanath 95], des formulaires de recensement sont traités, où chaque champ à remplir est délimité par un rectangle qui reçoit un mot unique de la part

du scripteur (voir figure 2.5 gauche) ; un simple calque permet de localiser les mots. Dans [Milewski 06a], des formulaires de pré-hospitalisation sont traités, à partir desquels cinq champs particuliers sont extraits. Comme les régions sont statiques, la localisation ne pose aucun problème (voir figure 2.5 droite).

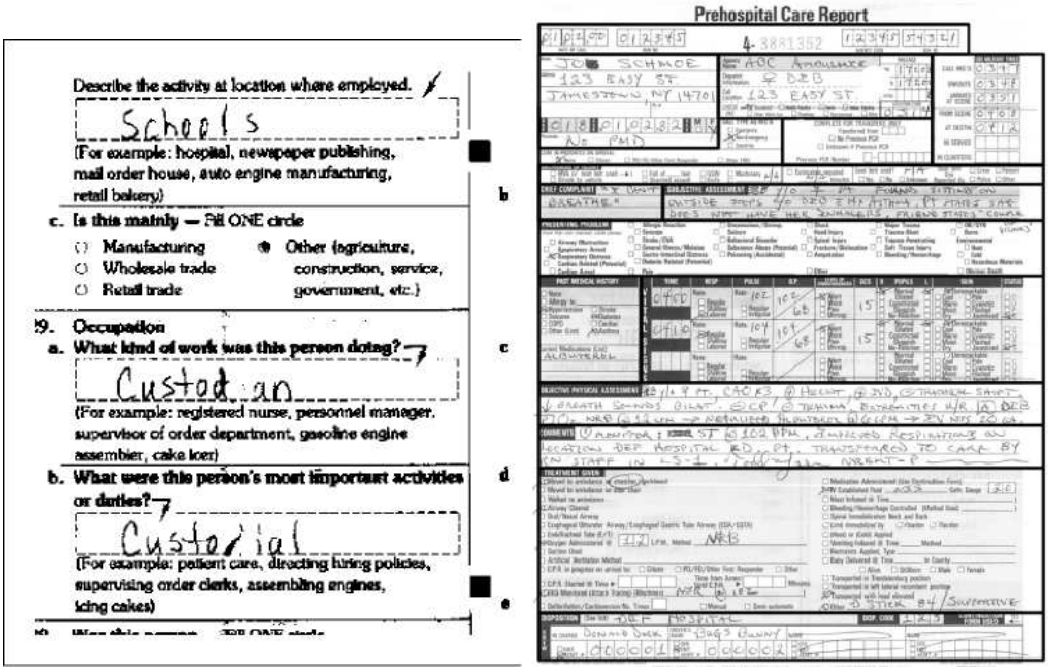


FIG. 2.5 – Exemples de formulaires : formulaire de recensement traité dans [Madhvanath 95] et formulaire de pré-hospitalisation[Milewski 06a].

Dans tous ces cas de figure, leur structure étant parfaitement connue, le modèle de document lié aux formulaires est tellement contraint qu'il autorise une localisation immédiate et sans problèmes des informations [Bayer 97, Niyogi 97], ne nécessitant pas d'analyse de la structure du document ni d'étape de reconnaissance.

2.3.2 Localisation de montants sur les chèques bancaires

Depuis les années 90, les systèmes de lecture automatique de chèques permettent de lire quotidiennement plusieurs millions de chèques. Dans ces applications, on cherche principalement à localiser et à reconnaître le montant littéral et/ou numérique [Ye 99, Djeziri 97, Kim 97a, Heutte 97], mais aussi parfois les dates [Morita 02, Xu 03], et les signatures [Madasu 03].

L'aspect des chèques comporte des variations suivant les banques : les logos sont différents, la texture du fond varie, etc. Malgré ces variations, la structure des chèques d'un même pays reste stable (voir figure 2.6). En France par exemple, la position des guides pour l'écriture des champs par l'utilisateur est fixée par une norme qui

donne les dimensions de chacun des éléments (norme AFNOR NFK 11-111). Si cette norme n'est pas toujours parfaitement respectée dans la pratique, la position des informations est approximativement connue et certains repères permettent de retrouver facilement les champs d'intérêt. Le montant littéral est toujours précédé de la mention «payez contre ce chèque» [1] et guidé par une ligne de base. Concernant le montant numérique, il est entouré de deux symboles euro : un petit à gauche et un grand à droite, dont la position, la taille et la forme sont précisées dans la norme. Ces symboles peuvent donc être facilement retrouvés par template matching. Le montant numérique est également souvent délimité par un rectangle.

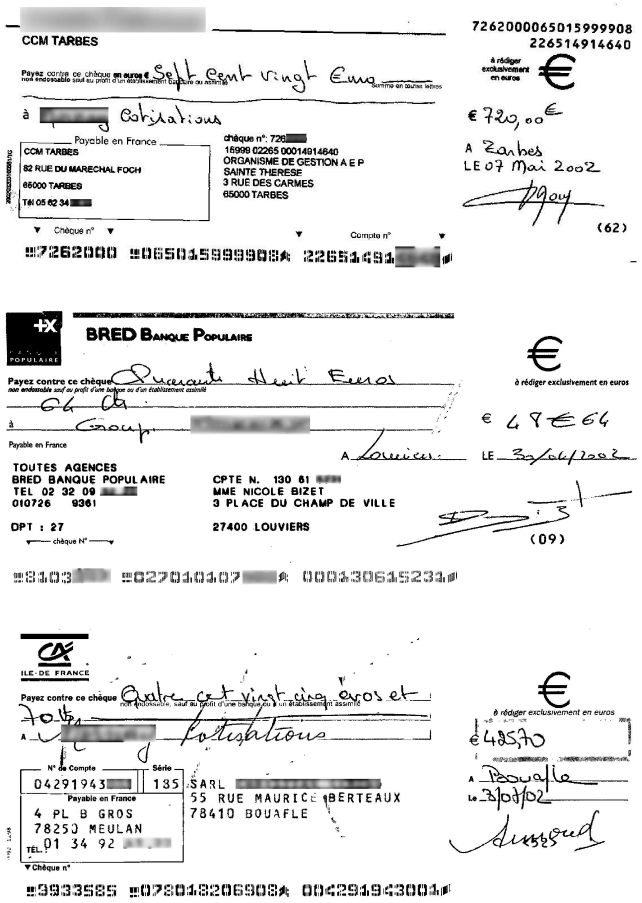


FIG. 2.6 – Exemples de chèques bancaires français binarisés. Si les styles de logo et les fonds varient suivant les banques, la position des champs d'intérêt est approximativement stable.

Dans la littérature, on peut distinguer deux types d'approches permettant de localiser le montant littéral et le montant numérique : les approches qui reposent

[1] Avant le passage à l'euro au début de l'année 2002, cette mention était «B.P.F.».

entièrement sur la connaissance de la position des guides pour les montants ou des symboles particuliers, et les approches plus génériques procédant à une véritable analyse de la structure du chèque.

Dans [Kim 97a], les auteurs présentent un système de lecture automatique de chèques effectuant la lecture des deux montants. Les deux montants sont localisés par une analyse de l'image du chèque : présence de deux longues lignes horizontales proches du montant littéral, signe $ et rectangle fermé autour du montant numérique. Dans [Heutte 97], un système de localisation et de lecture du montant numérique des chèques français est présenté. La localisation automatique du montant numérique est effectuée grâce à la détection de la mention «BPF» juste avant le montant. Dans [Lee 97], la localisation des informations sur des chèques bancaires brésiliens est effectuée par un simple calque grâce au patron très stable de ces chèques.

Dans [Ye 99], les auteurs présentent une méthode de localisation des informations manuscrites sur les chèques bancaires. La méthode est basée sur une détection et une élimination des lignes de base par morphologie mathématique. La connaissance *a priori* sur la position des informations est ensuite intégrée dans une combinaison d'analyses ascendante et descendante du chèque. Les informations manuscrites sont extraites par un seuillage adaptatif local. Dans [Djeziri 97] l'extraction des éléments d'un chèque se fait par une élimination des lignes de base en extrayant un modèle de chèque vierge au chèque à traiter. Dans un second temps, un étiquetage des composantes connexes du chèque est réalisé à partir de leurs boîtes englobantes.

On peut constater que d'une manière générale, la localisation des entités numériques sur un chèque ne pose pas de problème. Cette opération est dans tout les cas réalisée indépendamment de la reconnaissance des montants.

2.3.3 Localisation d'entités dans les adresses postales

Dans les nombreuses applications de lecture des adresses postales développées récemment, la localisation des informations a lieu à deux niveaux. Dans un premier temps, une localisation du bloc adresse est effectuée [Jain 92, Tulyakov 03, Pfister 00, Lii 93, Wang 88]. Dans un second temps, une interprétation du bloc adresse est réalisée afin de localiser le code postal [Cohen 91, Jarousse 98, de Waard 94], le bureau distributeur [Park 02] ou un nom de rue [Kim 98].

La localisation du bloc adresse peut paraître aisée sur des enveloppes «propres», mais il existe de nombreuses enveloppes contenant des images ou des messages publicitaires en plus du timbre et du tampon postal. Les techniques employées ne font généralement pas intervenir les connaissances *a priori* sur la position du bloc adresse ; elles reposent plutôt sur des approches géométriques analysant la taille et la disposition des boîtes englobantes des composantes connexes [Yeh 87, Wang 88], ou sur des approches à base de détection de texture qui distinguent les zones «écriture» des zones «fond» [Jain 92].

Contrairement à la détection du bloc adresse dans l'enveloppe, les méthodes mises en œuvre pour la localisation des champs d'intérêt dans le bloc adresse reposent sur l'exploitation d'un certain nombre de contraintes régissant la structure des adresses

postales sur les enveloppes : le bloc adresse est disposé en lignes dont le nombre peut varier, et dans lequelles on retrouve toujours les champs prénom, nom, numéro et nom de rue, code postal et nom de ville (voir figure 2.7).

FIG. 2.7 – Image d'enveloppe en niveau de gris traitée dans [Jain 92] (en haut) et blocs adresse provenant d'enveloppes françaises (à gauche [El-Yacoubi 02]) et américaine (à droite [Kim 97b]).

Dans [Pfister 00], les auteurs décrivent les algorithmes de traitement pour la localisation et la reconnaissance de codes postaux sur des enveloppes allemandes. Pour cela le bloc adresse est dans un premier temps segmenté en lignes grâce à une approche itérative. La segmentation des lignes en mots est réalisée en se basant sur l'hypothèse que la ligne contenant le code postal ne contient que deux entités : le code postal à gauche et le nom de la ville à droite. La localisation du code postal se fait donc en séparant les lignes en deux sur un critère de distance. Les hypothèses de segmentation générées sont évaluées par un ratio hauteur/largeur, de manière à

fournir à gauche une hypothèse de localisation réaliste pour un code postal.

Dans [Jarousse 98], une méthode de localisation du code postal dans les blocs adresse est basée sur trois modules principaux. Le premier recompose, après une phase de pré-étiquetage des composantes connexes, les caractères mal formés. Le second module réalise inversement la séparation des mots en graphèmes et établit une description syntaxique des séquences rencontrées. Enfin, une phase de décision inspecte l'ensemble des données obtenues pour extraire le code postal au sein du bloc adresse.

Dans [Kim 98], les auteurs présentent un système d'interprétation d'adresses postales qui vise à localiser et reconnaître les mots clefs. Après une étape de segmentation en lignes de texte, les lignes sont segmentées en mots selon une approche ne faisant pas appel à la reconnaissance. Les espaces inter-mots sont déterminés à l'aide d'un réseau de neurones alimenté par des caractéristiques extraites des boîtes englobantes des composantes. Un moteur de reconnaissance de mot isolé procédant par programmation dynamique est ensuite appliqué sur le résultat de la segmentation en mots. Les mots sont identifiés en mettant en concurrence les entrées du lexique. Les phases de localisation (segmentation en mots) et de reconnaissance sont donc indépendantes.

Le système HWAI (HandWritten Address Identification), détaillé dans de nombreux articles [Srihari 97a, Cohen 94, Cohen 91], réalise une chaîne de traitement complète pour l'interprétation des adresses postales américaines, les champs recherchés étant le bureau distributeur et le code postal. Le système prend en entrée des images en niveaux de gris de blocs adresse. Après une étape de segmentation du bloc en lignes, plusieurs hypothèses de segmentation des lignes en mots sont générées en se basant sur les espaces entre les composantes connexes. À l'aide de caractéristiques spatiales, une étape de classification grossière est appliquée sur les hypothèses de mots pour les étiqueter en tant que lettre ou regroupement de lettres, nom de l'État, chiffre, code postal, boîte postale, bruit. Une analyse syntaxique à deux dimensions est alors effectuée en mettant en correspondance les hypothèses de segmentation/classification avec des règles syntaxiques contenant les connaissances *a priori* sur la structure d'un bloc adresse. L'une des syntaxes possibles est par exemple : *1ère ligne : boîte postale + nombre ; 2ème ligne : numéro de rue + nom de rue ; 3ème ligne : code postal + nom de ville + nom de l'Etat.* L'ensemble d'étiquettes donnant le meilleur score de "matching" est conservé. Dans un second temps, une reconnaissance des entités ainsi étiquetées est effectuée.

Comme le montre cette étude des systèmes de localisation d'entités manuscrites dans les blocs adresse, les connaissances *a priori* varient suivant les pays, mais constituent d'une manière générale un modèle de document suffisamment contraint pour réaliser une localisation des codes postaux et des mots clefs sans reconnaissance. Il existe toutefois des travaux réalisant conjointement la localisation et la reconnaissance des entités dans les adresses postales [El-Yacoubi 02]. Nous aborderons ces travaux dans la section 2.3.5.

2.3.4 Localisation/reconnaissance de mots dans des textes libres

Il y a quelques années sont apparus les premiers travaux concernant la lecture de textes manuscrits dits «libres». Lorsque des textes libres pleine page sont traités, le modèle physique de document est peu contraint : la structure, le contenu et l'objet du document sont inconnus. La seule contrainte généralement connue est une orientation privilégiée des lignes de texte. On ne peut donc plus exploiter les connaissances *a priori* sur la disposition physique des entités pour localiser l'information. Contrairement aux applications de lecture automatique de chèques ou d'adresses postales où l'industrialisation a motivé les recherches, les besoins applicatifs vis-à-vis des textes libres ne sont pas encore parfaitement identifiés. Il est donc difficile de savoir ce que l'on cherche à localiser et à reconnaître. Actuellement, les travaux portent donc essentiellement sur la reconnaissance intégrale de textes dont le lexique plus ou moins grand est supposé connu. Dans tous ces travaux, on procède à une segmentation du document en lignes sans reconnaissance. Deux stratégies peuvent ensuite être utilisées pour réaliser la segmentation des lignes en mots :

- La première stratégie consiste à effectuer la localisation sans reconnaissance. La segmentation en mots est généralement effectuée par une analyse des espaces entre composantes afin de distinguer les espaces inter-mots des espaces inter-lettres [Marti 01a, Nosary 02, Srihari 93, Kim 01]. Des méthodes de reconnaissance de mots isolés (voir chapitre précédent) sont ensuite appliquées sur les hypothèses de segmentation. On peut comparer cette statégie avec les approches «segmentation-based» pour la reconnaissance de séquences numériques dans le sens où le séquencement des traitements empêche toute remise en cause des hypothèses de segmentation à l'issue de l'étape de reconnaissance. Une erreur lors de l'étape de segmentation ne peut donc être rattrapée.
- La deuxième stratégie que l'on peut qualifier de «localisation/reconnaissance» consiste à réaliser conjointement la segmentation et la reconnaissance sur l'ensemble de la ligne de texte [Marti 01b, Vinciarelli 04]. On peut ainsi voir cette stratégie comme une extension à la ligne de texte des méthodes de segmentation implicite ou de segmentation-reconnaissance mises en œuvre à l'échelle du mot ou des séquences numériques. Plutôt que de considérer des décisions locales de segmentation ne prenant pas en compte le contexte, ce type d'approche propose des solutions de segmentation/reconnaissance sur l'ensemble de la ligne de texte. L'inconvénient de cette stratégie réside toutefois dans l'explosion combinatoire engendrée par la multiplication des hypothèses de segmentation/reconnaissance sur une ligne de texte.

Nous donnons maintenant des exemples de ces deux stratégies.

Localisation des mots sans reconnaissance

Dans [Marti 01a, Nosary 02], une reconnaissance de textes libres où certaines contraintes sont imposées aux scripteurs est présentée. Du fait de ces contraintes, il n'existe pas d'applications industrielles mettant en œuvre de tels documents, et les travaux sont essentiellement académiques. Il s'agit en particulier de contraintes

d'espacement entre les lignes et d'espacement entre les mots, afin de faciliter la segmentation en lignes et la segmentation en mots (voir figure 2.8). Nous qualifions par la suite ces documents de textes faiblement contraints.

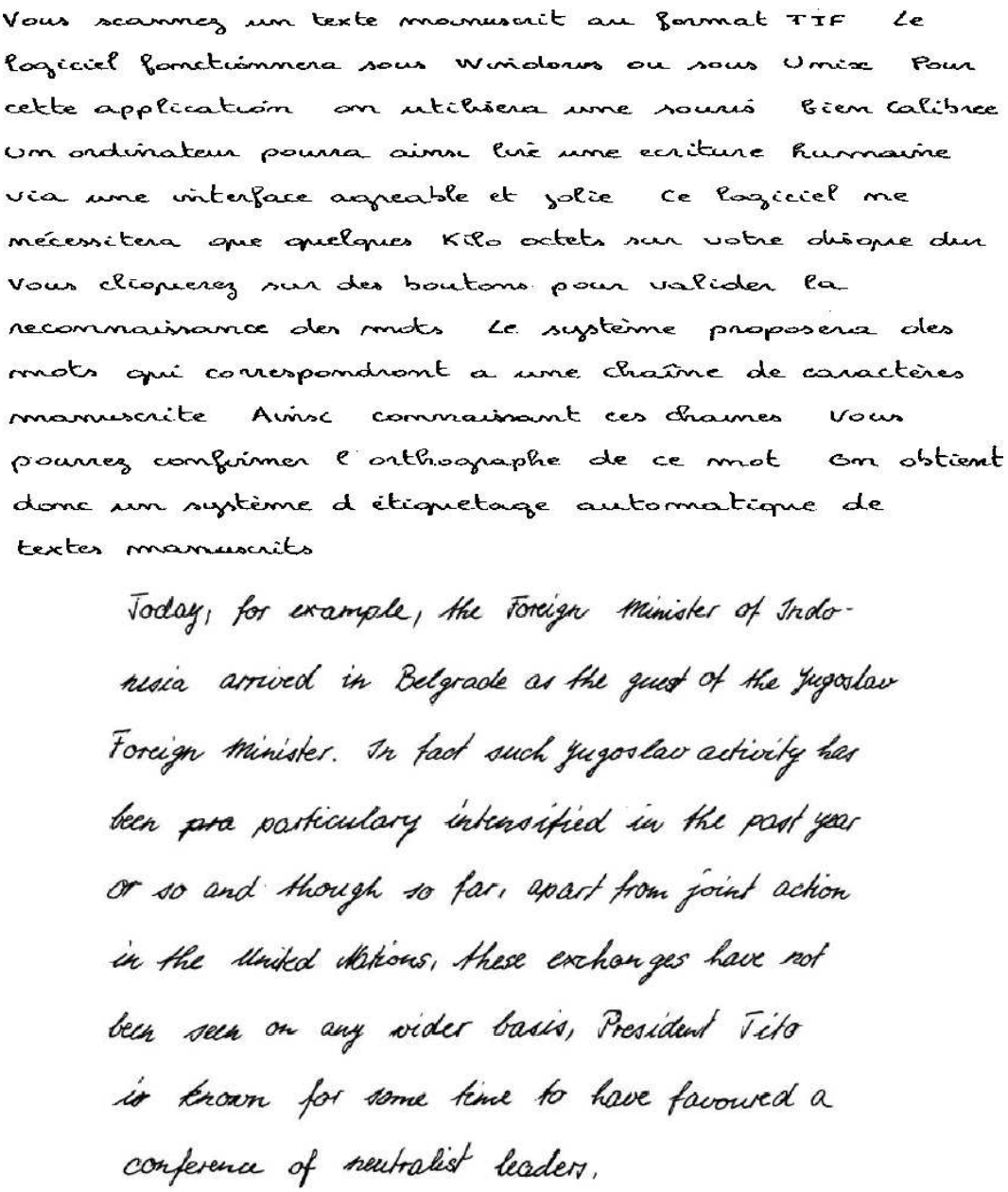


FIG. 2.8 – Exemples de textes manuscrits français et anglais traités dans [Nosary 02] et [Marti 01a]. Certaines contraintes d'espacement inter-lignes et inter-mots ont été imposées aux scripteurs.

Dans ce cas, on ne dispose pas d'un modèle physique de document, mais des connaissances *a priori* sur les espacements entre les lignes et entre les mots permettent d'effectuer une localisation des mots sans faire appel à la reconnaissance. Dans [Marti 01a] et [Nosary 02], les documents exploités ont été produits en imposant deux contraintes aux scripteurs : premièrement, les lignes de texte doivent être suffisamment espacées de telle sorte qu'elles sont parfaitement séparables par une simple recherche de lignes horizontales de pixels blancs. Deuxièmement, on impose aux scripteurs de suffisamment séparer les mots pour que les espaces entre deux mots (espaces inter-mots) soient toujours plus grands que des espaces séparant deux lettres d'un même mot (espaces intra-mots). Sous cette contrainte, la tâche de segmentation d'une ligne de texte en mots consiste à estimer un seuil maximum au-delà duquel les espaces entre deux composantes seront considérés comme espaces inter-mots.

L'approche proposée dans [Srihari 93] segmente les lignes de texte en mots. Chaque mot est ensuite soumis à un moteur de reconnaissance dont on conserve les N meilleures propositions. Le treillis de reconnaissance de la ligne est alors exploré en considérant des contraintes linguistiques d'ordre grammatical (nom, verbe, adjectif, ...). Le principal inconvénient de cette approche est d'enchaîner les traitements séquentiellement. Ainsi, une erreur de segmentation commise en amont est fatale pour la reconnaissance.

Nous pouvons constater que les travaux mettant en œuvre une localisation des mots sans reconnaissance dans des documents dont la structure physique est inconnue sont limités. Dans [Marti 01a] et [Nosary 02], les contraintes imposées aux scripteurs ne sont que rarement respectées dans le cas de documents réels non destinés à être lus par un système de lecture automatique de documents [Seni 94]. Dans [Srihari 93], une segmentation des lignes de texte en mots sans reconnaissance est également effectuée, et l'exploitation de connaissances grammaticales ne peut corriger toutes les erreurs faites lors de la phase de segmentation.

Localisation/reconnaissance de mots

Pour remédier au problème difficile de segmentation des lignes en mots dans le contexte de textes libres, les approches proposées dans [Marti 01b] et [Vinciarelli 04] ne réalisent pas de segmentation préalable de la ligne en mots. Dans les deux cas, les méthodes développées sont testées sur la base IAM [Marti 99] comportant des textes relativement propres (voir figure 2.8, texte du bas). Après une étape de segmentation du document en lignes, les lignes de texte sont considérées dans leur intégralité, et une décision globale de localisation/reconnaissance sur l'ensemble de la ligne est effectuée. La localisation/reconnaissance est réalisée par une approche à segmentation implicite. Des modèles de lignes de textes sont réalisés grâce à des HMM de lettres, concaténés pour former des modèles de mots, eux-mêmes concaténés pour former un modèle de ligne (voir figure 2.9). Lors du décodage, la segmentation et la reconnaissance des mots sur l'ensemble de la ligne de texte sont alors réalisées simultanément par l'algorithme de Viterbi. Ces modèles de ligne considèrent donc qu'une ligne de

texte est composée uniquement de mots connus, ce qui impose de travailler avec de grands lexiques : jusqu'à 50 000 mots dans [Vinciarelli 04], et plusieurs milliers dans [Marti 01b].

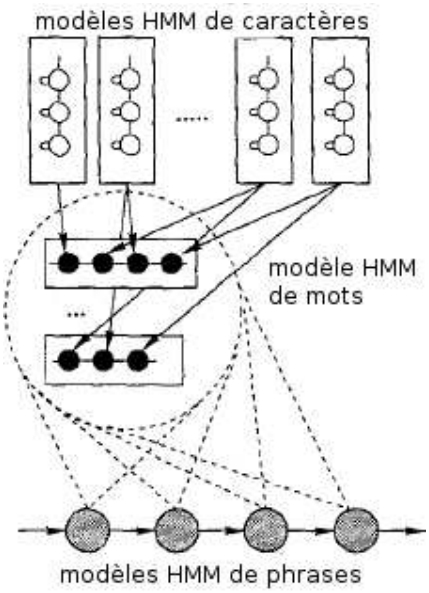


FIG. 2.9 – Modèle de ligne de texte utilisés dans [Marti 01b]

Afin de compenser la baisse de performances en reconnaissance induite par ces grands lexiques, les auteurs introduisent des connaissances linguistiques sous la forme de modèles statistiques de langage (N-gramme de mots [Rosenfeld 00]). Ces stratégies de localisation/reconaissance semblent ainsi une solution intéressante pour la segmentation en mots dans le cadre de la reconnaissance de textes libres. Rappelons toutefois que ces deux travaux sont appliqués sur des textes ayant été écrits dans l'optique d'une lecture automatique avec les contraintes d'espacement décrites plus haut.

2.3.5 Documents non contraints : vers des systèmes d'extraction d'information

Nous avons pu constater que les stratégies employées pour la localisation d'information manuscrite dans les documents contraints et faiblement contraints dépendaient fortement du contexte de l'application. Lorsque le modèle de document est suffisamment contraint (chèques, formulaires, adresses postales), la localisation est entièrement basée sur des connaissances *a priori* sur la structure du document. Lorsque cette structure est inconnue (cas des textes faiblement contraints), on cherche à localiser tous les mots du texte, soit par des approches utilisant des connaissances *a priori* sur les espacements inter-mots et inter-lettres, soit en faisant

intervenir la reconnaissance. Ces dernières méthodes basées sur une localisation et une reconnaissance conjointes des entités manuscrites sont toutefois possibles dans les travaux présentés précédemment car les textes traités sont relativement propres et ne contiennent que des mots appartenant à un lexique connu.

Le problème de lecture intégrale de texte devient plus délicat dans le cas de documents non contraints «réels» tels que les courrier entrants (voir figure 2.10), pour plusieurs raisons. Premièrement, le lexique des documents ne contient pas uniquement des mots d'un lexique connu, mais tous les mots d'une langue, ainsi que des séquences numériques, des noms propres, ratures, signature, symboles divers etc. Imaginons la mise en place d'une stratégie de localisation/reconnaissance semblable à celles proposées dans [Marti 01b] et [Vinciarelli 04]. Les modèles de lignes doivent pouvoir intégrer toutes ces informations n'appartenant pas au lexique sous peine de ne pouvoir réaliser un alignement correct des modèles. Le processus de reconnaissance doit également être capable de reconnaître des classes autres que lettres : chiffres, symbole, bruit, etc. Deuxièmement, la structure en lignes des documents réels est parfois hasardeuse, et l'on rencontre fréquemment des lignes dont les composantes sont liées avec une autre ligne, ou se chevauchant. Cette remarque est également valable pour les mots qui peuvent se chevaucher ou comporter des espaces inter-lettres plus importants que certains espaces inter-mots. Enfin les images de documents réels numérisés peuvent contenir des défauts de numérisation engendrant du bruit.

En l'état actuel des recherches, la lecture intégrale de documents réels sans connaissance *a priori* semble donc extrêmement délicate, et la difficulté d'une telle tâche ne peut pas conduire à des résultats fiables. Dans le cas des courriers entrants, on peut d'ailleurs s'interroger sur l'intérêt d'une lecture intégrale du document, puisque seules certaines informations nous intéressent : identité et coordonnées de l'expéditeur, objet du courrier, etc.

A partir de ce double constat d'impuissance et d'intérêt limité, une solution alternative à la lecture intégrale des documents est la lecture partielle visant à extraire l'information d'intérêt : nom de l'expéditeur, numéro de client, objet du courrier, etc. Contrairement à la localisation de montants dans les chèques ou de champs dans les formulaires, on souhaite localiser des champs manuscrits particuliers dans un environnement de texte manuscrit, sans pour autant localiser toutes les entités d'un texte comme dans la lecture intégrale de document. On se situe donc dans une problématique d'*extraction d'information dans des documents manuscrits.*

Peu de travaux ont abordé cette problématique. Selon nous, seuls les travaux de Koch [Koch 06], réalisés en parallèle des travaux présentés dans cette thèse, traitent de l'extraction d'information dans des documents manuscrits non contraints réels. Les travaux présentés concernent l'extraction de mots clefs appartenant à un lexique dans les courriers entrants présentés dans la section 2.2.1 en vue d'effectuer une catégorisation des documents. Bien que n'étant pas appliquée sur des textes libres, l'approche développée dans [El-Yacoubi 02] visant à localiser et reconnaître simultanément des noms de rue dans des lignes d'adresses postales mérite également d'être mentionnée car elle permet d'effectuer une réelle opération d'extraction d'informa-

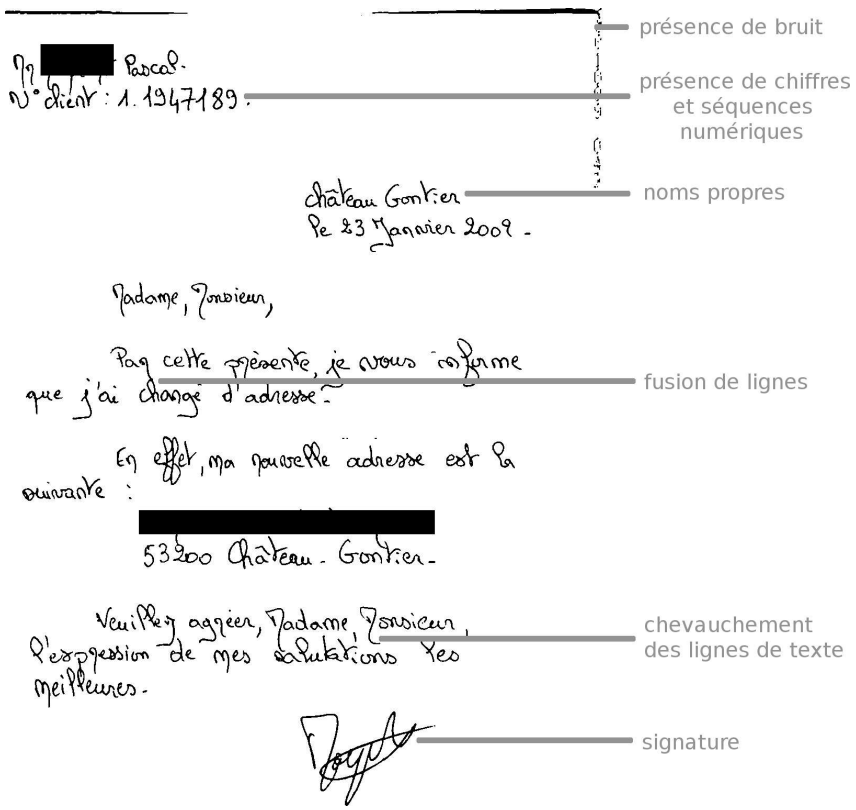


FIG. 2.10 – Courrier entrant et difficultés rencontrées par rapport à un texte libre propre dont le lexique est connu.

tion. Nous décrivons maintenant ces deux approches.

Extraction de mots clefs dans des courriers entrants

Dans [Koch 06], un système de catégorisation de courriers entrants basé sur l'extraction de mots clefs appartenant à un lexique (jusqu'à 1000 mots) est présenté. Il s'agit de déterminer l'objet d'un courrier en détectant la présence d'un certain nombre de mots dans le texte.

La stratégie utilisée pour l'extraction des mots clefs repose sur une analyse globale des lignes de texte. Un modèle de ligne comprenant à la fois les mots appartenant au lexique, les mots hors lexique et les espaces est proposé (voir figure 2.11). Une stratégie de segmentation/reconnaissance appliquée sur l'ensemble de la ligne produit un treillis d'observations qui, aligné sur le modèle de ligne, propose des hypothèses de localisation et de reconnaissance des mots appartenant au lexique.

La modélisation des éléments hors lexique est effectuée par un modèle ergodique permettant toutes les transitions possibles entre lettres. Ce modèle ergodique est mis en concurrence dans le modèle de ligne avec les modèles de mots

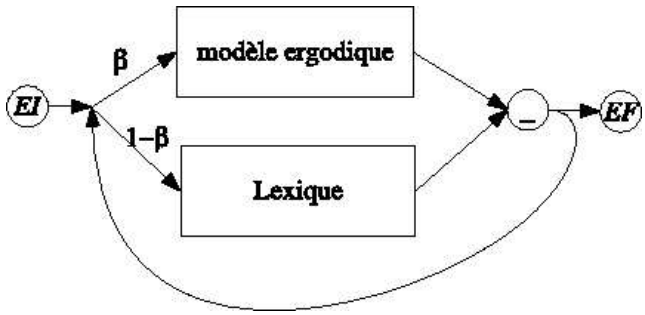


FIG. 2.11 – Modèle de ligne utilisé par Koch [Koch 06]. *EI* et *EF* désignent l'état initial et l'état final du modèle, et « _ » désigne les espaces inter-mots

du lexique. La stratégie de segmentation/reconnaissance met en œuvre une étape de sur-segmentation des composantes par une analyse des contours et un classifieur neuronal. Une méthode de réduction de lexique est mise en œuvre afin de limiter la combinatoire des hypothèses de segmentation/reconnaissance.

L'extraction des mots clefs repose donc sur une modélisation partielle des lignes de texte permettant à la fois de localiser et reconnaître les mots appartenant à un lexique, et d'absorber les mots hors lexique sans toutefois les reconnaître.

Extraction du nom de rue dans des adresses postales

Dans [El-Yacoubi 02], une approche similaire basée sur la modélisation d'une ligne de texte est appliquée à l'extraction de nom de rue dans des adresses postales. La modélisation des lignes est réalisée par des modèles de Markov cachés. Le modèle de ligne est constitué du modèle de nom de rue recherché, auquel on concaténe un modèle générique à gauche et un modèle générique à droite permettant d'absorber les informations non pertinentes (numéro de rue, nature de la voie, etc.). Signalons que la taille du lexique atteint plusieurs milliers de mots, mais que le système inclut des méthodes efficaces de réduction de lexique. Comme dans [Koch 06], l'absorption des éléments hors lexique est réalisée par un modèle ergodique permettant toutes les transitions possibles entre lettres.

Les travaux présentés dans [Koch 06] et [El-Yacoubi 02] réalisent ainsi une extraction d'information dans les documents manuscrits. Les stratégies reposent sur une modélisation permettant une reconnaissance partielle des lignes de texte par l'intermédiaire de modèles ergodiques qui modélisent l'information hors lexique.

L'extraction d'information est toutefois un vaste domaine qui, s'il a peu été étudié sur des documents manuscrits, a connu de nombreux travaux sur les documents électroniques. Nous présentons maintenant ce domaine de recherche et ses applications.

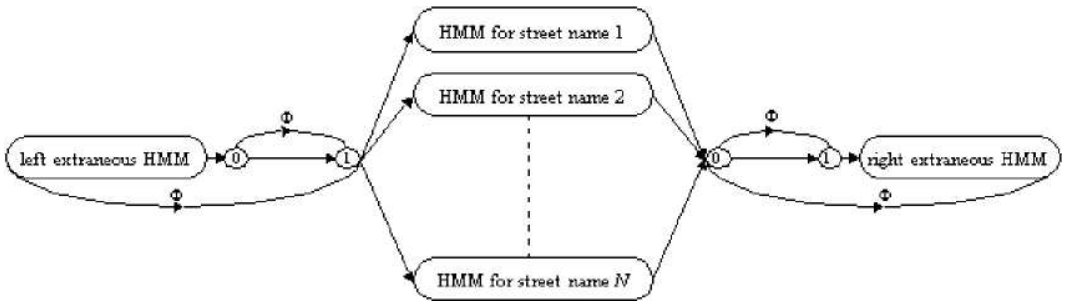


FIG. 2.12 – Modèle de ligne utilisé par El Yacoubi [El-Yacoubi 02], obtenu par concaténation des modèles de nom de rue recherchés et des modèles ergodiques à gauche et à droite permettant d'absorber les éléments hors lexique.

2.4 Extraction d'information dans les documents textuels

2.4.1 Définition

L'extraction d'information est un terme assez général et donc difficile à définir du fait que l'on peut extraire de l'information à partir de sources très différentes : une scène vidéo, une séquence de nucléotides, un livre, une séquence sonore, etc. Dans la langue française, selon le dictionnaire «Le petit Robert», le verbe «extraire» signifie *tirer*, *dégager*, *isoler* ou *relever* une information de quelqu'un ou quelque chose. Lorsque l'on considère l'extraction d'information dans des documents, il s'agit donc d'une opération de *sélection de l'information pertinente*.

L'extraction d'information dans les documents a connu un grand intérêt ces dernières années avec l'explosion du nombre de documents disponibles sur internet. On a coutume de distinguer les méthodes d'extraction suivant le type de documents traités : documents structurés, semi-structurés ou textes en langage naturel. Les documents structurés ont une structure extrêmement rigide et stable pour une même catégorie de documents. Il s'agit par exemple de pages internet structurées par des balises HTML qui constituent des repères privilégiés pour une extraction d'information immédiate. Les documents texte en langue naturelle sont au contraire considérés comme non structurés. L'extraction d'information dans ce type de document nécessite généralement des traitements plus importants. Enfin on considère souvent une catégorie intermédiaire de documents dits «semi-structurés». Ils possèdent une structure, mais cette structure est plus variable que dans le cas de documents structurés. Ils peuvent contenir certains éléments en langue naturelle.

La tâche d'extraction d'information dans les documents en langue naturelle nous concerne plus particulièrement dans la mesure où ces documents ne possèdent pas de structure particulière et sont par conséquent à rapprocher des documents manuscrits non contraints. On se focalise donc dans cette section sur les méthodes

d'extraction d'information dans les textes en langage naturel. On trouve dans la littérature plusieurs définitions pour l'extraction d'information dans les textes en langue naturelle :

- Selon Appelt [Appelt 99], un système d'extraction d'information consiste à analyser du texte libre dans le but d'extraire différents types d'informations spécifiques.
- Pour Pillet [Pillet 00], l'extraction d'information consiste à identifier de l'information bien précise d'un texte en langue naturelle mais aussi à pouvoir la représenter sous forme structurée.
- Selon Califf [Califf 03], l'extraction d'information est une forme d'analyse superficielle de texte qui localise un ensemble spécifié de champs pertinents dans un document en langue naturelle.
- Dans [Cowie 96], il s'agit d'extraire des informations factuelles précises d'un ensemble de documents homogènes pour remplir automatiquement un formulaire défini à l'avance.
- Selon Poibeau [Poibeau 02], l'extraction d'information désigne l'activité qui consiste à extraire automatiquement de l'information structurée à partir d'un texte en langage naturel non structuré.

On constate que les définitions ne sont pas toutes identiques, même si certaines notions sont communes à la plupart d'entre elles. On peut ainsi dégager assez précisément la notion de «champs pertinents», également désignée par les termes d'«information spécifique» ou d'«information bien précise» dans les définitions ci-dessus. L'information recherchée est ainsi «pertinente» ou «spécifique» pour un problème d'extraction considéré, et signifie implicitement que le reste du message n'est pas pertinent (ou moins pertinent) pour le problème donné. La deuxième notion véhiculée dans presque toutes ces définitions est la notion d'«information structurée», qui est sous-entendue dans les termes d'«*ensemble* spécifié de champs pertinents» ou de «formulaires». Plus floue, cette notion signifie qu'il existe des relations plus ou moins fortes entre les différents champs pertinents extraits. La dernière notion à retenir selon nous dans ces définitions concerne l'«analyse superficielle» des textes. Si cette notion n'est pas nécessaire à la définition de l'extraction d'information, elle fait allusion aux techniques mises en œuvre pour l'extraction d'information dans les textes. Le qualificatif de «superficielle» sous-entend que l'on ne cherche pas à modéliser finement l'ensemble du texte, mais plutôt à réaliser une modélisation faisant cohabiter une description précise de l'information pertinente, et une description plus floue de l'information non pertinente. Nous verrons par la suite que cette modélisation à la fois en profondeur et surfacique constitue la clef des systèmes d'extraction d'information.

Une compilation de toutes ces définitions pourrait donc être la suivante : *l'extraction d'information consiste en l'analyse superficielle de texte en langue naturelle en vue d'une identification de champs pertinents entre lesquels il existe une relation.* Afin d'illustrer les différentes notions que nous venons de dégager, nous présentons

deux exemples d'extraction d'information.

Exemples

Un exemple classique d'extraction d'information provenant de la sixième campagne d'évaluation américaine *Message Understanding Conferences* (MUC), consiste à remplir un certain nombre de champs à partir du texte suivant :

> *San Salvador, 19 avril 1989 (ACAN-EFE) – Le président du San Salvador Alfredo Cristani a condamné l'assassinat d'origine terroriste de l'Attorney General Roberto Garcia Alvarado et a accusé Le Front de Libération National Farabundo Marti (FMLN) du meurtre. (...)*

Dans le cadre de l'extraction d'information relative aux attentats en Amérique du sud, les participants doivent remplir le formulaire suivant :

Date de l'incident : "19 avril 1989"
Lieu de l'incident : "San Salvador"
Auteur : Organisation "FMLN"
Cible humaine : "Roberto Garcia Alvarado"

On constate que les champs extraits sont pertinents au sens de la tâche d'extraction considérée : les attentats en Amérique du sud. Concernant la «structuration» de l'information extraite, elle désigne les relations qui existent entre les différents champs extraits : ici ces relations sont très fortes puisque tous les champs sont relatifs à l'attentat. En se basant sur ces relations, il est possible de créer une phrase générique du type *Le <Date de l'incident> à <Lieu de l'incident> en Amérique du sud, <Auteur> a réalisé un attentat contre <Cible humaine>*. Lorsque différents champs sont extraits comme dans cet exemple, on parle d'extraction d'information «multislot».

Un autre exemple d'extraction d'information à partir d'une dépêche est présenté en figure 2.13 (l'exemple est tiré de [Bikel 99]). Les noms de lieu, de personnes et d'organisations ont été extraits de textes anglais et espagnols. Remarquons dans ce cas que la structuration des informations extraites est beaucoup moins évidente puisque les relations entre les différents noms propres ne sont pas établies.

L'extraction d'information dans des documents vise donc à renseigner un certain nombre de champs pertinents en parcourant le document. Récemment, l'explosion du nombre de documents a motivé une automatisation du processus d'extraction d'information, pour les raisons suivantes :

– L'extraction d'information pertinente d'une base de document permet d'effectuer une indexation des documents en fonction de l'information recherchée. Le stockage de l'information pertinente dans une base de données autorise des re-

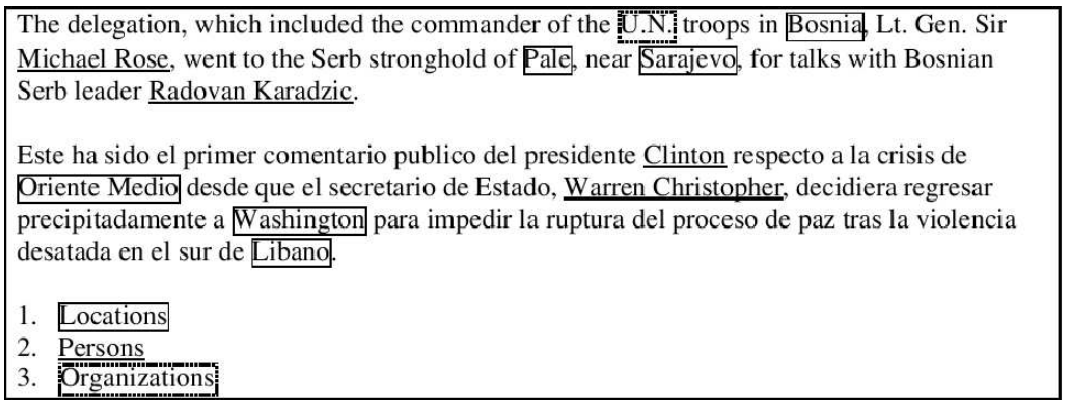


FIG. 2.13 – Exemple d'extraction d'information dans des textes anglais et espagnol provenant de [Bikel 99].

cherches ultérieures. L'automatisation de l'extraction d'information se justifie lorsque le nombre de documents traités est important et/ou que l'opération d'extraction d'information manuelle est fastidieuse ou impossible (cas de documents anciens et fragiles difficilement manipulables). L'automatisation peut également se révéler précieuse dans le cas d'une base dynamique de documents comportant de nouvelles entrées fréquentes.

- L'extraction d'information pertinente permet ensuite une catégorisation ou un tri automatique des documents.
- L'extraction automatique d'information peut également faciliter la lecture d'un document en mettant en évidence certains mots, groupes de mots ou passages.
- Dans le domaine de la fouille, la génération d'une base de données issue d'un processus automatique d'extraction d'information peut également permettre de découvrir de nouvelles connaissances par l'apprentissage de règles ou de statistiques.

Les applications d'une extraction automatique d'information sont donc nombreuses : traitement de rapports de filature d'une agence de surveillance, gestion de dépêches d'une agence de presse, manipulation de rapports d'incidents d'une compagnie d'assurances, etc. Entre 1987 et 1998, les "Message Understanding Conferences" (MUC [MUC 91, MUC 92, MUC 93, MUC 95, MUC 98]) organisées par l'ARPA (Advanced Research Projects Agency) ont encouragé la recherche en extraction d'information en vue d'améliorer le management de l'information dans le secteur militaire. Ces conférences américaines organisaient chaque année une évaluation des méthodes d'extraction d'information dans de larges corpus dont les sujets variaient suivant les années. Lors de MUC1 et MUC2, les corpus étaient constitués de messages de la marine américaine. MUC3 en 1991 puis MUC4 en 1992 ont abordé un corpus de dépêches de presse traitant de récits d'attentats en Amérique du Sud. En 1993, avec MUC5, il s'agissait d'extraire des informations à partir de deux corpus :

des dépêches sur des annonces de fusions / acquisitions d'entreprises, et des documents traitant de microélectronique. En 1995, le corpus de MUC6 était constitué de dépêches sur les nominations d'individus et les changements de position dans les entreprises. En 1998, la dernière des conférences MUC7 était destinée à l'analyse de dépêches sur des lancements de satellites.

Nous décrivons maintenant les différents étapes d'un sytème d'extraction d'information.

2.4.2 Chaîne de traitement pour l'extraction d'information dans des textes en langue naturelle

Selon Soderland [Soderland 94], un système d'extraction d'information dans les textes en langue naturelle intervient à deux niveaux. Une première phase est destinée à structurer le document en l'enrichissant de connaissances à l'aide d'outils spécifiques au traitement automatique des langues : ajout de connaissances lexicales, sémantiques, ou linguistiques connues du système. Une analyse syntaxique peut également permettre d'ajouter des informations syntaxiques ou grammaticales. Dans un second temps, l'étape d'extraction d'information est réalisée par une analyse syntaxique de la structure du message qui transforme la séquence structurée en une représentation cohérente du texte pour le problème considéré. Cette analyse repose sur l'exploitation d'un modèle de connaissance de haut niveau. Nous décrivons brièvement ces deux étapes.

Outils du Traitement Automatique des Langues (TAL)

Les méthodes de TAL destinées à enrichir un texte en langue naturelle pour l'extraction d'information sont généralement constituées de différents niveaux de traitement :

- La segmentation consiste à segmenter le texte en unités lexicales et à y repérer des marques de paragraphe ou autres marques indiquant la structure logique du document. On obtient alors les mots sous leur forme dite «fléchie». Cette étape n'est pas triviale à cause des sigles et des abbréviations possédant des points pouvant être confondus avec des fins de phrase. Par exemple : *il marche vers l'U.F.R.* → *il / marche / vers / l' / U.F.R.*
- L'analyse morphologique et lexicale associe aux mots sous leurs formes fléchies un lemme accompagné de propriétés morphologiques, syntaxiques et sémantiques. Au cours de cette étape, des variables spécifiques sont associées à chaque mot : lemme, type grammatical, genre et nombre du mot. On obtient alors une représentation sémantique et morpho-syntaxique des mots pour chaque lemme. Par exemple, pour le mot *marche* :
  1. *lemme : marcher, catégorie : verbe, nombre : singulier, personne : première ou troisième, temps : présent, mode : indicatif ou subjonctif.*
  2. *lemme : marche, catégorie : nom, nombre : singulier, genre : féminin.*
  L'étiquetage est réalisé à l'aide de dictionnaires des formes fléchies contenant pour chaque forme fléchie sa base lexicale, sa catégorie grammaticale et ses

variations.

– L'analyse syntaxique est une étape de «décodage» qui fournit la structure grammaticale des phrases à partir de la représentation lexicale et morpho-syntaxique de la séquence de mots. Il s'agit donc d'une analyse de séquence, qui peut être complète ou partielle, suivant l'objectif. L'analyse syntaxique partielle («Shallow Parsing» ou «chunking») est une technique permettant d'obtenir une compréhension de la structure de la phrase, sans pour autant réaliser une analyse intégrale du texte sous la forme d'un arbre. La sortie d'un chunker est une division de la phrase en une série de groupe grammaticaux (nom, verbe ou phrase prépositionnelle). Même partielle, cette analyse syntaxique permet d'extraire de l'information. En opposition, le «Part Of Speech» tagging cherche à trouver la meilleure séquence de tag pour une séquence de mots donnée. Contrairement au chunking, chaque mot est étiqueté.

Analyse de la structure du message

A l'issue de l'analyse syntaxique, on dispose de textes enrichis de connaissances lexicales, syntaxiques, grammaticales, etc. En exploitant des connaissances de plus haut niveau telles que les relations entre les différentes entités, on peut effectuer une analyse visant à extraire le sens général des phrases. Cette analyse de la structure du message va ainsi générer une représentation sémantique du texte, plus abstraite que la représentation syntaxique, qui permet l'extraction des informations recherchées. L'analyse de la structure du message suppose une modélisation de celui-ci. Comme nous l'avons déjà mentionné, la modélisation peut être faite soit en profondeur sur les informations pertinentes, soit en surface sur les entités non pertinentes. Pour cela, on peut distinguer deux types de méthodes :

Les *approches à base de règles* : issues des techniques de traitement automatique des langues, elles cherchent à modéliser les relations entre les entités du message pour détecter les champs d'intérêts dans les textes. Les premiers travaux ont utilisé des approches à base de règles et de conjonction de règles définies à la main. La définition manuelle des règles étant particulièrement fastidieuse, des travaux fondés sur des techniques d'apprentisage automatique de règles ont ensuite vu le jour, reposant sur la programmation logique inductive [Muggleton 92] ou l'inférence grammaticale [Gold 67, Angluin 87].

Les *approches statistiques* cherchent à effectuer une modélisation statistique du message à extraire. Les premiers travaux se sont basés sur les approches utilisées pour les taggers syntaxiques [E.Charniak 93]. Ce type d'approche repose sur les outils statistiques pour la modélisation de séquence, en particulier les modèles de Markov cachés ou, plus récemment, les champs conditionnels aléatoires. Dans [Bikel 99], un système d'extraction de noms de lieu, de personnes ou d'organisations, de dates et de nombres est effectué sur les textes de MUC6 et MUC7. Dans ce système, les mots sont étiquetés par le HMM soit en tant que nom de personne, de lieu, etc., soit avec l'étiquette «NOT-A-NAME». Les états du HMM correspondent à ces étiquettes (voir figure 2.14).

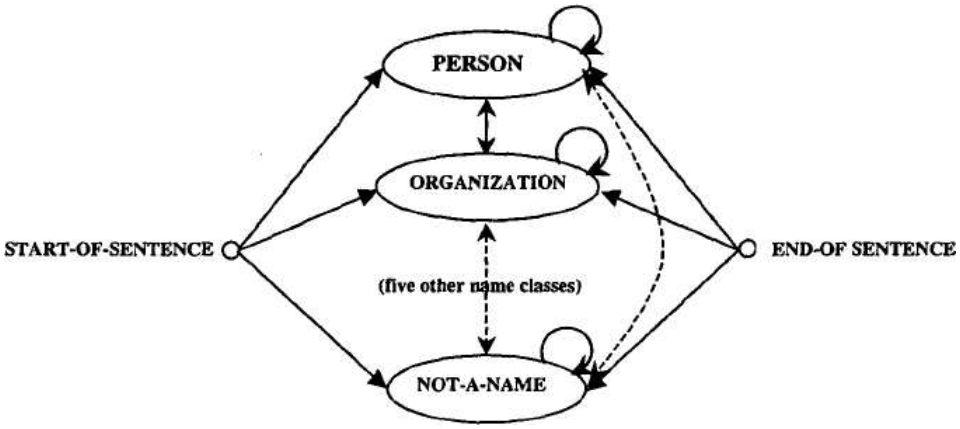


FIG. 2.14 – Modèle de phrase utilisé dans [Bikel 99] pour l'extraction de noms de personne, d'organisation, etc. dans des textes en langue naturelle.

Dans [Freitag 99], les HMM sont appliqués à l'extraction d'informations telles que le titre du document, l'abstract, etc. à partir de textes. Pour cela, un HMM est utilisé, où chaque état représente une étiquette particulière. Dans [Leek 97], les HMM sont utilisés pour l'extraction d'information factuelle à partir d'un corpus de prose anglaise. Dans [Zaragoza 98], les HMM sont appliqués au problème d'extraction d'information dans les dépêches de nomination des individus dans les entreprises de la conférence MUC6.

Les champs aléatoires conditionnels (ou Conditional Random Field : CRF) se situent dans un cadre probabiliste et sont basés sur une approche conditionnelle pour étiqueter et segmenter les séquences de données [Lafferty 01]. Le principal avantage des CRF sur les HMM est qu'ils permettent de relacher les hypothèses faites sur l'indépendance des observations. En effet, les modèles conditionnels considèrent la probabilité conditionnelle $p(x|y)$ plutôt que la probabilité jointe $p(x, y)$. Contrairement aux modèles génératifs, on ne cherche donc pas à modéliser les observations. Plusieurs expériences ont montré la supériorité des CRF par rapport aux HMM sur des problèmes réels [Lafferty 01, Pinto 03]. Les CRF ont en particulier été appliqués sur des problèmes d'extraction d'information, pour le parsing (POS tagging) [Kristjannson 04, Lafferty 01], le shallow parsing [Sha 03] ou l'annotation sémantique [Cohn 05].

Si ces outils d'extraction d'information ont été appliqués avec succès sur les documents électroniques, nous nous intéressons maintenant à leur application aux documents manuscrits.

2.4.3 Application des techniques d'extraction d'information aux documents manuscrits

Etant donné le succès des systèmes d'extraction d'information sur les textes en langue naturelle, on peut se demander si les méthodes d'extraction d'information décrites précédemment peuvent être appliquées aux textes manuscrits.

Rappelons que les méthodes pour l'extraction d'information dans les textes naturels sont composées de deux phases : une première phase destinée à ajouter des informations au texte en segmentant le texte en mots, puis en associant à chaque mot des informations lexicales, sémantiques, linguistiques ou syntaxiques à l'aide de dictionnaires de formes fléchies. Dans un second temps, une analyse syntaxique est chargée d'analyser le texte dans sa globalité plus ou moins superficiellement, afin d'identifier les passages pertinents.

Dans l'hypothèse d'une adaptation aux textes manuscrits des méthodes d'extraction d'information sur les textes en langue naturelle, la première phase semble difficilement applicable à l'écriture manuscrite. En effet, l'enrichissement du texte par des connaissances suppose une «reconnaissance» du texte parfaite, c'est-à-dire que l'on suppose que le texte a pu être segmenté en mots et que chaque mot a pu être identifié. Dans le cas de textes manuscrits, nous avons vu dans la section 2.3.4 que la segmentation d'un texte en mots nous confrontait au paradoxe de Sayre selon lequel les mots ne peuvent être localisés sans avoir été reconnus au préalable, et inversement. La reconnaissance des entités, immédiate et sans faille dans le cas des textes numériques grâce aux dictionnaires de formes fléchies, génère au contraire de nombreuses erreurs et incertitudes dans le cas de textes manuscrits.

Certains travaux ont cherché à prendre en compte les erreurs et les incertitudes d'une reconnaissance. Dans [Ishitani 01, Miller 00, Taghva 04], il s'agit des erreurs provenant d'une reconnaissance OCR de documents numérisés. Dans [Mulbregt 98, Miller 00], des textes sont prononcés oralement et soumis à un processus de reconnaissance de la parole produisant également des erreurs. Dans [Ishitani 01] par exemple, les auteurs répertorient lors d'une phase d'apprentissage les erreurs les plus fréquentes de l'OCR sur chaque mot. En phase de décision, cette connaissance *a priori* est exploitée afin de corriger les erreurs de l'OCR. Dans [Miller 00], un OCR est passé sur des textes en langue naturelle au niveau lettre (approche non dirigée par le lexique). Les mots sur lesquels l'OCR produit au moins une erreur au niveau lettre sont détectés puisqu'ils n'appartiennent pas au dictionnaire (OOV : Out Of Vocabulary), et sont ignorés pour la phase d'extraction d'information. Une telle approche n'est pas transposable au cas de l'écriture manuscrite pour deux raisons : premièrement, elle suppose d'avoir localisé les mots du texte ce qui, nous l'avons vu, est particulièrement délicat dans le cas du manuscrit. Deuxièmement, l'approche est basée sur la non prise en compte des mots dont le résultat d'une reconnaissance sans lexique n'appartient pas au dictionnaire. Concernant l'écriture manuscrite, on peut imaginer les faibles performances que produirait une méthode de reconnaissance de mots sans lexique dans le cas d'un grand lexique.

Ainsi, le caractère incertain des techniques d'analyse de structure de textes ma-

nuscrits (segmentation en ligne et surtout en mots) et l'incertitude engendrée par les méthodes de reconnaissance des entités manuscrites interdisent l'utilisation des techniques classiques de tagging pour l'extraction d'information sur des textes manuscrits.

Puisque ces méthodes ne peuvent être appliquées directement, on peut se demander quelle stratégie adopter pour l'extraction d'information dans les documents manuscrits. C'est ce que nous abordons dans la section suivante.

2.5 Stratégies pour l'extraction de champs numériques dans des courriers entrants

Dans cette section, nous posons la problématique de l'extraction de champs numériques dans les courriers entrants, et nous envisageons les différentes stratégies possibles pour y répondre. Rappelons que les courriers entrants peuvent être considérés comme des documents non contraints puisque nous connaissons seulement la langue et l'orientation privilégiée des lignes de texte.

2.5.1 Un problème d'extraction d'information dans les images de document

Dans la mesure où la reconnaissance de séquences numériques et donc de champs numériques ne pose plus vraiment de problème lorsque ceux ci sont isolés (voir section 1.5), la réelle difficulté du problème consiste à localiser les champs d'intérêt dans les courriers manuscrits.

Or, selon le paradoxe de Sayre, la localisation des entités ne peut se faire qu'avec une étape de reconnaissance des entités. Les champs numériques pouvant apparaître n'importe où dans le document, deux solutions s'offrent alors à nous : une reconnaissance intégrale du document, ou une méthode de localisation et de reconnaissance des champs numériques.

Nous avons déjà mentionné les difficultés qu'engendrerait la reconnaissance intégrale d'un document manuscrit libre. Rappelons la difficile réalisation d'un modèle complexe des lignes de texte intégrant toutes les entités susceptibles d'être rencontrées dans une ligne de texte quelconque : intégralité des mots du dictionnaire français et leurs déclinaisons, champs numériques, bruit, chiffres n'appartenant pas à des séquences numériques recherchées, ponctuation, symboles, etc. La réalisation du moteur de reconnaissance associé permettant de reconnaître tous les caractères élémentaires de ces entités (chiffres, lettres, bruit, ponctuation) poserait également des problèmes de fiabilité, même si certains travaux ont cherché à développer de tels classifieurs [Prevost 03].

La deuxième solution de localisation et de reconnaissance des champs paraît ainsi plus réaliste. Elle nous rapproche alors des méthodes d'extraction d'information abordées dans la section 2.4.

On peut toutefois se demander si l'extraction des champs numériques dans les courriers entrants est bien un problème d'extraction d'information au sens de la définition donnée dans la section 2.4.1 : *l'extraction d'information consiste en l'analyse superficielle de texte en langue naturelle en vue d'une identification de champs pertinents entre lesquels il existe une relation.* D'après les définitions des courriers entrants et des champs numériques des sections 2.2.1 et 2.2.2, il apparaît que les courriers entrants sont effectivement des documents non structurés, et que les champs numériques constituent une information pertinente pour notre application puisqu'ils permettent un tri du courrier, une identification du client, etc. On peut en revanche discuter de la notion de relation entre les champs pertinents. Au niveau champ numérique, il est difficile de parler de relations puisqu'on ne peut pas affirmer *a priori* que deux champs sont relatifs à la même personne. Cependant, rappelons que contrairement aux travaux concernant l'extraction d'information dans les textes électroniques, il s'agit ici d'extraire de l'information à partir d'une image de document. La localisation et la reconnaissance d'un champ numérique dans une image de document peut ainsi être vue comme une extraction multislot de chiffres entre lesquels il existe de fortes relations. En se plaçant à un niveau encore inférieur, on peut considérer la localisation et la reconnaissance de champs numériques du point de vue de l'image ; l'opération peut alors être vue comme l'extraction de pixels groupés en chiffres, eux-même regroupés en un champ numérique d'un type donné. La localisation et la reconnaissance de champs numériques dans des images de document peut donc être considérée comme un réel problème d'extraction d'information.

Comme nous venons de le voir, les méthodes classiques d'extraction d'information ne sont toutefois pas directement applicables à l'écriture manuscrite, et doivent ainsi être adaptées afin de prendre en compte l'incertitude liée à la reconnaissance de l'écriture manuscrite.

Qu'il s'agisse des travaux d'extraction d'information dans les documents manuscrits [Koch 06, El-Yacoubi 02] ou dans des documents en langue naturelle [Miller 00], nous avons pu constater que toutes les stratégies pour l'extraction d'information reposaient sur une modélisation des lignes de texte ou des phrases du document intégrant à la fois l'information d'intérêt (mots clefs, noms de rue, noms propres, etc.), et le reste du document : mots hors lexique, numéro de rue, mots non pertinents, ponctuation, etc. Cette modélisation présente l'avantage de pouvoir s'aligner et donc segmenter n'importe quelle phrase ou n'importe quelle ligne de texte, tout en couplant une modélisation précise des entités recherchées avec une modélisation plus grossière des informations non pertinentes. Dans le cas de l'extraction des champs numériques, la modélisation de la phrase n'apportant *a priori* aucun bénéfice pour la localisation et la reconnaissance des champs, nous nous orientons vers une modélisation des lignes de texte. Le modèle d'une ligne de texte pouvant contenir un champ numérique peut ainsi être représenté par la figure 2.15.

Si un tel modèle permet d'extraire les champs numériques, il soulève plusieurs questions :

- Comment modéliser les champs numériques ?
- Comment modéliser les informations non pertinentes ?

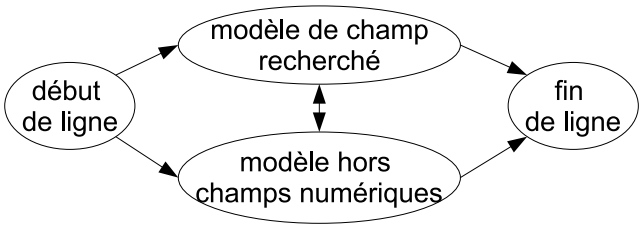


FIG. 2.15 – Modèle de ligne de texte pouvant contenir un champ numérique.

– Quels sont les différents traitements à mettre en œuvre et comment les enchaîner ?

La première question consiste à se demander quelles sont les connaissances *a priori* dont nous disposons pour modéliser les champs numériques. Si l'on ne peut pas bénéficier de l'apport d'un lexique comme dans le cas de la modélisation des mots, il est possible de modéliser la syntaxe de chaque type de champ numérique. En effet, les séquences de chiffres recherchées obéissent à un certain nombre de règles syntaxiques plus ou moins fortes suivant le type de champ considéré : le nombre de chiffres, la présence et la position d'éventuels séparateurs sont généralement connus. Par exemple, un numéro de téléphone français est toujours constitué de dix chiffres regroupés par paires éventuellement séparées par des points ou tirets (voir figure 2.2). Soulignons que ces règles syntaxiques relatives aux champs sont les seules connaissances *a priori* dont nous disposons pour le problème difficile d'extraction des champs numériques dans des documents quelconques. Elles doivent donc être exploitées en les injectant dans les modèles de champs recherchés.

La modélisation des informations non pertinentes est un problème difficile à cause de la diversité et de la variété des formes qui les composent. Dans le cas de l'extraction des champs numériques, les informations non pertinentes sont constituées des informations textuelles (mots), de la ponctuation, du bruit, des chiffres n'appartenant pas aux champs recherchés, de symboles, etc. Deux modélisations sont possibles :

– Une modélisation relativement fine inspirée de [Koch 06] où un modèle ergodique contenant tous les modèles de caractères, de chiffres et de symboles susceptibles d'être rencontrés dans les entités non pertinentes peut être réalisée. On peut ainsi modéliser n'importe quel enchaînement de caractères textuels et numériques en autorisant toutes les transitions entre les entités. Cette modélisation «générique» des informations non pertinentes suppose toutefois l'utilisation d'un moteur de reconnaissance capable d'identifier toutes les classes d'entités susceptibles de constituer le rejet : lettres minuscules et majuscules, chiffres, ponctuaction, symboles, etc.
– Comme nous ne cherchons pas à reconnaître les entités non pertinentes, il est possible de mettre en œuvre une modélisation plus grossière des entités non pertinentes. Par exemple, les 26 classes de lettres peuvent être regroupées

en une seule classe ; les 10 classes de chiffres également ; etc. En poussant ce raisonnement à l'extrême, la modélisation la plus simple consiste à ne définir qu'une seule et unique classe de «rejet» englobant toutes les formes non pertinentes. C'est la méthode utilisée par Bikel [Bikel 99] pour absorber toutes les informations non pertinentes lors de l'extraction de noms propres dans des textes électroniques (voir figure 2.14). En appliquant cette solution radicale à notre problème, une classe unique de rejet contiendrait toutes les informations non pertinentes n'appartenant pas à un champ numérique : mots, fragments de mots, ponctuation, bruit, etc.

Enfin la troisième question consiste à se demander quelles sont les étapes nécessaires à la réalisation de la tâche d'extraction des champs numériques dans les courriers manuscrits, et comment enchaîner ces étapes. Comme nous avons opté pour une modélisation des lignes de texte pour localiser les champs, une étape de segmentation du document en lignes s'impose. Afin d'aligner les séquences de composantes sur les modèles de ligne, une étape de distinction entre les entités pertinentes (composantes numériques) et non pertinentes est également nécessaire. Enfin une étape de reconnaissance des entités numériques est requise afin de déterminer la valeur des champs. À partir de la segmentation du document en lignes de texte, nous avons ainsi identifié les 3 étapes de traitement incontournables pour l'extraction des champs numériques dans les courriers manuscrits :

- Distinction entre les composantes numériques/non numériques (rejet des composantes non numériques).
- Reconnaissance des composantes numériques.
- Localisation des champs à l'aide des contraintes syntaxiques du modèle de ligne.

Il est alors possible d'envisager deux grandes stratégies suivant l'ordre dans lequel sont appliqués ces traitements (voir figure 2.16).

La stratégie la plus intuitive consiste à appliquer séquentiellement les 3 étapes dans l'ordre dans lequel elles ont été mentionnées ci-dessus : premièrement, l'identification des composantes numériques à l'aide d'un classifieur chiffres propose des hypothèses de localisation des informations pertinentes. Cette première étape peut également être vue comme une phase de rejet des composantes non numériques. La localisation des champs peut alors être effectuée en alignant les hypothèses de localisation et de reconnaissance chiffre sur les modèles de lignes. Dans cette stratégie, l'étape d'extraction s'appuie sur la reconnaissance des entités numériques pour reconstruire les champs numériques pertinents vérifiant la syntaxe du modèle.

La seconde stratégie, moins intuitive, consiste à effectuer l'étape de localisation des champs numériques le plus tôt possible dans la chaîne de traitement, avant la phase de reconnaissance. Dans un tel cas de figure, la phase de reconnaissance est appliquée en fin de chaîne de traitement, uniquement sur les séquences localisées auparavant. Les phases de localisation et de reconnaissance des champs sont ainsi complètement dissociées.

Nous venons de dégager deux stratégies générales pour l'extraction de champs numériques dans des documents manuscrits faiblement contraints. Dans cette thèse,

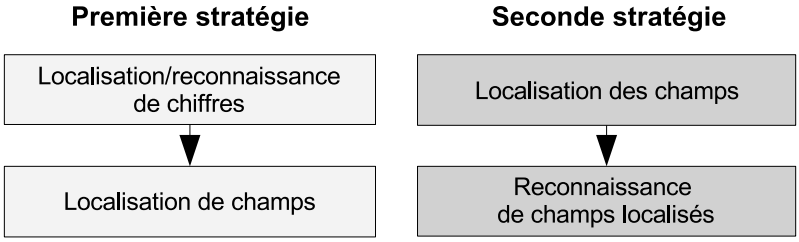


FIG. 2.16 – Stratégies générales pour l'extraction des champs numériques

nous avons choisi d'implémenter ces deux stratégies afin de les comparer. Il existe toutefois de nombreuses manières de les mettre en œuvre ; nous étudions donc maintenant les différents choix relatifs à la mise en œuvre de ces deux stratégies, et discutons en particulier du problème central de la segmentation.

2.5.2 Première approche : une stratégie de segmentation / reconnaissance / rejet

On souhaite donc mettre en place une méthode d'extraction des champs numériques fondée sur une reconnaissance numérique appliquée sur l'intégralité du document. Cette étape est suivie d'une phase de dicrimination des composantes numériques/non numériques. La localisation des champs est ensuite effectuée par l'alignement des hypothèses de localisation/reconnaissance de chiffres sur les modèles intégrant la connaissance *a priori* relative aux champs.

Nous devons ainsi mettre en place une stratégie de reconnaissance d'entités numériques. Les approches holistiques (ou approches globales) n'étant pas applicables ni aux champs numériques, ni aux informations non pertinentes, les approches étudiées seront obligatoirement analytiques, c'est-à-dire qu'une segmentation doit être mise en œuvre pour traiter les lignes de texte.

Une étape de rejet doit également être mise en place pour écarter tout ce qui n'appartient pas à un champs numérique (mots, fragments de mots, ponctuation, bruit, etc.). En comparant cette stratégie avec les méthodes de reconnaissance de séquences numériques basées sur une strategie de segmentation/reconnaissance, on constate que le paradigme segmentation/reconnaissance est alors augmenté d'un degré de liberté puisqu'il nous faut à la fois segmenter, reconnaître et rejeter les éléments d'une ligne de texte. On voit ainsi apparaître toute la complexité d'une telle tâche : en effet, il faut segmenter pour reconnaître et reconnaître pour segmenter, mais il faut également reconnaître pour rejeter et rejeter pour reconnaître, segmenter pour rejeter et rejeter pour segmenter ! Schématiquement, on peut représenter ce triple paradoxe par la figure 2.17

En supposant que nous sommes capables de fournir les hypothèses de segmentation/reconnaissance/rejet sur une ligne d'écriture, l'extraction des champs consiste alors à prendre une décision globale de segmentation/reconnaissance/rejet sur une

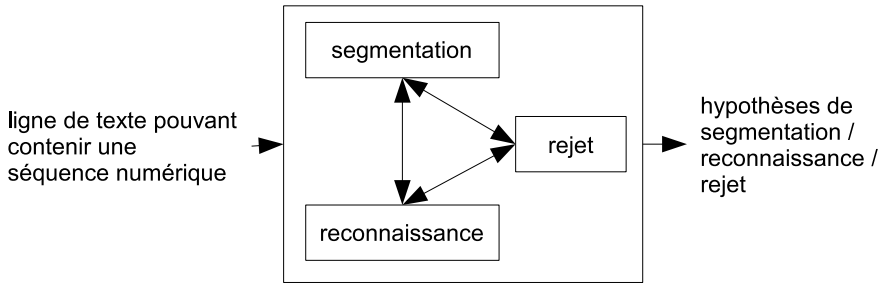


FIG. 2.17 – Stratégie de segmentation/reconnaissance/rejet.

ligne de texte intégrant les contraintes syntaxiques connues du système.

Un point central concernant la mise en place de cette stratégie est la segmentation des composantes. Deux segmentations peuvent être mises en œuvre : segmentation explicite ou segmentation implicite par une fenêtre glissante.

Cas d'une segmentation explicite

Dans ce cadre, la modélisation des champs numériques représente les champs sous la forme d'une succession de chiffres agencés selon les règles syntaxiques connues du système pour chaque type de champ. Prenons l'exemple d'un numéro de téléphone français : ils sont constitués de 10 chiffres quelconques regroupés en paires pouvant être séparés par des séparateurs. Le modèle du champ «numéro de téléphone français» peut donc être représenté par la figure 2.18 ou [0..9] désigne n'importe quel chiffre compris entre 0 et 9, et S désigne un séparateur.

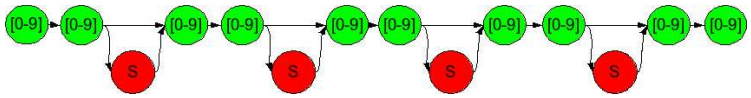


FIG. 2.18 – Modèle de champ numérique de type «numéro de téléphone».

Comme pour la reconnaissance de séquences numériques ou de mots, deux stratégies peuvent être appliquées selon que la reconnaissance et la segmentation sont menées séquentiellement ou conjointement : *segmentation puis reconnaissance* ou *segmentation/reconnaissance* (voir section 1.5). Rappelons le caractère beaucoup plus fiable [Fujisawa 92] mais aussi plus coûteux en temps de calculs de la seconde méthode où les hypothèses de segmentation sont validées par la reconnaissance. L'extraction des champs est alors effectuée en alignant les observations provenant du classifieur sur le modèle de ligne de texte.

Les stratégies issues d'une segmentation explicite en chiffres sont calquées sur les méthodes de reconnaissance de séquences numériques. Si la modélisation de la ligne de texte ne détaille que les entités pertinentes, la segmentation et la reconnaissance (éventuellement couplées) sont en revanche systématiquement appliquées sur toutes

les composantes d'une ligne de texte en vue d'y localiser tous les chiffres. L'avantage de ces stratégies est toutefois leur simplicité de mise en œuvre et leur forte capacité à discriminer les séquences.

Cas d'une segmentation implicite

Dans l'hypothèse de l'application d'une stratégie de segmentation implicite, il faudrait disposer de classifieurs dynamiques de type modèles de Markov cachés ou réseaux de neurones récurrents afin de réaliser conjointement la localisation et la reconnaissance des entités. Dans ce cas, la modélisation des champs numériques reposerait sur une concaténation de modèles de chiffres agencés selon les règles syntaxiques du type de champ recherché, selon le même modèle que pour la segmentation explicite. En ce qui concerne le rejet, il peut dans le cas de l'utilisation des HMM être modélisé statistiquement par un modèle ergodique de caractères ou d'états appris sur une base d'entités non pertinentes. Dans le cas des réseaux de neurones récurrents, une classe de rejet unique pourra être considérée.

Les déclinaisons des stratégies basées sur une segmentation implicite permettent de contourner le problème délicat de la segmentation. Dans le cas de l'utilisation de HMM, un apprentissage statistique à l'aide de l'algorithme de Baum-Welsh assure une modélisation efficace des lignes de texte, et l'algorithme de Viterbi permet d'aligner la séquence d'observation sur les modèles de ligne. En revanche, on connaît les lacunes de ce type d'approche pour la discrimination des séquences. Comme nous ne disposons pas de lexique permettant de limiter le nombre de valeurs que peut prendre un champ numérique, l'utilisation des HMM seuls n'est donc pas recommandée. Ces derniers peuvent toutefois être couplés avec un classifieur de type réseau de neurones afin de bénéficier des qualités discriminantes de ceux-ci, ainsi que de l'existence d'algorithmes intégrés permettant l'apprentissage conjoint des HMM et du réseau de neurones (voir section 1.5.3)

Cette première stratégie peut donc se décliner en deux approche selon qu'on considère une segmentation explicite ou implicite. Dans l'optique de la mise en œuvre de cette stratégie, il nous a fallu faire un choix entre les deux types de segmentation. Bénéficiant d'une certaine expertise dans les stratégies à segmentation explicite mises en œuvre dans les travaux précédents de l'équipe [Heutte 97, Koch 04, Nosary 02], nous avons privilégié ce type d'approche.

2.5.3 Seconde approche : une stratégie dirigée par la syntaxe

Contrairement à la première approche où la localisation cloturait la chaîne de traitement, nous souhaitons ici faire intervenir la localisation des champs le plus rapidement possible. Nous envisageons ainsi la mise en oeuvre de l'extraction des champs par une méthode en deux étapes :

– La première étape est chargée de localiser les champs numériques sans faire appel à un reconnaisseur chiffre. Le but est d'extraire rapidement des séquences de composantes constituant les champs d'intérêt, et de rejeter le reste du

document.
- La deuxième étape procède à la reconnaissance des entités localisées : les séquences de composantes extraites par la première étape sont soumises à un moteur de reconnaissance de champs qui détermine leur valeur numérique. On se ramène alors aux méthodes classiques de la littérature (voir section 1.5).

L'approche proposée ici pour la localisation des champs est basée sur une modélisation markovienne d'une ligne de texte. Ce modèle exploite la syntaxe spécifique des champs numériques que l'on souhaite extraire (nombre de chiffres, présence et position de séparateurs) pour parvenir à localiser les séquences numériques, sans toutefois procéder à la reconnaissance des chiffres. C'est en effet par une étape de pré-reconnaissance «syntaxique» que l'on va chercher à interpréter globalement les lignes de texte. Des classes syntaxiques doivent ainsi être définies afin de décrire la nature alphabétique ou numérique des composantes, sans pour autant préciser leur valeur numérique.

Ici encore, plusieurs stratégies de segmentation sont possibles : segmentation explicite ou implicite. Ces deux stratégies nécessitent toutefois une étape de reconnaissance pour fiabiliser leur résultat. Une troisième possibilité, plus originale, consiste à ne pas effectuer de segmentation. En effet, on effectue généralement une segmentation des composantes pour éviter d'avoir à considérer les 10^n classes de nombre pour une composante qui contiendrait n chiffres connectés. Comme la valeur numérique des composantes n'est pas requise dans notre cas, on peut réduire la valeur syntaxique d'une composante numérique à son nombre de chiffres et ainsi éviter sa segmentation. Les composantes numériques peuvent ainsi correspondre à un ou plusieurs chiffres, ou même à un séparateur (point, tiret...). De ce fait, on doit introduire dans le modèle de ligne des étiquettes correspondant à ces situations : D (Digit ou chiffre), DD (Double Digits ou chiffres liés), S (Séparateur). Notons que l'on pourrait également chercher à détecter les chiffres liés comportant plus de deux chiffres, ce qui ne modifierait pas fondamentalement l'approche. Toutefois, ces composantes étant très rares, nous n'avons pas considéré ces classes pour le moment. En ce qui concerne les composantes textuelles, le modèle ne comprend qu'une seule classe, appelée classe Rejet, pour décrire l'ensemble des situations possibles : caractère isolé, fragment de mot, mot, diacritique, signe de ponctuation, symbole.

Ces quatres classes syntaxiques constituent les états du modèle markovien, sur lequel on alignera les séquences de composantes de manière à ne conserver que les séquences syntaxiquement correctes. On peut ainsi qualifier cette approche de «dirigée par la syntaxe». En disposant d'un classifieur capable de discriminer ces 4 classes, la localisation d'un champ numérique dans une ligne manuscrite consistera à rechercher dans le treillis fourni par le classifieur la meilleure séquence d'étiquettes valide au sens du modèle de Markov utilisé pour modéliser la ligne (voir figure 2.19).

Signalons qu'un tel modèle ne permet pas de reconnaître les champs puisque la valeur des chiffres n'est pas modélisée. La stratégie est alors radicalement opposée aux stratégies avec segmentation puisque la modélisation vise strictement à localiser le champ sans chercher à le reconnaître.

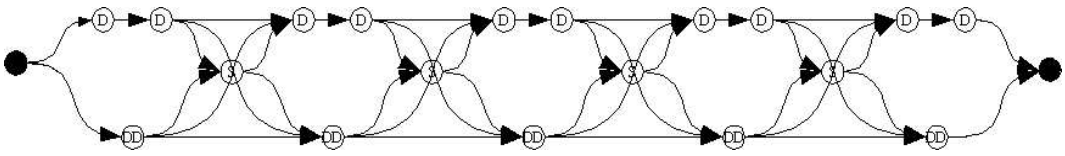


FIG. 2.19 – Modèle de champ numérique de type «numéro de téléphone» avec une stratégie sans segmentation.

Cette méthode d'extraction des champs est une alternative intéressante à l'utilisation d'une stratégie de segmentation-reconnaissance sur l'intégralité du document, puisque seuls les champs extraits sont soumis à un reconnaisseur. L'étape de reconnaissance des champs est alors ramenée à un problème de reconnaissance beaucoup plus contraint, ce qui permet d'envisager des performances intéressantes.

2.5.4 Chaîne de traitement des deux stratégies

Dans la section précédente, nous avons présenté les différentes stratégies envisageables pour l'extraction des champs numériques dans les courriers manuscrits. Nous avons dégagé deux stratégies que nous proposons de mettre en œuvre dans les chapitres suivants. On peut représenter schématiquement l'enchaînement des différentes étapes de traitement pour les deux stratégies par la figure 2.20.

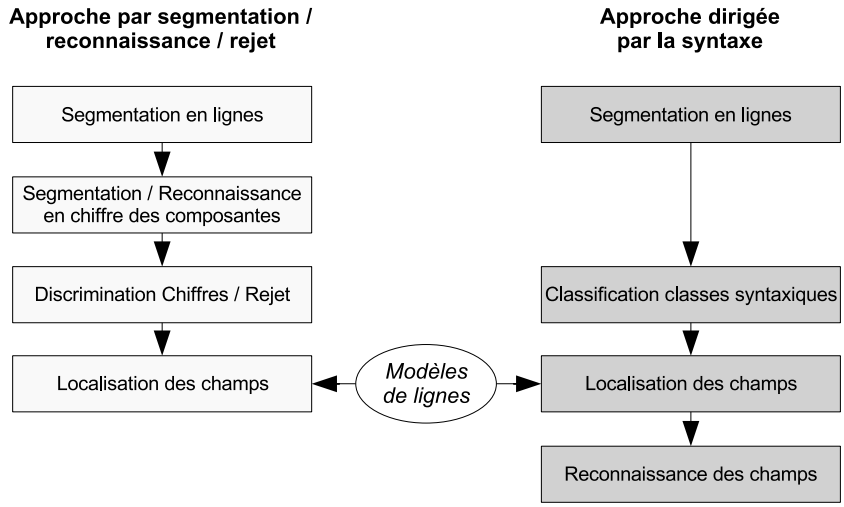


FIG. 2.20 – Enchaînement des modules de traitement pour les deux stratégies.

Les deux stratégies étant fondées sur une modélisation des lignes de texte, la première étape des deux chaînes de traitement est une segmentation du document en lignes de texte. Dans les deux cas, nous souhaitons éviter au maximum la re-

connaissance intégrale du document et par conséquent développer une modélisation la plus grossière possible pour les entités non pertinentes. Nous avons ainsi choisi de modéliser les information non numériques par une classe «Rejet» unique. En revanche, la modélisation des champs numériques doit être plus précise afin d'incorporer dans les modèles de ligne la connaissance *a priori* sur leur syntaxe.

La méthode de *segmentation/reconnaissance/rejet* applique une méthode de localisation/reconnaissance sur l'ensemble du document, alors que la méthode *dirigée par la syntaxe* sépare les phases de localisation et de reconnaissance. La première approche peut être vue comme une extension des stratégies de segmentation-reconnaissance appliquée aux séquences numériques isolées, et constitue l'approche la plus «évidente» qu'on puisse mettre en place. La deuxième approche, plus originale, localise les champs recherchés sans segmentation ni reconnaissance chiffre. Elle a plutôt pour origine les stratégies d'extraction d'information utilisée sur les textes en langue naturelle.

2.6 Conclusion

Dans ce chapitre, nous avons commencé par décrire le contexte de notre étude : l'extraction de champs numériques dans les courriers entrants. Ces champs numériques constituent une information pertinente dans la mesure où leur extraction permet un tri automatique des courriers entrants. Il s'agit d'un problème complexe où les documents traités sont faiblement contraints, et où l'information recherchée n'est pas régie par un lexique.

Afin de positionner le problème, nous avons étudié les différents systèmes existants de lecture automatique de documents et en particulier l'étape de localisation des informations. Nous avons constaté que moins les documents étaient contraints, plus la reconnaissance des entités était utilisée afin de fiabiliser leur localisation. Dans le cas de document très faiblement contraints (textes libres), le paradoxe de Sayre devient incontournable puisque la localisation des entités sans reconnaissance devient impossible, et inversement.

Afin d'éviter le difficile problème de localisation/reconnaissance de la totalité des entités des documents manuscrits, nous nous sommes donc tournés vers les méthodes d'extraction d'information largement employées dans les documents électroniques. Après en avoir décrit le fonctionnement, nous avons ainsi cherché à adapter ces méthodes afin de les rendre applicables aux images de document. Cela ne peut se faire qu'en faisant intervenir des étapes de distinction entre information pertinente/information non pertinente, et de reconnaissance des entités manuscrites. Dans le cadre de notre problème d'extraction de champs numériques, des étapes de localisation et de reconnaissance des composantes numériques sont donc requises.

Après avoir envisagé les différentes stratégies possibles pour notre problème, nous en avons dégagé deux que nous souhaitons mettre en œuvre afin de les comparer. La première est basée sur une stratégie de segmentation/reconnaissance/rejet fournissant des hypothèses de localisation/reconnaissance de chiffres qui permet l'ex-

traction des champs. La seconde procède à une localisation et une reconnaissance des champs disjointes. La phase de reconnaissance n'est ainsi appliquée que sur les séquences localisées, ce qui nous ramène aux méthodes classiques de reconnaissance d'entités isolées.

Nous décrivons dans les deux chapitres suivants la réalisation d'une chaîne de traitement complète permettant la mise en place de ces deux stratégies, et nous montrons que la deuxième approche se révèle plus pertinente pour la problématique posée dans le cadre de cette thèse.

Chapitre 3

Localisation et reconnaissance de champs numériques par une stratégie de segmentation - reconnaissance - rejet

Nous présentons dans ce chapitre la réalisation d'une première chaîne de traitement complète pour la localisation et la reconnaissance de champs numériques dans des documents manuscrits quelconques. Comme discuté dans le chapitre 2, cette chaîne de traitement est fondée sur la stratégie la plus évidente qui consiste à effectuer une localisation et une reconnaissance des chiffres dans le document, pour ensuite localiser les champs recherchés à l'aide des règles syntaxiques connues qui régissent ces champs. La contribution majeure de ce chapitre se situe dans la mise en place d'une stratégie de segmentation/reconnaissance/rejet capable de simultanément localiser et reconnaître les champs numériques dans les textes. Cette stratégie peut être vue comme une extension des méthodes de segmentation-reconnaissance destinée à la reconnaissance de séquences numériques isolées aux séquences numériques entourées de formes à rejeter. Dans la première section de ce chapitre, nous montrons comment l'intégration d'une étape de rejet dans les stratégies classiques de segmentation-reconnaissance nous permet de génerer les hypothèses de localisation et de reconnaissance de chiffres. Nous détaillons et justifions ensuite nos choix pour la réalisation de chaque étape de la chaîne de traitement : segmentation du document en lignes, mise en place d'une stratégie de segmentation-reconnaissance, conception d'un classifieur chiffre, réalisation et intégration d'une méthode de rejet efficace pour l'identification des formes non numériques, et filtrage des hypothèses de segmentation-reconnaissance-rejet valides.

3.1 Une stratégie de segmentation - reconnaissance - rejet

3.1.1 Intégration du rejet dans une stratégie de segmentation - reconnaissance

S'il existe de nombreux travaux mettant en œuvre des stratégies de segmentation/reconnaissance (voir chapitre précédent), peu de travaux traitent le problème du rejet. Notons toutefois la modélisation du rejet par un modèle ergodique pour les mots hors lexique dans les travaux de Koch [Koch 06] et El-Yacoubi [El-Yacoubi 02], mais ces travaux bénéficient de la présence d'un lexique, ce qui n'est pas notre cas. On peut donc se demander (i) comment rejeter les composantes ou graphèmes non numériques, et (ii) comment intégrer cette étape de rejet dans la stratégie de segmentation-reconnaissance. Le premier point est discuté dans la partie 3.5 ; nous admettrons pour le reste de cette section que l'on dispose d'un tel mécanisme de rejet afin de traiter le second point.

Dans le cas d'une segmentation reconnaissance, le système fournit généralement un treillis d'hypothèses de segmentation-reconnaissance à plusieurs niveaux de segmentation, en intégrant des scores de confiance fournis par le classifieur (voir figure 3.1).

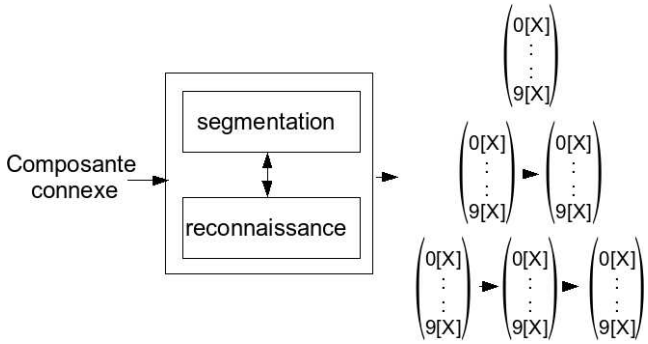


FIG. 3.1 – Treillis d'hypothèses de segmentation et de reconnaissance produit par une stratégie de segmentation-reconnaissance sur 3 niveaux avec scores de confiance associés.

Afin de pouvoir à la fois localiser et reconnaître les chiffres dans les lignes de texte, on souhaite non seulement appliquer une méthode de segmentation reconnaissance sur l'ensemble de la ligne, mais aussi rejeter les formes non numériques pour fournir des hypothèses de segmentation-reconnaissance-rejet. Étant donnée la difficulté du problème de discrimination chiffre/rejet, la décision de rejet ne peut être binaire (acceptation ou rejet des formes). Une décision plus «souple» doit être prise afin de pouvoir remettre en cause les hypothèses de classification rejet. Un moyen simple et efficace est de faire appel aux scores de confiances que les classifieurs usuels sont

généralement capables de produire (MLP, SVM, etc). En admettant que le rejet des composantes soit effectué par un système capable de produire un score de confiance, ces scores peuvent être intégrés au treillis de reconnaissance « classique » par le biais de l'ajout d'une classe aux hypothèses existantes (voir figure 3.2).

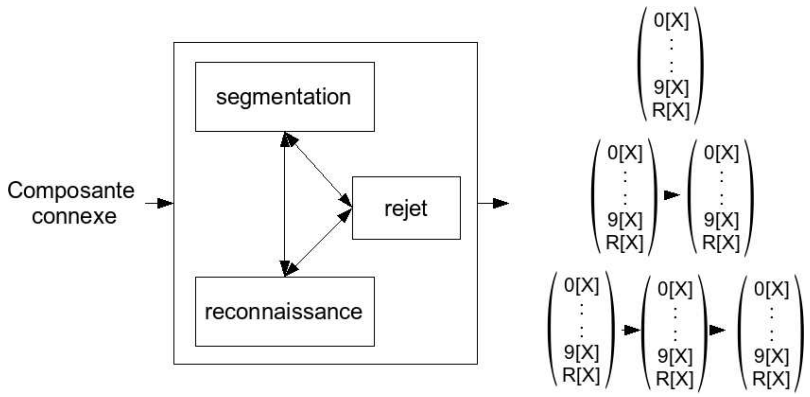


FIG. 3.2 – Prise en compte du rejet dans le treillis de reconnaissance par ajout d'un score de confiance pour la classe rejet.

Appliquée à toutes les composantes d'une ligne de texte, la stratégie de segmentation-reconnaissance-rejet produit le treillis de la figure 3.3.

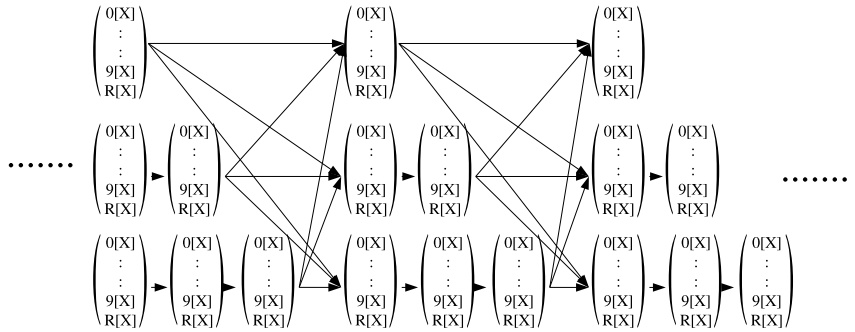


FIG. 3.3 – Treillis de segmentation-reconnaissance-rejet pour une ligne de texte.

A partir de ces hypothèses de segmentation-reconnaissance-rejet, la localisation des champs numériques dans une ligne de texte manuscrite se fait par une recherche de meilleur chemin dans le treillis des hypothèses. Cette recherche doit s'effectuer en intégrant la connaissance *a priori* disponible concernant le type de champ numérique recherché. Par exemple, on sait qu'un code postal français est constitué de cinq chiffres ; une ligne de texte contenant un code postal sera donc constituée d'un certain nombre de composantes rejet suivis de 5 chiffres puis d'une aute série de

composantes rejet. Nous discutons dans la partie 3.6 de la conception des modèles de ligne et de la recherche du meilleur chemin.

3.1.2 Vue globale du système

Nous avons décrit dans la partie précédente la stratégie de segmentation-reconnaissance-rejet utilisée par notre système pour la localisation et la reconnaissance conjointe des champs numériques dans des documents quelconques. Nous présentons dans cette section une vue globale de la chaîne de traitement. Après avoir identifié et décrit brièvement chacun des modules nécessaires à la localisation et à la reconnaissance des champs numériques, nous donnons un schéma global du système complet.

La localisation et la reconnaissance des champs numériques dans des documents quelconques selon la stratégie décrite dans la section précédente a permis d'identifier les modules suivants :

- Comme la localisation et la reconnaissance des champs numériques se fait par une analyse des lignes de texte, un module de segmentation des documents en lignes est nécessaire (voir partie 3.2).
- Une stratégie de segmentation-reconnaissance chiffre classique est appliquée à toutes les composantes (numériques et textuelles) des lignes de texte afin de segmenter et reconnaître les composantes numériques. Elle repose classiquement sur un classifieur chiffre ainsi qu'une méthode de segmentation adaptée aux chiffres. La méthode de segmentation et la stratégie de segmentation-reconnaissance est décrite dans la section 3.3, et le classifieur chiffre est décrit dans la partie 3.4.
- Une stratégie de rejet des composantes non numériques capable de fournir des scores de confiance doit être mise en place. Nous avons développé dans ce chapitre une première méthode basée sur l'exploitation des capacités de rejet intrinsèques du classifieur chiffre (voir section 3.5).
- Enfin, une méthode de recherche de meilleur chemin dans le treillis de segmentation-reconnaissance-rejet sous les contraintes à notre disposition doit prendre une décision globale sur l'ensemble de la ligne de texte pour localiser et reconnaître les éventuels champs numériques (voir partie 3.6).

L'ensemble de ces modules s'articule selon le schéma 3.4.

Nous décrivons dans la suite du chapitre la réalisation de ces modules.

3.2 Segmentation en lignes

3.2.1 Présentation de la méthode

Il existe de nombreuses méthodes de localisation des lignes d'écriture, plus ou moins robustes à la variabilité de l'inclinaison et à la fluctuation des lignes du document traité. Les méthodes developpées peuvent être basées sur l'analyse d'histogrammes [Cohen 91, Nosary 02], sur une analyse complète de la mise en page du

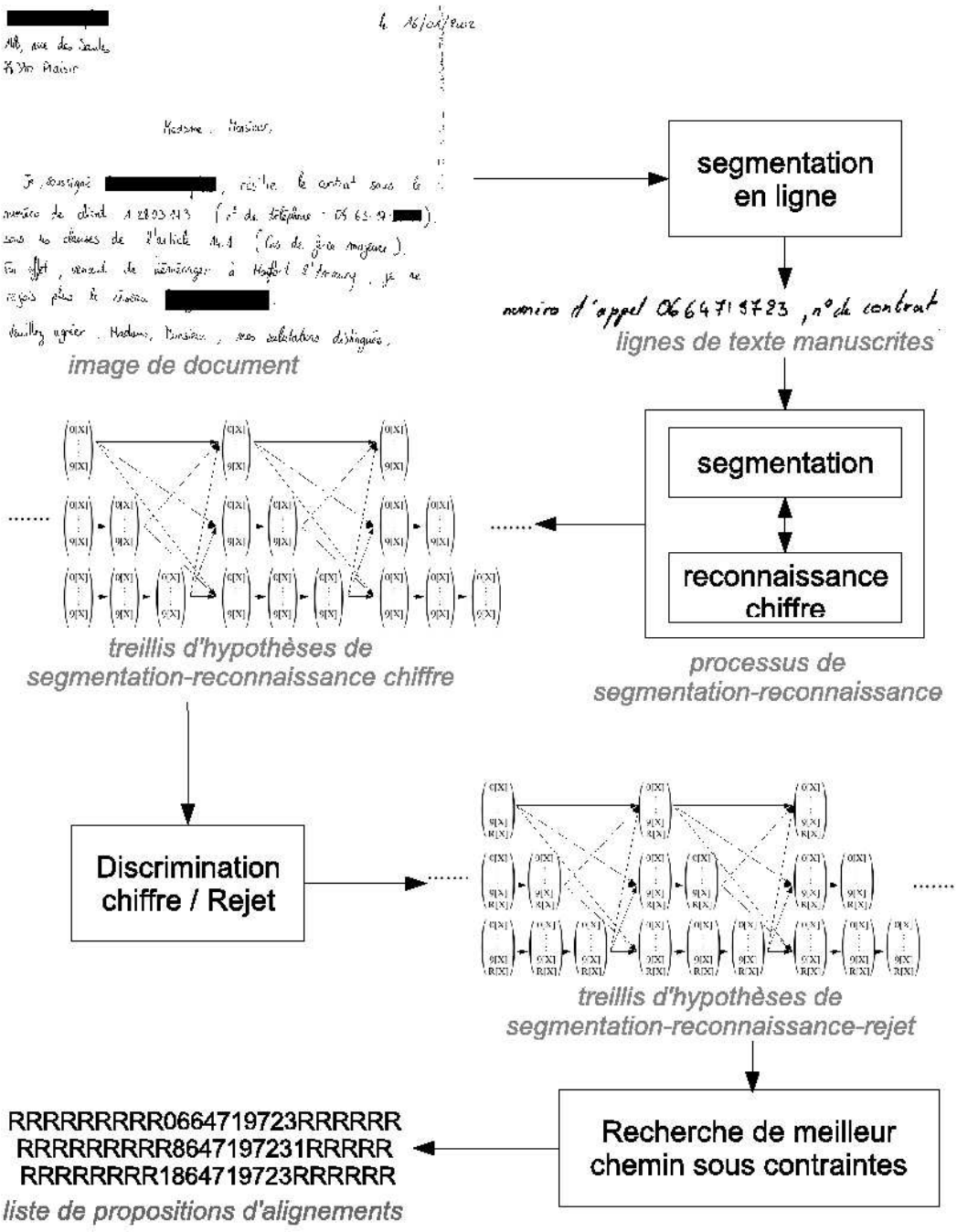


FIG. 3.4 – Architecture du système de localisation et de reconnaissance de champs numériques

document [Likforman-Sulem 95b, Nagy 00], par accroissement de groupes d'entités connexes [Likforman-Sulem 95a], ou par analyse des *minima* et *maxima* des contours [Kim 99].

Les documents traités par notre système, bien que comportant des lignes généralement parallèles et horizontales, possèdent parfois une inclinaison importante et où les extensions hautes et basses interférent avec les lignes supérieures et inférieures (voir figure 3.5).

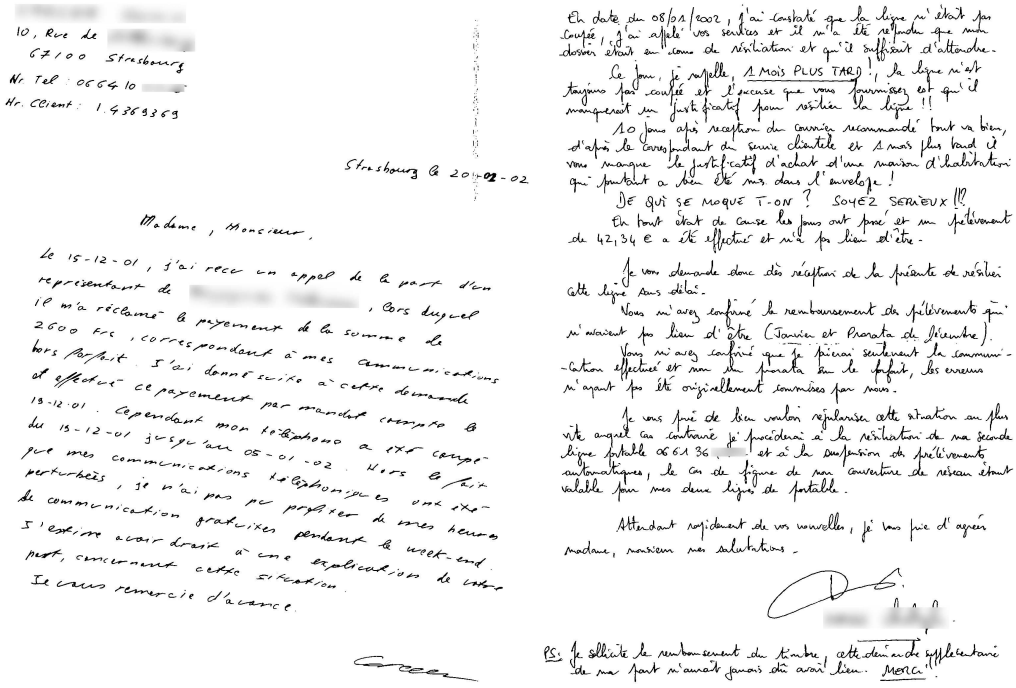


FIG. 3.5 – Inclinaison des lignes et extensions hautes et basses interférant avec les lignes supérieures et inférieures.

Afin de traiter ces problèmes, nous avons choisi une méthode par aggrégation successive inspirée de [Likforman-Sulem 95a] où une méthode de détection des lignes de texte sans connaissance *a priori* de leur orientation est présentée. Cette méthode est basée sur les trois étapes suivantes :

- Association des composantes connexes dont la taille est supérieure à un seuil donné. Ceci permet de ne prendre en compte que les composantes connexes correspondant à des mots ou parties de mot, alors que les signes de ponctuation et les accents sont ignorés dans cette première étape. Le regroupement est réalisé selon un critère de distance favorisant la dimension horizontale (voir figure 3.6 a).
- Fusion d'alignements trop proches : plusieurs alignements peuvent être détectés dans une même ligne de texte. La fusion de ces alignements est réalisée

selon un critère de distance prenant en compte la taille moyenne des inter-lignes sur l'ensemble du document (voir figure 3.6 b).
– Affectation des composantes isolées à la ligne la plus proche : cette dernière phase permet de prendre en compte l'ensemble des composantes connexes ignorées lors de la première étape (voir figure 3.6 c).

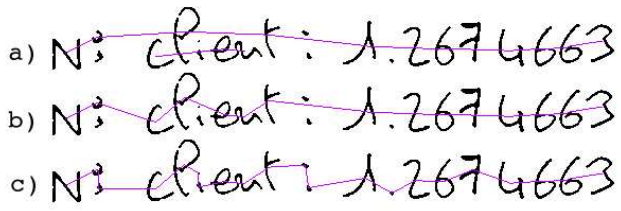


FIG. 3.6 – Étapes de la segmentation en ligne : a) regroupement initial des composantes connexes, b) fusion des alignements trop proches, c) attachement des composantes isolées à la ligne la plus proche.

La figure 3.7 montre le résultat de l'étape de segmentation en lignes sur une image test. Le résultat semble satisfaisant, même sur une image dont les lignes sont assez inclinée. Il convient toutefois d'évaluer notre approche sur plusieurs images.

3.2.2 Évaluation des performances

Nous avons évalué notre méthode sur 20 images de courrier entrants. Les performances de la méthode sont récapitulées dans le tableau 3.1. Une ligne est considérée comme «bien segmentée» si et seulement si l'ensemble des composantes connexes qui la constituent sont effectivement regroupées au sein d'un même alignement.

nombre de lignes	333	%
bien segmentées	262	79%
sur-segmentées	29	9%
sous-segmentées	42	12 %

TAB. 3.1 – Performance de la méthode de segmentation en lignes

On remarque que près de 80 % des lignes sont parfaitement segmentées, alors que 9% et 12% des lignes sont respectivement sur et sous-segmentées. Les sous-segmentations concernent essentiellement les cas où des composantes connexes appartiennent à plus d'une ligne (voir figure 3.8). Ce cas de figure n'est en effet pas pris en compte par notre méthode. Il faudrait pour y remédier mettre en œuvre une méthode de segmentation dédiée.

La méthode ne produit donc pas une segmentation en ligne idéale. L'évaluation sur les 20 courriers montre toutefois qu'aucun des champs numériques qui s'y trouvait

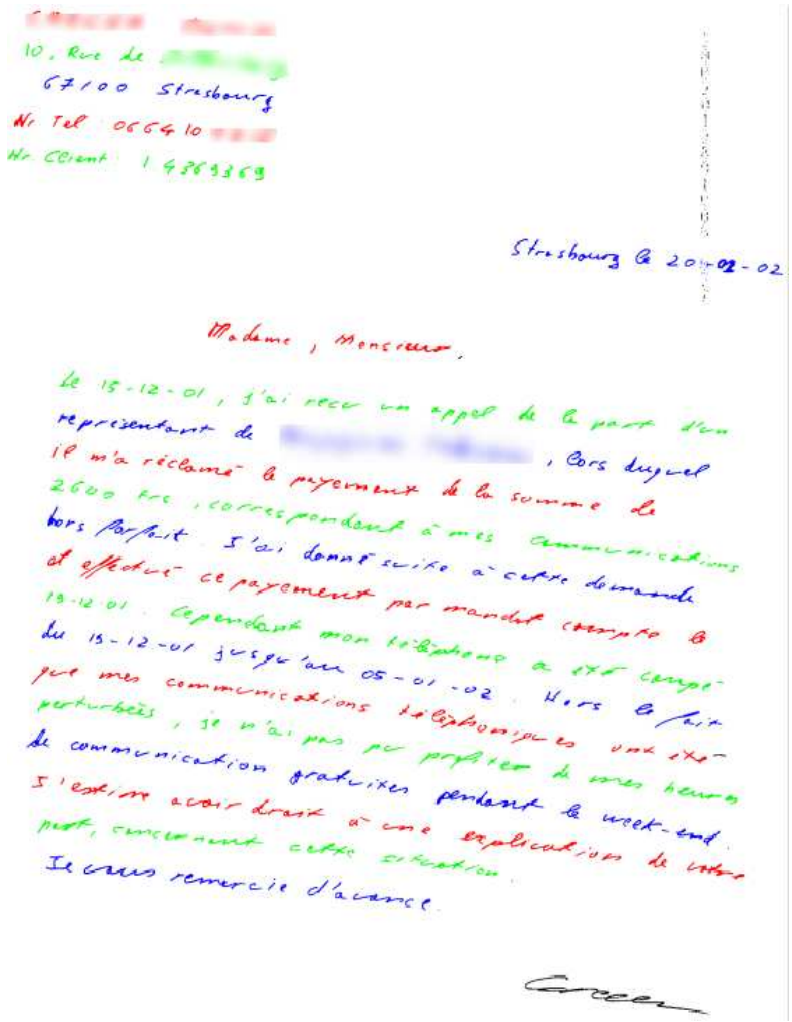


FIG. 3.7 – Résultat de la segmentation en lignes sur une image.

FIG. 3.8 – Exemples de sous-segmentation dus à une composante appartenant à deux lignes.

n'a été sous-segmenté. Pour chaque champ, l'ensemble des composantes connexes qui le constitue forme une sous-chaîne d'une ligne, et la méthode semble ainsi suffisante

pour le problème d'extraction des champs numériques. Afin de confirmer ce résultat, nous avons appliqué la segmentation en lignes sur une base plus conséquente de 293 courriers. L'évaluation montre que la segmentation en ligne «casse» la séquence de composantes d'un champ dans seulement 2% des cas. Ces erreurs sont principalement dûes à l'intrusion dans l'alignement du champ de composantes extérieures, ou d'une connexion entre une composante du champ et une composante d'une autre ligne (voir figure 3.9).

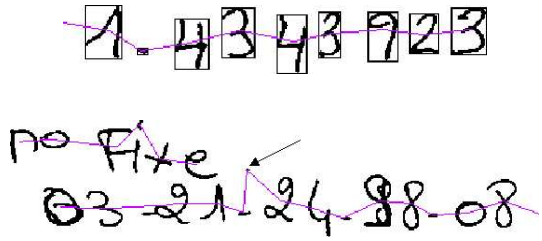


FIG. 3.9 – Exemples de champs bien et mal alignés.

Au vu des résultats, notre méthode de localisation des lignes de texte nous a paru suffisante pour l'application de localisation et de reconnaissance de champs numériques.

3.3 Une méthode de segmentation-reconnaissance descendante

Une stratégie de segmentation-reconnaissance repose sur la génération d'hypothèses de segmentation qui sont ensuite évaluées par un moteur de reconnaissance afin d'en déterminer les plus vraisemblables à l'aide des scores de confiance du classifieur. Nous avons vu dans la section 3.1.2 qu'une stratégie de segmentation-reconnaissance chiffre pouvait être appliquée sur l'ensemble des composantes, quelle que soit leur nature (numérique ou textuelle). Toutes les composantes sont ainsi reconnues en tant que composante numérique : chiffre ou chiffres liés. Les formes non numériques seront identifiées par la suite à l'aide de la méthode de discrimination chiffre-rejet décrite dans la section 3.5. Le comportement de la statégie de segmentation-reconnaissance sur les composantes rejets est donc volontairement occulté dans cette section. On se focalise ainsi sur la conception d'une stratégie produisant une segmentation et une reconnaissance efficaces des composantes numériques lorsqu'elles se présentent.

Nous proposons une méthode de segmentation-reconnaissance à l'échelle de la composante connexe afin de déterminer successivement la meilleure hypothèse de reconnaissance pour les composantes numériques susceptibles d'être rencontrées : chiffre isolé, double chiffre, triple chiffre, etc. Il nous faut donc mettre en place une

méthode de segmentation adaptée aux chiffres liés ainsi qu'un classifieur chiffre. Nous décrivons maintenant ces deux opérations.

3.3.1 Segmentation des composantes

En ce qui concerne les méthodes de segmentation de caractères manuscrits, elles varient suivant le type de caractères traités : mots ou chiffres. Nous renvoyons à [Casey 96] pour un état de l'art sur les méthodes de segmentation de caractères. D'une manière générale, les méthodes de segmentation-reconnaissance utilisées pour la segmentation des mots et des séquences numériques peuvent être qualifiées d'*ascendantes* dans la mesure où les composantes connexes sont systématiquement sur-segmentées en graphèmes, puis tous les regroupements de graphèmes, généralement organisés en niveaux, sont soumis au classifieur qui détermine l'hypothèse de segmentation la plus probable en fonction des scores de reconnaissance du classifieur. On retrouve ce type d'approche dans [Liu 06, Lei 04, Lethelier 95] où elle est appliquée avec succès à la reconnaissance de séquences numériques. Dans la mesure où une sur-segmentation est souvent effectuée afin de ne pas manquer les bons points de coupure, ce type d'approche a l'inconvénient de générer un nombre conséquent de regroupements, particulièrement lorsque le nombre de niveaux est élevé. Partant de l'hypothèse que les composantes «chiffres liés» ne comportent généralement pas plus de quelques chiffres[1], nous avons privilégié une méthode de segmentation-reconnaissance *descendante*, c'est-à-dire partant de la composante connexe complète pour arriver aux chiffres.

Les méthodes de segmentation adaptées aux chiffres génèrent le plus souvent un chemin plutôt qu'un point de coupure. Cela est dû aux connections parfois multiples ou prolongées entre les chiffres liés (voir figure 3.10).

FIG. 3.10 – Exemples de chiffres liés montrant la diversité des types de liaisons.

Nous avons utilisé une méthode de segmentation inspirée de l'algorithme «drop fall» [Congedo 95, Dey 99], qui consiste à segmenter la composante sur le chemin emprunté par une goutte d'eau qui coulerait selon les contours de la composante. Lorsque la goutte est bloquée au fond d'une vallée, celle-ci coupe la composante et continue sa chute. Cet algorithme permet de générer quatre chemins de coupures, suivant que la goutte descende ou qu'elle monte, et suivant la direction prioritaire (gauche ou droite) qu'on lui impose lorsqu'elle rencontre un *extrema* (mont ou vallée). Ces quatre variantes fournissent généralement des chemins différents contenant au moins une bonne segmentation (voir figure 3.11).

[1]Sur notre base de documents, le nombre maximum de chiffres liés dans une composante est 3.

FIG. 3.11 – Les 4 variantes de l'algorithme «drop fall» : a) ascendant gauche b) ascendant droit c) descendant gauche d) descendant droit.

Un paramètre déterminant de cet algorithme est le point de départ de la chute de la goutte. Dans [Congedo 95], les auteurs proposent de parcourir l'image de gauche à droite et de haut en bas, en cherchant le premier pixel blanc qui remplit ces deux conditions (voir figure 3.12 a) :

– le voisin gauche de ce pixel est noir ;
– il existe un pixel noir à droite de ce pixel.

Cependant, cette initialisation ne convient pas toujours puisqu'il est possible de tomber dans un minimum local qui ne correspond que très rarement à un espace inter-chiffre (voir figure 3.12 b).

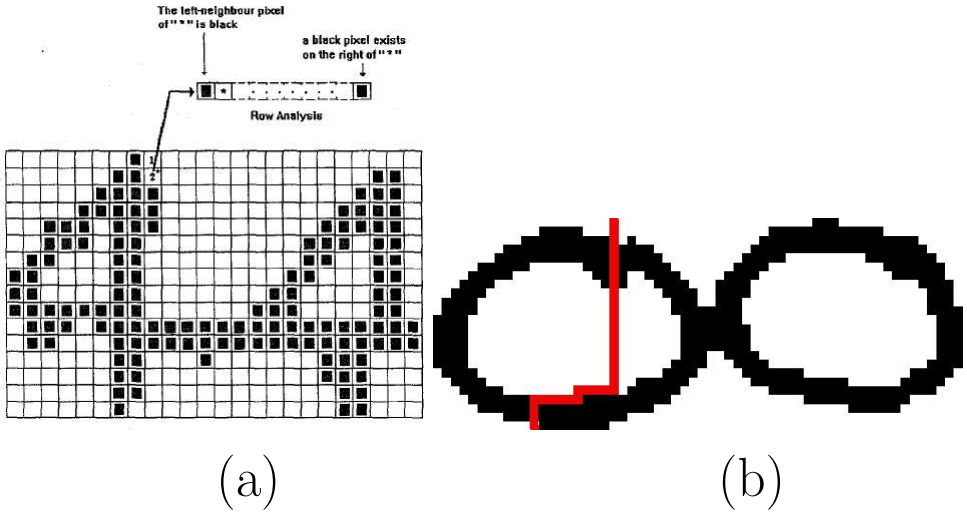


FIG. 3.12 – a) Point de départ pour l'algorithme drop fall par Congedo [Congedo 95]. b) Échec de la segmentation avec une telle initialisation pour un drop fall descendant.

Nous avons donc développé une méthode permettant de trouver le meilleur point de départ pour la goutte en évitant les *minima* locaux. Cette méthode est basée sur la recherche des «water reservoir» de la composante. Un «water reservoir» est une métaphore pour illustrer les vallées d'une composante [Pal 03] (voir figure 3.13). Ils sont obtenus en considérant les zones inondées après un lacher d'eau sur la composante, depuis le haut ou le bas de celle-ci. Si l'eau est versée depuis le haut de la

composante, il s'agit d'un réservoir «haut» ; si l'eau est versée depuis le bas, il s'agit d'un réservoir «bas».

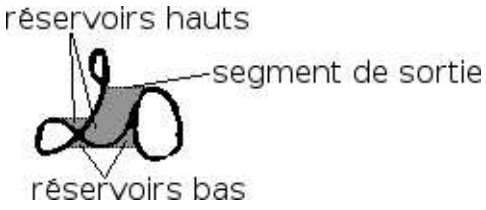


FIG. 3.13 – «Water reservoirs» haut et bas d'une composante. Le segment de sortie correspond à la surface extérieure du réservoir.

Une fois les réservoirs extraits de la composante, nous choisissons le plus grand réservoir haut, et le plus grand réservoir bas. L'abscisse initiale pour la chute de la goutte est déterminée à partir des «segments de sortie» de ces deux réservoirs (voir figure 3.13) : l'abscisse du milieu du segment de sortie du plus grand réservoir bas est choisie comme point de départ du drop fall ascendant ; l'abscisse du milieu du segment du plus grand réservoir haut est choisie pour le drop fall descendant. La goutte est donc assurée de tomber dans un grand réservoir, évitant ainsi les *minima* locaux (voir figure 3.14).

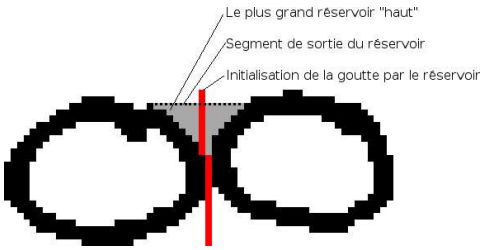


FIG. 3.14 – Segmentation «drop fall» descendant avec initialisation par les réservoirs.

3.3.2 Sélection des chemins de coupures

Rappelons que chaque composante est successivement reconnue comme un chiffre isolé, un double chiffre, un triple chiffre, etc. Comme les composantes possédant plus de trois chiffres liés sont extrèmement rares, nous nous limitons aux triples chiffres. Si la production de l'hypothèse de reconnaissance «chiffre isolé» est immédiate par le biais du classifieur chiffre, il nous faut trouver le meilleur chemin de segmentation pour les doubles chiffres, et les deux meilleurs pour les triples chiffres.

Reconnaissance des composantes en tant que double chiffre : le choix du meilleur chemin de segmentation parmi les quatre variantes du drop fall est

déterminé suivant les principes d'une stratégie de segmentation-reconnaissance, c'est-à-dire en faisant l'hypothèse qu'une bonne segmentation produit des scores de confiance élevés. Dans le cas contraire, les hypothèses de classification chiffre devraient voir leur score chuter. Nous choisissons donc comme critère le produit des confiances des deux premières propositions du classifieur chiffre. La figure 3.15 présente la segmentation et la reconnaissance d'une composante «double digit» selon les quatre variantes du «drop fall» ; ici le drop fall ascendant gauche maximise le produit des confiances, cette hypothèse est donc conservée.

Drop fall	ascendant gauche	ascendant droit	descendant gauche	descendant droit
chemin de coupure				
classifieur chiffres	0[98] 8[82]	2[27] 8[35]	0[73] 8[36]	0[92] 8[34]
produit des confiances	81	09	26	32

FIG. 3.15 – Exemple de segmentation d'un chiffre lié selon les quatre variantes du drop fall, et reconnaissance par le classifieur chiffre. Le chemin de coupure généré par le drop fall ascendant gauche produit des confiances maximum ; nous conservons donc cette hypothèse de segmentation.

Reconnaissance des composantes en tant que triple chiffre : il s'agit ici de déterminer les deux chemins de coupures séparant les trois chiffres. Or, étant donnée l'initialisation de l'algorithme drop fall par la méthode des *water reservoir*, les quatre variantes ont tendance à segmenter la composante selon des chemins voisins. Il ne fournissent donc généralement qu'un seul des deux bons chemins. Nous proposons donc de réitérer l'algorithme du drop fall sur la moitié de composante la «moins bien reconnue», c'est-à-dire sur l'hypothèse de segmentation produisant le plus faible score de confiance. Signalons que ce processus itératif peut être répété davantage afin de prendre en compte les éventuels chiffres liés contenant plus de trois chiffres.

En appliquant successivement les méthodes de reconnaissance de chiffre isolé puis de chiffres liés, on obtient le treillis escompté. La figure 3.16 donne des exemples du résultat de la stratégie de segmentation-reconnaissance descendante appliquée sur des composantes textuelles, doubles et triples chiffres. On remarque que les bonnes hypothèses de segmentation produisent les scores de confiance les plus élevés.

3.4 Classifieur chiffre

Ce chapitre décrit la conception du classifieur chiffre utilisé dans la stratégie de segmentation-reconnaissance. De nombreux extracteurs de caractéristiques [Trier 96] et classifieurs [Jain 00] ont été utilisés pour la reconnaissance de caractères ma-

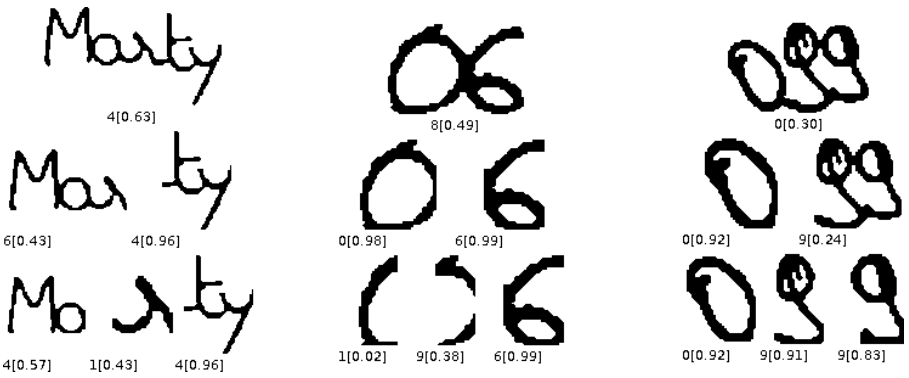


FIG. 3.16 – Résultat de la stratégie de segmentation-reconnaissance descendante appliquée sur (i) une composante textuelle (ii) un double chiffre (iii) un triple chiffre.

nuscrits. Cependant, il n'existe aucun résultat théorique prouvant la supériorité d'un extracteur de caractéristiques ou d'un type de classifieur par rapport à un autre. Partant de cette hypothèse, il est intéressant d'exploiter la complémentarité entre plusieurs classifieurs. Nous avons donc choisi d'utiliser plusieurs vecteurs de caractéristiques soumis à des classifieurs dont les sorties sont combinées. Nous décrivons maitenant les classifieurs et vecteurs de caractéristiques utilisés, la combinaison de classifieurs, et donnons les performances du système de reconnaissance de chiffres.

3.4.1 Choix du classifieur

Les systèmes de reconnaissance d'écriture actuels utilisent différents classifieurs tels que les machines à vecteur de support (SVM) [Vapnik 95, Oliveira 03], les réseaux de neurones (MLP, RBF, etc) [Bishop 95, LeCun 89, Liu 04] ou des combinaisons de plusieurs types [Bellili 01]. Tous ces classifieurs disposent toutefois de caractéristiques différentes en termes de rapidité, de généralisation ou de capacité de modélisation en grande dimension, et le choix du classifieur n'est donc pas trivial. Il doit être guidé par les contraintes imposées par la chaîne de traitement. Dans notre cas, les contraintes sont les suivantes :

- La première contrainte est une contrainte de performance en classification. Même si le classifieur doit traiter une majorité de composantes textuelles, celui-ci doit fournir la bonne hypothèse de classification lorsqu'un chiffre lui est présenté. Ce critère impose l'utilisation d'un nombre de caractéristiques relativement important afin de décrire toute la variabilité des chiffres manuscrits. Cela impose donc une contrainte supplémentaire au classifieur chiffre puisqu'il doit être capable de supporter les hautes dimensions.
- La deuxième contrainte concerne la rapidité en phase de décision. En effet, le classifieur est appliqué sur des documents entiers contenant fréquement plu-

sieurs centaines de composantes connexes, auxquelles il faut rajouter toutes les hypothèses de segmentation. Au total, le classifieur est ainsi lancé plus d'un millier de fois pour une page d'écriture normale, ce qui exclut dès maintenant l'utilisation de classifieur de type «plus proches voisins».

– Comme le classifieur intervient dans une stratégie de segmentation-reconnaissance, il doit être capable de générer des scores de confiance les plus fiables possibles, c'est-à-dire qu'il doit fournir un score de confiance élevé lorsqu'un chiffre bien formé lui est présenté, et un score de confiance faible sinon (forme non numérique, chiffre mal segmenté, double chiffre). Le classifieur doit donc posséder de bonnes capacités de rejet.

Le classifieur «parfait» n'existant pas, il s'agit de trouver le classifieur apportant le meilleur compromis possible.

Concernant la contrainte de performance en classification, les classifieurs «discriminants» (MLP, SVM) donnent généralement de meilleurs résultats que les classifieurs «modélisants» (fenêtres de Parzen, RBF) puisqu'ils permettent de constuire des frontières de décision plus précises entre les classes [Milgram 04, Liu 02b]. Le nombre important de caractéristiques joue également en défaveur des classifieurs modélisants, ainsi que des SVM. En effet, si en théorie les SVM peuvent supporter un espace de caractéristiques infini [Vapnik 95], dans la pratique les performances s'écroulent lorsque la dimension augmente. C'est la raison pour laquelle des méthodes de sélection de caractéristiques sont couramment employées [Guyon 02, J.Weston 00].

La contrainte de rapidité privilégie les classifieurs neuronaux, extrèmement rapides en phase de décision. La vitesse des SVM en phase de décision dépend du nombre de vecteurs support conservés à l'issue de la phase d'apprentissage et donc de la complexité du problème, mais on peut affirmer qu'ils sont d'une manière générale plus lents, en particulier en haute dimension.

Enfin la contrainte relative à la capacité de rejet aurait tendance à nous orienter vers les classifieurs modélisants qui possèdent une aptitude naturelle pour le rejet de distance [Milgram 04, Mouchère 06]. Cependant, le problème de la dimensionalité interdit une nouvelle fois la modélisation des classes et l'emploi de classifieurs modélisants.

Compte tenu des contraintes imposées par le système, les réseaux de neurones discriminants de type MLP semblent les plus adaptés à notre problème puisqu'ils possédent un excellent compromis rapidité/performances en classification, et sont peu sensibles aux problèmes de dimensionalité (sous réserve de disposer de suffisamment d'exemples). Il reste le problème de la gestion des scores de confiance et de la capacité de rejet de tels classifieurs. Nous discutons de ce point dans la section 3.5.

3.4.2 Extraction de caractéristiques

Parmi les nombreuses méthodes d'extraction de caractéristiques disponibles dans la littérature (voir [Trier 96] et le chapitre 1), nous avons retenu deux vecteurs de

caractéristiques :

– Le vecteur de caractéristiques du *chaincode* extrait du contour des composantes a montré son efficacité dans de nombreux problèmes de reconnaissance [Kimura 94]. Après avoir effectué un pavage de l'imagette, l'histogramme des directions de Freeman des pixels est extrait dans chaque zone de l'image. Les histogrammes constituent les caractéristiques du vecteur (voir figure 3.17). Nous considérons un voisinage 8-connexe et un pavage $4 * 4$, ce qui fournit un vecteur à 128 caractéristiques.

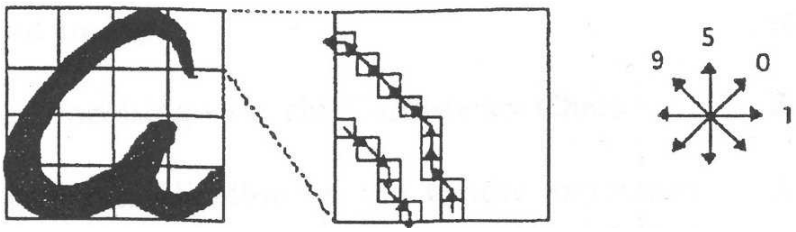


FIG. 3.17 – Image pavée, extraction du coutour et histogramme des directions du contour en 8-connexité sur un des pavé (l'image provient de [Kimura 94]).

– Nous utilisons également le vecteur statistique/structurel développé dans nos travaux antérieurs [Heutte 98]. Ce deuxième vecteur est constitué de 117 caractéristiques réparties en 6 familles (projections, profils, intersections, fin de traits et jonctions, concavités, et *extrema*), et a prouvé son efficacité dans la discrimination de caractères manuscrits tels que les chiffres, lettres majuscules et même graphèmes [Heutte 98].

3.4.3 Entraînement et combinaison des classifieurs

Entraînement des classifieurs

Nous avons donc conçu un MLP pour chaque vecteur de caractéristiques. Appelons «MLP128» et «MLP117» les classifieurs MLP entraînés respectivement sur le vecteur chaincode et le vecteur statistique/structurel. Ils sont tous les deux construits sur le même schéma :

– Une couche d'entrée contenant autant de neurones que de caractéristiques : 128 et 117.
– Une couche cachée dont le nombre de neurones a été fixé à *(nombre d'entrées + nombre de sorties) / 2* pour les deux réseaux.
– Une couche de sortie composée d'autant de neurones que de classes, soit 10.
– La fonction d'activation utilisée est une sigmoïde.

Les deux MLP sont entraînés avec l'algorithme itératif de rétropropagation du gradient (voir section 1.2.3.3), avec un pas adaptatif de type «line search» [Bishop 95] qui minimise l'erreur à chaque itération. La base d'apprentissage contient 114461 chiffres étiquetés provenant de formulaires.

Le tableau 3.2 donne les taux de reconnaissance en rang 1, 2 et 3 sans rejet des MLP117 et MLP128 sur une base de test de 38154 chiffres manuscrits.

taux de reconnaissance	RANG1	RANG2	RANG3
MLP128	97,03	98,98	99,50
MLP117	97,93	99,37	99,73

TAB. 3.2 – Taux de reconnaissance en première, deuxième et troisième proposition des classifieurs MLP117 et MLP128.

Comme nous pouvons le constater, ces deux classifieurs ont des performances intéressantes. Il semble donc naturel de les combiner afin de tirer le meilleur parti de leurs spécificités.

Combinaison de classifieurs

De nombreux travaux ont montré qu'une combinaison de classifieurs pouvait améliorer la robustesse d'une classification en prenant en compte la complémentarité entre les classifieurs [Zouari 02, Rahman 03]. Il existe plusieurs méthodes de combinaison de classifieurs, applicables en fonction de la nature de l'information à combiner [Xu 92] : les méthodes de type classe utilisent la meilleure solution de chaque classifieur, les méthodes de type rang utilisent les listes ordonnées de propositions des classifieurs, enfin les méthodes de type mesure utilisent la valeur de confiance associée à chaque proposition de la liste. Ce dernier type de combinaison fournit une mesure de confiance qui est l'information dont nous avons besoin pour l'analyseur syntaxique. Nous utiliserons donc le type mesure pour la combinaison de nos classifieurs.

Plusieurs règles de combinaison peuvent être utilisées pour fournir la sortie de la combinaison [Rahman 03] : le maximum, le minimum, la médiane, le produit, la combinaison linéaire sont les plus couramment utilisés. Nous avons essayé les méthodes de fusion produit, moyenne arithmétique et maximum (resp. «prod», «mean» et «max»).

Évaluation du système de classification

Le tableau 4.5 donne les taux de reconnaissance en rang 1, 2 et 3 sur la base de test.

taux de reconnaissance	RANG1	RANG2	RANG3
max (MLP117,MLP128)	98,56	99,60	99,85
prod(MLP117,MLP128)	98,64	99,60	99,84
mean(MLP117,MLP128)	98,72	99,63	99,85

TAB. 3.3 – Taux de reconnaissance sans rejet en première, deuxième et troisième proposition des combinaisons de classifieurs.

On remarque que les taux de reconnaissance sans rejet sont systématiquement supérieurs à ceux obtenus par les classifieurs seuls, et que la règle de combinaison de type moyenne arithmétique donne sensiblement les meilleurs résultats. Nous conservons donc par la suite ce type de combinaison.

On obtient ainsi un taux de reconnaissance sans rejet de 98,72% en première proposition, ce qui nous situe au niveau de l'état de l'art des classifieurs de la littérature [Liu 02a].

En rang 1, la matrice de confusion indique les erreurs les plus fréquentes (voir figure 3.18). On remarque les confusions «classiques» entre chiffres possédant des formes proches : 7/1 ; 8/2 ; 5/9 ; 3/9.

Matrice de confusion en TOP1

	0	1	2	3	4	5	6	7	8	9
0	0.996	0.000	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.002
1	0.001	0.987	0.001	0.001	0.002	0.000	0.000	0.005	0.001	0.003
2	0.001	0.001	0.980	0.000	0.002	0.000	0.001	0.004	0.005	0.005
3	0.000	0.002	0.000	0.990	0.000	0.000	0.000	0.001	0.002	0.004
4	0.001	0.004	0.002	0.000	0.984	0.000	0.002	0.002	0.001	0.004
5	0.000	0.001	0.001	0.003	0.001	0.983	0.004	0.001	0.003	0.006
6	0.001	0.000	0.000	0.000	0.001	0.002	0.990	0.000	0.005	0.000
7	0.000	0.003	0.001	0.000	0.002	0.000	0.000	0.990	0.001	0.005
8	0.001	0.000	0.002	0.001	0.000	0.000	0.000	0.002	0.991	0.002
9	0.001	0.002	0.001	0.005	0.002	0.002	0.000	0.002	0.004	0.982

FIG. 3.18 – Matrice de confusion du classifieur chiffre.

En ce qui concerne la rapidité du classifieur en phase de décision, il faut environ 43 secondes pour extraire les deux vecteurs de caractéristiques, exécuter les deux MLP et les combiner pour les 38154 chiffres de la base de test, sur une machine cadencée à 1,5GHz. Si l'on admet que le classifieur doit être lancé 1000 fois sur une page d'écriture manuscrite, on obtient un temps de traitement tout à fait raisonnable puisqu'il est légèrement supérieur à 1 seconde par image.

Afin d'estimer ses capacités de rejet, nous donnons la courbe ROC du classifieur. La courbe ROC (Receiver Operating Curve) [Bradley 97] présente sur un même graphique les taux de fausse acceptation et de faux rejet obtenus par un classifieur sur une base contenant des exemples positifs et négatifs pour une règle de rejet donnée. La règle de rejet la plus simple introduite par Chow [Chow 70] consiste à accepter une forme si la sortie du classifieur est supérieure à un seuil, et à la rejeter sinon. En faisant varier ce seuil d'acceptation, on obtient plusieurs compromis fausse acceptation/faux rejet qui constituent la courbe ROC. Pour le classifieur chiffre, la courbe ROC présente les taux de fausse acceptation (un rejet accepté) et de faux rejet (un chiffre rejeté) obtenus pour différents seuils d'acceptation sur une base de 14733 formes contenant 1/3 de chiffres et 2/3 de non chiffres (voir figure 3.19). On considère généralement que le classifieur dont l'aire sous la courbe ROC est la plus faible possède les meilleurs capacités de rejet.

Le tableau 3.4 présente quelques points de fonctionnement de la courbe ROC

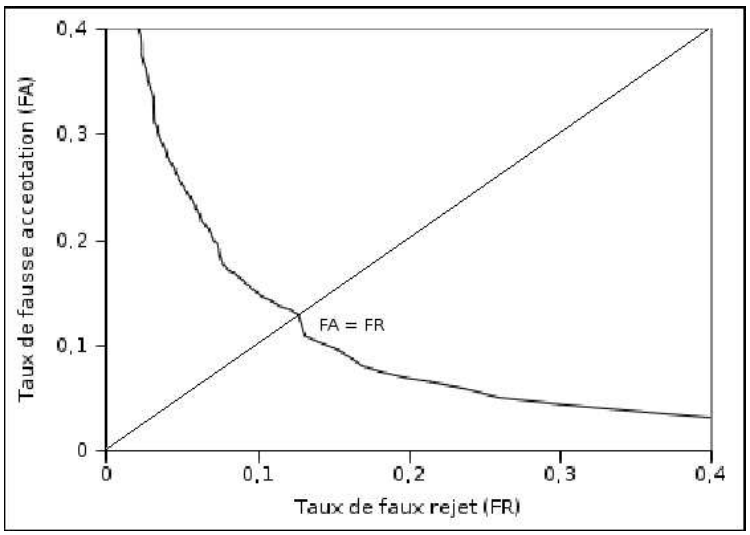


FIG. 3.19 – Courbe ROC du moteur chiffre obtenue sur une base de 14733 chiffres et rejets.

du classifieur chiffre. Il est par exemple possible d'obtenir un faux rejet de 5% pour une fausse acceptation de 25% avec un seuil = 0.49 appliqué sur la confiance de la première proposition du classifieur chiffre. L'équilibre FA = FR = 12,88% est obtenu pour un seuil de 0.83.

Seuil d'acceptation	0,12	0,49	0,78	0,83	0.91	0,95	0,97
Faux rejet (%)	1,00	5,00	10,00	**12,88**	20,00	30,00	45,00
Fausse acceptation (%)	49,19	25,15	14,83	**12,88**	6,84	4,44	2,51

TAB. 3.4 – Quelques compromis fausse acceptation/ faux rejet obtenus par le classifieur chiffre.

3.5 Rejet des composantes non chiffres

Dans cette section, nous décrivons le module chargé de distinguer les chiffres de toutes les autres formes : mots, fragments de mots, lettres, bruit, chiffres liés, chiffre mal formé, etc. Comme nous l'avons souligné dans la section 3.1.1, le rejet des formes ne peut se faire par une décision binaire (acceptation/rejet) puisque les mauvaises décisions et en particulier les rejets de chiffres auraient des conséquences difficile à rattraper par le système. Nous avons donc choisi d'ajouter aux hypothèses de classification chiffre une étiquette «Rejet» assortie d'un score d'appartenance à cette classe. Le problème qui se pose dans cette partie est donc la production de ce score.

Pour cela, nous proposons de nous baser sur l'analyse des sorties du classifieur chiffre développé dans la section 3.4 afin d'exploiter les capacités de rejet que nous venons de présenter. Le système repose sur le rejet d'ambiguïté du classifieur chiffre (combinaison de MLP) lorsque les formes à classer sont proches des frontières de décision. On estime donc un score de confiance «Rejet» à partir des scores de confiance de la première proposition du classifieur chiffre. Pour cela, nous avons généré une table de correspondance (ou LUT pour Look Up Table) en analysant le comportement du classifieur chiffre sur une base de 4500 éléments contenant des chiffres et des rejets. Les statistiques sur le score de la première proposition ont permis de donner la LUT présentée en figure 3.20.

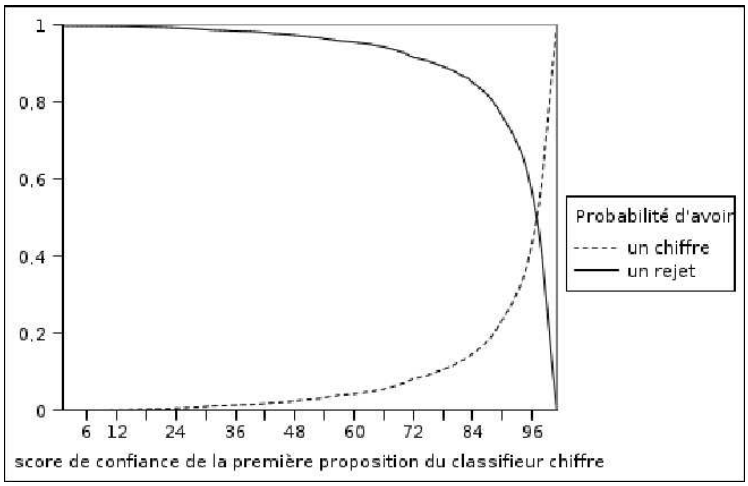


FIG. 3.20 – LUT fournissant le score de la classe rejet à partir du score de la première proposition du classifieur chiffre.

À l'issue de cette estimation, le système fournit 11 scores de confiance (10 scores du MLP + score rejet), sur lesquels nous appliquons la fonction *softmax* [Bridle 90] pour donner des estimations de probabilités *a posteriori*.

3.6 Filtrage des séquences valides

La dernière étape de la chaîne de traitement consiste à faire remonter les solutions susceptibles d'être rencontrées dans des documents réels parmi les hypothèses de segmentation/reconnaissance/rejet générées. On cherche donc dans le treillis de segmentation/reconnaissance (voir figure 3.21) les meilleures solutions valides au sens d'un certain nombre de contraintes qu'il nous faut expliciter. Comme l'analyse se fait sur les lignes de texte, des modèles de ligne de texte doivent être définis pour tous les types de champs qui nous intéressent.

La recherche des meilleurs chemins sous les contraintes des modèles présentés

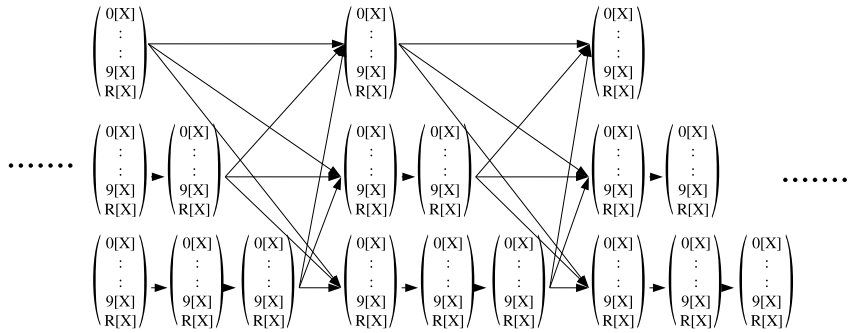


FIG. 3.21 – Treillis de segmentation-reconnaissance-rejet pour une ligne de texte.

ci-après est effectuée classiquement à l'aide de l'algorithme forward [Rabiner 90].

3.6.1 Définition des modèles

L'analyse des champs numériques rencontrés dans des documents manuscrits de type courriers entrants nous a permis d'identifier les règles syntaxiques régissant ces champs. Par exemple, un code postal est constitué de cinq chiffres, le modèle de ligne associé à ce type de champ est donc constitué de plusieurs composantes rejet suivies des cinq chiffres puis à nouveau de composantes rejet. La figure 3.22 présente les modèles de lignes pour a) un code postal, b) un code client, c) un numéro de téléphone.

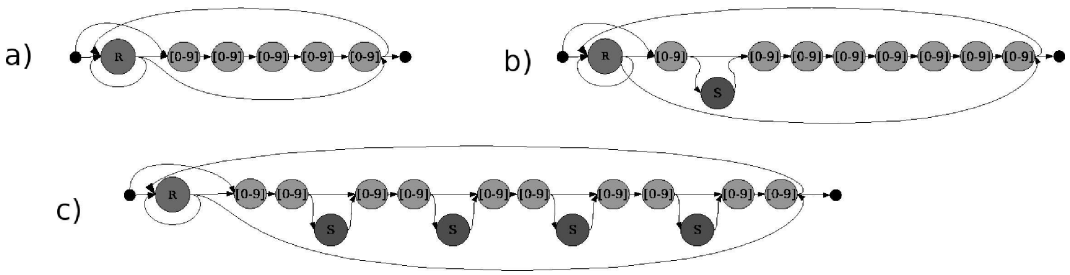


FIG. 3.22 – Modèles d'une ligne de texte contenant a) un code postal, b) un code client, c) un numéro de téléphone

Comme nous pouvons le constater sur la figure 3.22, nous avons choisi d'intégrer dans les modèles des états «séparateur». Ces états présents sur les modèles *numéro de téléphone* et *code client* permettent d'absorber les composantes de type point ou tiret souvent présents pour séparer des groupes de chiffres (par exemple entre les paires de chiffres d'un numéro de téléphone). L'ajout de cet état dans les modèles

implique donc une étape de reconnaissance supplémentaire pour les identifier que nous présentons maintenant.

3.6.2 Reconnaissance des séparateurs

Comme pour l'ajout des hypothèses de reconnaissance «rejet», l'ajout des hypothèses de reconnaisance «séparateur» se fait par la production d'un score de confiance pour cette classe à l'issue de la reconnaissance chiffre. Nous décrivons dans cette section la méthode utilisée pour reconnaître ces formes.

Contrairement aux chiffres et aux composantes non numériques, les formes «séparateurs» sont assez facilement identifiables puisqu'il s'agit essentiellement de points ou de tirets dont la position est toujours basse relativement aux autres composantes. Un vecteur de 9 caractéristiques contextuelles basées sur les boîtes englobantes des composantes a donc été développé. Soient C la composante considérée, C_{-1} et C_{+1} ses voisines gauche et droite. Soient H_C, W_C, G_{C_x} et G_{C_y} respectivement les hauteur, largeur, l'abscisse et l'ordonnée du centre de gravité de la composante C. Les 9 caractéristiques permettant d'identifier la régularité/irrégularité dans la taille et le positionnement des composantes sont :

$$f_1 = \frac{H_{C_{-1}}}{H_C}; \qquad f_2 = \frac{H_{C_{+1}}}{H_C}; \qquad f_3 = \frac{W_{C_{-1}}}{W_C}; \qquad f_4 = \frac{W_{C_{+1}}}{W_C}$$

$$f_5 = \frac{H_C}{W_C}; \qquad f_6 = \frac{G_{C_X} - G_{C_{X-1}}}{W_C}; \qquad f_7 = \frac{G_{C_X} - G_{C_{X+1}}}{W_C}$$

$$f_8 = \frac{G_{C_Y} - G_{C_{Y-1}}}{H_C}; \qquad f_9 = \frac{G_{C_Y} - G_{C_{Y+1}}}{H_C}$$

Ce vecteur de caractéristiques est soumis à un classifieur MLP entrainé sur une base de séparateurs et de non séparateurs (chiffres et fragments de chiffres, composantes textuelles, bruit, lettres isolées, etc.). Le taux de reconnaissance sans rejet est de 96%. Une observation montre que les séparateurs sont très rarement liés aux autres composantes ; nous avons donc choisi de ne rechercher les séparateurs que sur le premier niveau du treillis, où les composantes ne sont pas segmentées. À l'issue de cette reconnaissance, le treillis comporte les hypothèses de reconnaissance de chiffres, rejets et séparateurs pour le premier niveau, et des hypothèses de chiffres et rejets pour les niveaux 2 et 3 (voir figure 3.23).

3.7 Résultats

Nous présentons dans cette section les résultats de la méthode basée sur une stratégie de segmentation - reconnaissance - rejet pour la localisation et la reconnaissance de champs numériques. Nous présentons les résultats sur une base de 293 documents contenant des codes postaux, numéros de téléphone et codes clients dont la position et la valeur ont été annotées (voir section 2.2.3). La recherche des

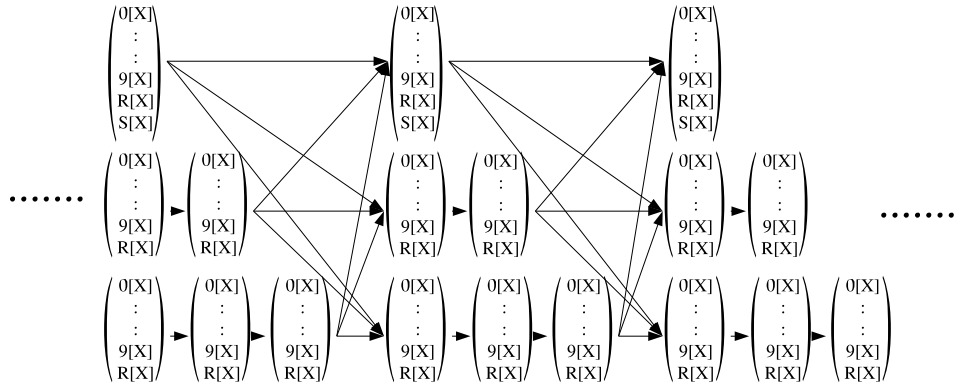


FIG. 3.23 – Treillis complet des hypothèses de reconnaissance chiffre/rejet/séparateurs.

champs est effectuée par la recherche des meilleurs chemins successivement sous les contraintes des trois types de champs.

Un champs est considéré comme bien reconnu si et seulement si :

- la localisation est exacte : toutes les composantes et seulement les composantes appartenant au champ annoté ont été étiquetées en tant que chiffre ou séparateur ;
- la reconnaissance est exacte : tous les chiffres du champ ont été correctement reconnus.

Les différents cas de figure rencontrés à l'issue des traitements sont les suivants (voir tableau 3.5) :

- **a)** Le système produit la bonne localisation et la reconnaissance est juste. Ce champ est considéré comme bien reconnu.
- **b)** Le champ numérique est bien délimité (ou segmenté) mais une des étiquettes chiffre est erronée. Ce champ n'est pas considéré comme bien reconnu et génère une fausse alarme.
- **c)** L'alignement proposé par le système est faux puisqu'un des chiffres est rejeté. Le champ n'est pas considéré comme bien reconnu et génère une fausse alarme.
- **d)** Une séquence numérique est détectée dans une ligne ne contenant pas de champ. Comme pour le cas de figure c), il s'agit d'une «fausse alarme».

Comme nous proposons un système d'extraction d'informations, le critère de performance utilisé est le compromis rappel-précision, ces deux quantités sont définies comme suit :

$$\textbf{rappel} = \frac{\text{nb de champs bien reconnus}}{\text{nb de champs à extraire}}$$

a)	N° de client : 1. 4761154 N° de client : 1. 4761154 RRRRR RR R R RR 1 S 476 1 1 54	79 370 Plaisir 79 370 Plaisir 79 370 RRR RRR	b)
c)	21540 VERRET-SOUS-DRCE 21540 VERRET-SOUS-DRCE R 115 40 R R R R RR RR RRRR	avec AIR Sous avec AIR Sous R RR 1112 1 R R	d)

TAB. 3.5 – Exemples de champs bien/mal reconnus. Chaque exemple présente un extrait de ligne de texte, associé à la segmentation produite par le système ainsi qu'au résultat de la reconnaissance.

$$\textbf{précision} = \frac{\text{nb de champs bien reconnus}}{\text{nb de champs proposés par le système}}$$

L'analyse syntaxique effectuée par l'algorithme forward donne les n meilleurs alignements. Un champ détecté en *TOPn* signifie que la bonne hypothèse de reconnaissance d'un champ est contenue dans les n meilleures propositions de l'analyseur syntaxique.

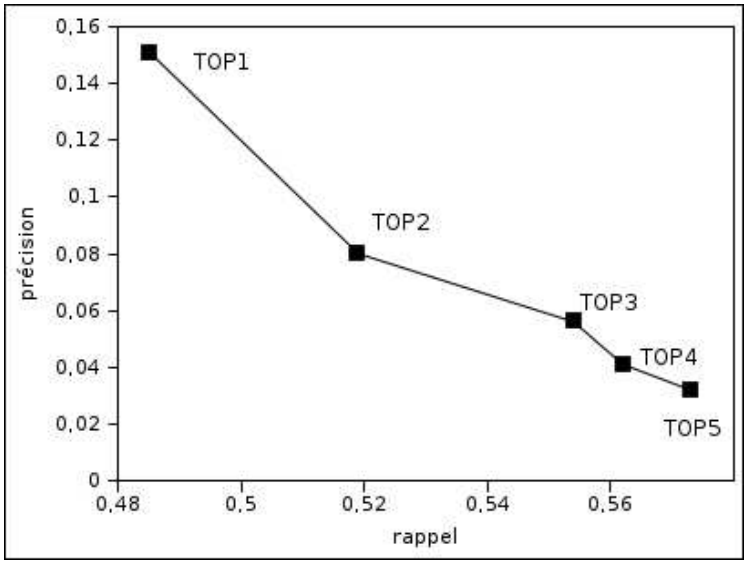


FIG. 3.24 – Rappel-précision du système.

Nous pouvons constater que le système permet d'obtenir un rappel de 48, 5% pour une précision de 15, 5% en ne considérant que la première proposition (TOP1). En considérant les propositions suivantes, on fait augmenter le rappel jusqu'à 57, 6%

mais on diminue la précision du système. Rappelons qu'il s'agit d'un système complet, et que ces résultats finaux rendent compte du cumul des erreurs observées tout au long de la chaîne de traitement : segmentation du document en lignes, segmentation/reconnaissance et rejet des composantes, identification des séparateurs et filtrage des séquences valides.

Au final, sur la base de test contenant 718 champs, le système fournit en TOP1 2247 champs répartis suivant les 4 cas recensés dans le tableau 3.5 de la manière suivante : 15,5% des champs détectés sont des bonnes hypothèses de localisation avec la valeur numérique correcte, (cas **a**) ; 1,1% concerne des hypothèses de localisation correcte avec une fausse hypothèse de reconnaissance chiffre (cas **b**) ; 15,3% sont des hypothèses de localisation mal alignées sur le champ à détecter (cas **c**) ; et enfin 68% des champs proposés sont des réelles fausses alarmes (détection d'un champ dans une ligne ne contenant pas de champ d'intérêt : cas **d**). Concernant ce dernier cas de fausse alarme, il apparaît dans les deux cas suivants :

- Un champ est détecté dans une ligne de texte contenant des chiffres n'appartenant pas au type de champ recherché (voir figure 3.25 haut).
- Un champ est détecté dans une séquence de mots dont la segmentation a fait apparaître des formes proches de chiffres (voir figure 3.25 bas).

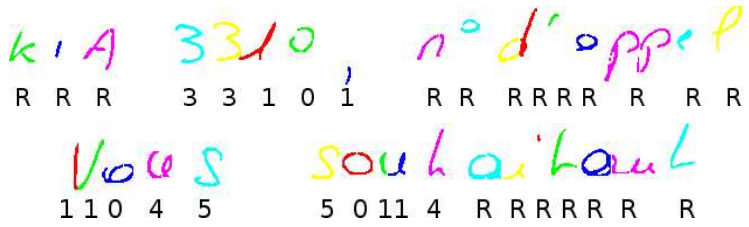


FIG. 3.25 – Deux cas fréquents de fausse alarme. En haut : détection d'un code postal due à la présence de chiffre n'appartenant pas à un champ recherché. En bas : détection d'un numéro de téléphone dans des mots segmentés.

On constate donc que la segmentation systématique des composantes du document engendre un nombre de fausses alarmes important.

Concernant les temps de traitement, l'extraction des 3 types de champs sur les 293 courriers de notre base de test montre qu'il faut environ 4,5 secondes pour traiter une image de document sur une machine cadencée à 1,5GHz, sans optimisation particulière. Ces temps de traitement rendent le système industrialisable.

3.8 Conclusion

Nous avons implémenté dans cette section une première chaîne de traitement générique pour l'extraction des champs numériques dans des courriers manuscrits quelconques. La méthode repose sur la stratégie la plus évidente qui consiste à localiser et reconnaître les chiffres d'un document pour localiser les champs numériques

recherchés grâce aux connaissances *a priori* sur leur syntaxe. Pour cela, plusieurs phases de traitement ont été identifiées et mises en œuvre. Premièrement, une étape de segmentation du document en lignes a été développée, basée sur une méthode d'aggregation de composantes. La méthode n'est pas parfaite mais suffisante dans notre contexte. Deuxièmement, trois phases de traitements (segmentation, reconnaissance chiffre et rejet) doivent ensuite être appliquées sur l'ensemble des composantes des lignes de texte pour conjointement localiser et reconnaître les chiffres. Afin de faire face au paradoxe segmentation/reconnaissance/rejet, nous avons mis en place deux étapes : (i) une méthode de segmentation-reconnaissance descendante efficace, comparable aux méthodes utilisées sur des séquences numériques isolées (ii) une méthode de rejet capable de discriminer les hypothèses de chiffre valides des rejets, fondée sur une analyse des scores de confiance du classifieur chiffre. La troisième et dernière étape de la chaîne de traitement réalise l'extraction des champs par une recherche des meilleures séquences valides dans le treillis des hypothèses de segmentation/reconnaissance/rejet.

Les résultats montrent que le système permet d'extraire une majorité de champs dans des temps de traitement raisonnables, autorisant une industrialisation du système. Soulignons que la méthode est parfaitement générique puisqu'il suffit d'incorporer dans le système la syntaxe d'un champ pour permettre sa localisation, sans aucun réapprentissage.

Les résultats ont également montré une précision assez faible. Dans un contexte industriel, on pourra filtrer les différentes hypothèses de champs à l'aide d'une base client jouant le rôle d'un lexique. Une analyse des causes de fausses alarmes montre qu'elle est due en particulier à la détection de champs dans des groupes de mots écrits en script. Un moyen de limiter ce type de fausse alarme est donc d'améliorer les capacités de rejet du système. Nous discutons de ce point dans le chapitre 5.

Chapitre 4

Une méthode dirigée par la syntaxe pour l'extraction de champs numériques

Dans ce chapitre, nous décrivons une seconde chaîne de traitement pour l'extraction des champs numériques dans les courriers entrants. Cette chaîne de traitement est basée sur la deuxième stratégie retenue dans la section 2.5, qui constitue une alternative à la stratégie la plus évidente consistant à procéder à une reconnaissance systématique des entités en chiffres. Rappelons qu'il s'agit d'une stratégie où la localisation et la reconnaissance des champs numériques sont effectuées de manière disjointe, ce qui permet de reconnaître les champs localisés à l'aide d'une des stratégies classiques de reconnaissance de séquences isolées décrites dans la section 1.5. La méthode de localisation est quant à elle réalisée en exploitant la syntaxe connue des champs recherchés, sans faire appel à la segmentation des composantes ni à un classifieur chiffre. Afin de localiser les champs numériques sans les reconnaître, nous mettons en œuvre une stratégie neuro-markovienne qui repose sur deux points clefs : une classification syntaxique des composantes effectuée à l'aide d'un classifieur neuronal, et une analyse syntaxique des lignes de texte à base de modèles de Markov cachés. Les connaissances syntaxiques relatives au type de champ recherché sont donc exploitées dès le début de la chaîne de traitement, indépendamment de l'étape de reconnaissance chiffre. Cette reconnaissance chiffre étant dissociée de la localisation, elle permet une étape de vérification des hypothèses de champs numériques à la fin de la chaîne de traitement.

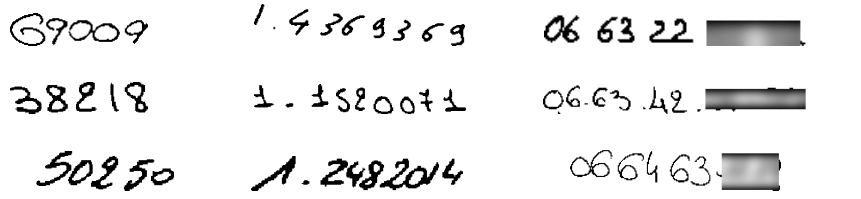
Ce chapitre est organisé de la manière suivante : une description générale de l'approche et de la chaîne de traitement est donnée dans la section 4.1. Nous décrivons dans la section 4.2 la phase de localisation des champs reposant sur une étape de classification des composantes et une analyse syntaxique à base de modèles de Markov. La phase de reconnaissance appliquée sur les champs localisés est décrite dans la section 4.3. Nous présentons les résultats du système à l'issue de cette étape de reconnaissance dans la section 4.3.3, puis nous appliquons une méthode de vérification des

hypothèses de champs dans la section 4.4 afin d'améliorer la précision du système.

4.1 Approche dirigée par la syntaxe

Comme discuté dans la section 2.5.1, la stratégie présentée dans ce chapitre pour l'extraction des champs numériques dans les courriers entrants est composée de deux phases distinctes : une phase de localisation des champs visant à isoler l'information recherchée sans faire intervenir la segmentation ni la reconnaissance chiffre, et une étape de reconnaissance des champs localisés reposant sur des techniques classiques de reconnaissance de séquences numériques, telles que celles mises en œuvre pour la lecture de montants numériques de chèques ou de code postaux par exemple (voir section 1.5).

Notre approche repose sur l'exploitation de la syntaxe connue des champs (nombre de chiffres, présence et position des séparateurs) comme information *a priori* pour parvenir à localiser les champs numériques dans une ligne de texte, sans faire intervenir les étapes de segmentation et de reconnaissance chiffre. Pour cela, nous mettons en œuvre une modélisation neuro-markovienne des lignes de texte. L'idée est d'exploiter la structure syntaxique connue des champs recherchés : on peut voir sur la figure 4.1 que chaque type de champ (code postal, numéro de téléphone, code client) possède une structure syntaxique propre, correspondant à une séquence de chiffres et de séparateurs. Par exemple, les champs numériques de type numéro de téléphone sont constitués de 10 chiffres, généralement regroupés par paires éventuellement séparées par des points ou tirets (séparateurs). Ces champs étant inclus dans une ligne de texte, la modélisation doit prendre en compte l'intégralité de la ligne (champ numérique et composantes non numériques).



codes postaux numéros de clients numéros de téléphone

FIG. 4.1 – Exemples de champs numériques.

Supposons qu'une segmentation du document en lignes ait été effectuée, on dispose alors de la succession des composantes connexes de chaque ligne. La tâche d'extraction d'information va consister à interpréter globalement cette séquence pour associer à chaque composante son étiquette : textuelle ou numérique. Toutefois, puisque l'approche ne procède pas à la segmentation des composantes connexes, une composante numérique peut correspondre à un ou plusieurs chiffres, ou un séparateur

(point, tiret, ...). Le modèle de ligne doit donc être constitué des étiquettes correspondant à ces entités : D (Digit ou chiffre), DD (Double Digits ou chiffres liés), S (Séparateur). Nous avons choisi de ne pas prendre en compte les chiffres liés contenant plus de deux chiffres puisqu'ils sont relativement rares.

Concernant les composantes textuelles, elles sont modélisées par une classe unique appelée classe Rejet pour décrire l'ensemble des composantes non numériques : caractère isolé, fragment de mot, mot, diacritique, signe de ponctuation. Cette classe unique permet d'effectuer une modélisation à la fois fine pour les champs recherchés, et grossière pour les informations non pertinentes.

La figure 4.2 montre l'étiquetage idéal des composantes connexes d'une ligne de texte extraite d'un document.

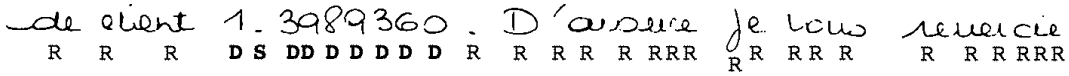


FIG. 4.2 – Exemple d'étiquetage des composantes d'une ligne comprenant un code client. R : Rejet, D : Digit, S : Séparateur, DD : Double Digit.

Les quatre classes ainsi définies peuvent être qualifiées de «syntaxiques» puisqu'elles décrivent la qualité syntaxique des composantes plutôt que leur valeur numérique ou textuelle. En faisant une analogie avec le terme «part of speech» employé dans les méthodes de tagging pour les documents électroniques (voir section 2.4), on peut appeler «part of handwriting» nos composantes syntaxiques. Elles peuvent ainsi constituer les états d'un modèle markovien de ligne de texte qui permet de prendre en compte les connaissances *a priori* sur la syntaxe des champs. Cette syntaxe et les états du modèle étant connus, on peut représenter la structure d'un modèle d'une ligne de texte pouvant contenir un numéro de téléphone par la figure 4.3.

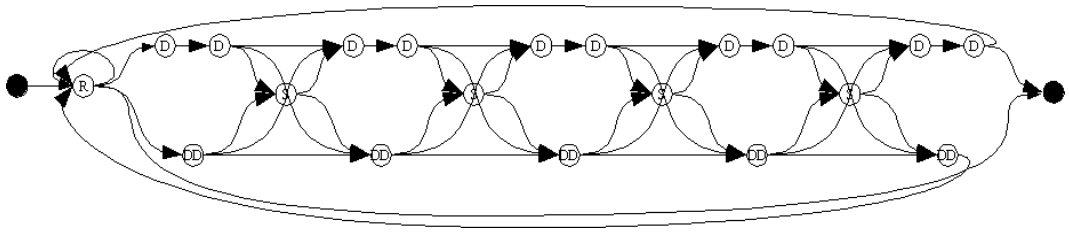


FIG. 4.3 – Modèle de ligne de texte pouvant contenir un champ numérique de type «numéro de téléphone» avec une stratégie sans segmentation.

Si la structure du modèle de ligne est connue, un apprentissage doit être mené afin de déterminer les probabilités de transitions entre les états. Nous discutons de ce point dans la section 4.2.2.

En phase de décision, les séquences de composantes seront ainsi alignées sur les modèles afin de ne conserver que les séquences syntaxiquement correctes. L'extraction d'un champ numérique dans une ligne manuscrite consistera donc à rechercher dans le treillis fourni par le classifieur la meilleure séquence d'étiquettes valide au sens du modèle de Markov utilisé pour modéliser la ligne. La recherche du meilleur alignement dans le treillis est effectuée par l'algorithme de Viterbi [Forney 73, Rabiner 90]. Une fois les champs localisés, un moteur de reconnaissance de champs est chargé de les reconnaître.

Cette stratégie de localisation originale sans segmentation est une alternative à l'utilisation d'une stratégie de segmentation-reconnaissance chiffre sur l'intégralité du document. Comme pour les méthodes d'extraction d'information dans les documents textuels en langue naturelle, une phase de «part of handwriting» tagging enrichit les séquences de composantes d'une information syntaxique qui permet la localisation des champs par une analyse syntaxique. Une fois les champs localisés, une étape de reconnaissance chiffre peut alors être mise en œuvre afin de déterminer leur valeur numérique. Puisque la reconnaissance chiffre n'est appliquée qu'en fin de traitement, elle peut être exploitée pour la vérification des hypothèses de localication et de reconnaissance de champs numériques.

4.1.1 Formalisation du problème

Nous précisons ici le cadre théorique dans lequel se situe notre approche. Nous en dégageons les deux éléments principaux qui seront ensuite discutés dans les sections suivantes : l'utilisation des modèles de Markov cachés et d'un classifieur de type réseau de neurones formant un modèle neuro-markovien.

Il est assez naturel de modéliser une ligne de texte par une séquence de caractères alphanumériques classés selon les quatre états : numériques (Digit, Double Digit, Séparateur) et non numérique (Rejet) présentés dans la partie précédente. Dans tous les problèmes où l'on doit modéliser des séquences (reconnaissance de la parole, de l'écrit, extraction d'informations textuelles dans des documents, etc.), les modèles de Markov cachés se sont révélés particulièrement efficaces pour deux raisons principales. Tout d'abord le cadre statistique dans lequel ils se placent les rend très robustes aux variabilités des signaux réels. Ensuite, ce sont des modèles dynamiques capables de segmenter les séquences en faisant intervenir la reconnaissance. Ils sont donc tout à fait adaptés au problème que nous considérons.

Il nous reste à définir la nature des observations fournies aux modèles de Markov cachés. Les modèles de Markov cachés peuvent être discrets si les observations appartiennent à un alphabet fini de symboles, ou continus si les observations sont continues. La grande variabilité de l'écriture manuscrite nous a conduit à privilégier des observations continues. Dans les modèles de Markov cachés «classiques», la vraisemblance des observations continues est modélisée par des mélanges de gaussiennes dont les paramètres sont estimés lors de l'apprentissage du modèle. Un autre type d'approche consiste à remplacer les mélanges de gaussiennes par des classifieurs de type réseau de neurones. Dans ce cas, les sorties des réseaux de neurones four-

nissent des probabilités *a posteriori* qui doivent être divisées par les probabilités *a priori* des classes pour obtenir des vraisemblances normalisées. Nous choisissons cette dernière approche, qualifiée d' «hybride» ou de «neuro markovienne», car elle permet de bénéficier du pouvoir discriminant des réseaux de neurones et de la capacité de modélisation des séquences des modèles de Markov cachés (voir section 1.5.3).

Un modèle de Makov caché continu se définit par les éléments suivants :
- Un ensemble de N états $S_1, S_2, \ldots, S_N$
- La matrice des probabilités de transition entre les états $A = \{a_{ij}\}$. Si q_t désigne l'état courant au temps t, on a :

$$a_{ij} = P(q_{t+1} = S_j \mid q_t = S_i), \quad 1 \leq i, j \leq N$$

- Les probabilités d'émission des symboles $b_j(k) = P(O_t \mid q_j)$ sont obtenues à partir des probabilités a posteriori $P(q_j \mid O_t)$ fournies par le réseau de neurones grâce à la règle de Bayes :

$$\frac{P(O_t \mid q_j)}{P(O_t)} = \frac{P(q_j \mid O_t)}{P(q_j)}$$

  La vraisemblance normalisée $P(O_t \mid q_j)/P(O_t)$ est ainsi obtenue en divisant les probabilités *a posteriori* par les probabilités *a priori*.
- La matrice des distributions des états initiaux π :

$$\pi_i = P(q_1 = S_i), \quad 1 \leq i \leq N$$

La construction et l'apprentissage des modèles consistera à déterminer les états des modèles, la matrice de transition entre les états A, et la matrice des distributions des états initiaux π. Cette étape est décrite dans la section 4.2.2.

4.1.2 Description de la chaîne de traitement

La mise en oeuvre de cette approche nécessite un certain nombre de modules de traitement, qui s'organisent de manière séquentielle selon la figure 4.4.

Segmentation en lignes : les lignes de texte sont extraites grâce à une approche de regroupement des composantes connexes identique à celle de la première chaîne de traitement. Nous renvoyons à la section 3.2 pour une description de la méthode.

Classification des composantes connexes : il s'agit de classifier les composantes connexes de chaque ligne selon qu'elles appartiennent à un champ numérique (Digit, DoubleDigit, Séparateur) ou non (Rejet). La description du processus de classification de ces classes syntaxiques basé sur un classifieur neuronal est présentée dans la section 4.2.1.

Analyse syntaxique : cette étape permet d'extraire les champs recherchés grâce à l'analyse syntaxique des lignes de texte. L'analyseur syntaxique corrige les éventuelles erreurs de classification de l'étape précédente en alignant les hypothèses

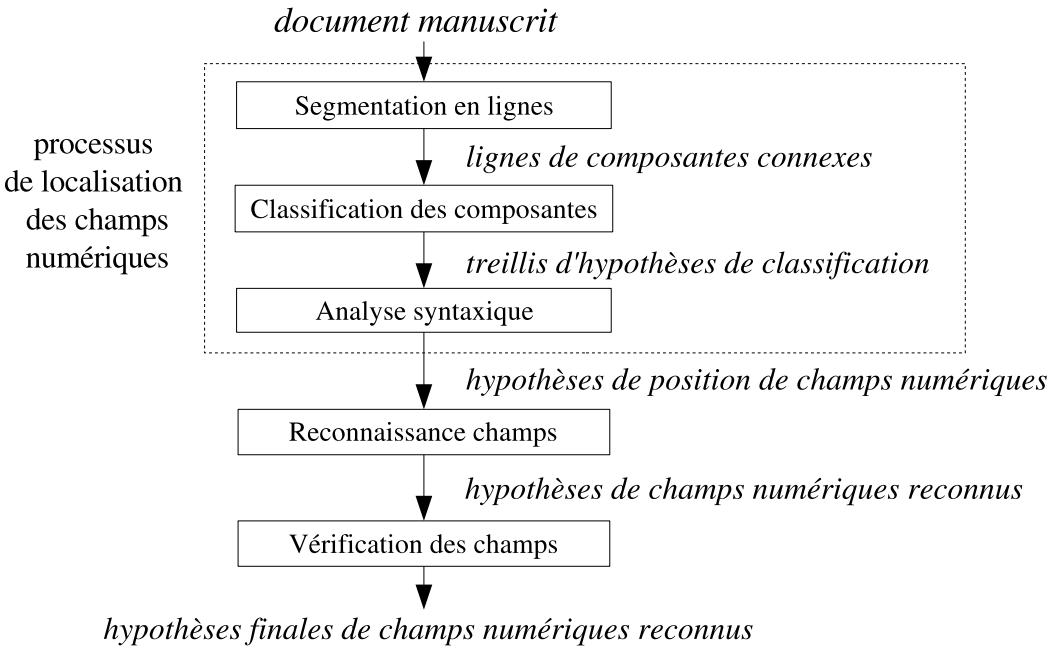


FIG. 4.4 – Chaîne globale de traitement pour l'extraction et la reconnaissance de champs numériques selon la méthode dirigée par la syntaxe.

de reconnaissance sur un modèle markovien d'une ligne de texte pouvant contenir un champ numérique. La construction et l'apprentissage de ce modèle sont décrits dans la section 4.2.2.

Reconnaissance des champs numériques : ce module traite les hypothèses de localisation des champs fournies par l'analyseur syntaxique. Il s'agit de déterminer la valeur numérique des champs à partir des séquences de composantes extraites. Ce module repose sur un classifieur chiffre et une méthode de segmentation de chiffres liés décrits en section 4.3.

Vérification des hypothèses de localisation : afin de limiter la fausse alarme, et fiabiliser notre système, une étape de vérification des hypothèses de localisation et de reconnaissance des champs numériques est proposée (section 4.4).

Nous décrivons maintenant l'implémentation de ces différentes étapes.

4.2 Localisation des champs

4.2.1 Classification des composantes

Nous proposons dans cette partie une méthode de classification permettant de discriminer les composantes du document. Rappelons qu'il ne s'agit pas ici de re-

connaître des chiffres, mais de classer les composantes selon les quatre classes syntaxiques : Digit, Séparateur, Double Digit et Rejet («D», «S», «DD» et «R»). Bien que ne comportant que quatre classes, cette tâche est relativement difficile puisqu'il s'agit d'un problème de classification où les classes possèdent d'une part une grande variabilité intra-classe, et d'autre part un variabilité inter-classe parfois relativement faible (voir figure 4.5).

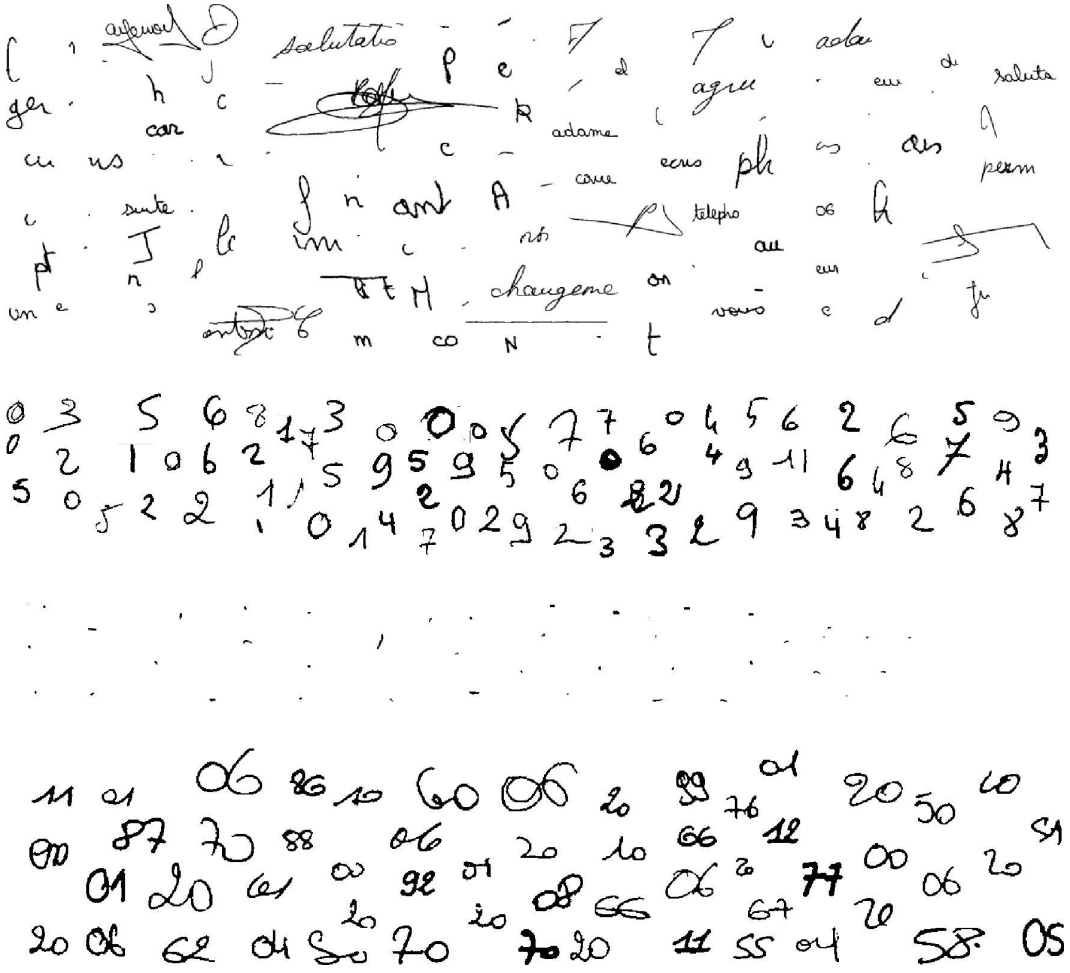


FIG. 4.5 – Les quatre classes syntaxiques : *Rejet, Digit, Séparateur, Double Digit.* Remarquons l'hétérogénéité des classes Rejet, Digit et Séparateur.

Concernant la variabilité intra-classe, elle est *a priori* importante pour trois des quatres classes du problème. En effet, la classe «Digit» est composée des 10 sous-classes de chiffres qui, comme nous l'avons montré dans la section 1.4, possèdent déjà une certaine variabilité. La classe «double digit» contient théoriquement les

100 doublons que l'on peut former avec les 10 chiffres[1], et ces chiffres sont liés de différentes manières (liaison haute, basse, double, etc.), ce qui augmente encore les combinaisons et donc la variabilité. La classe «rejet» est certainement la classe la plus éclatée puisque par définition ses éléments ne respectent pas une forme ou une structure particulière. On y retrouve donc des éléments dont la forme est très variable : mots entiers, fragments de mot, lettres isolées, bruit, etc.

La variabilité inter-classe n'est en revanche pas toujours très importante puisque certains élément appartenant à deux classes distinctes peuvent avoir une forme proche. Par exemple, il peut être difficile de discriminer certains fragments de mots des chiffres ou des chiffres liés. Les séparateurs peuvent également être facilement confondus avec du bruit.

Nous constatons donc que notre problème de classification est particulièrement délicat. Pour y remédier, nous avons choisi d'utiliser plusieurs vecteurs de caractéristiques, associés à une combinaison de classifieurs, afin de caractériser au mieux ces classes.

4.2.1.1 Classification

Pour discriminer au mieux ces quatre classes, nous avons choisi de reprendre le schéma du moteur de reconnaissance chiffre de la première approche, décrit dans la section 3.4. Rappelons qu'il s'agit d'une combinaison de classifieurs MLP alimentés par deux vecteurs de caractéristiques conséquents : un vecteur à 117 caractéristiques de 6 familles statistique/structurelle [Heutte 98], et un vecteur à 128 caractéristiques issues du chaincode [Kimura 94].

Afin de constituer la base d'apprentissage nécessaire à l'entrainement des MLP, nous disposons d'une base de courriers entrants étiquetée au niveau champ (voir section 2.2.3). La position et la valeur de chaque champ numérique sont donc étiquetées, mais les composantes connexes ne le sont pas. Les composantes rejets étant par définition toutes celles qui n'appartiennent pas à un champ numérique, il nous faut donc étiqueter les composantes Digits, Séparateurs et Double Digits des champs numériques. Plutôt que d'effectuer un étiquetage manuel long et fastidieux, nous avons préféré procéder à un étiquetage semi-automatique. À l'aide d'un filtrage sur les dimensions des composantes, la majorité des séparateurs sont identifiés. Pour identifier les chiffres, nous avons appliqué le classifieur chiffre de la section 3.4 et seuillé la sortie possédant la plus grande confiance. Au dessus du seuil, la composante est considérée comme chiffre. Enfin le «sac» de composantes restantes comprend une majorité de chiffres liés. Les quatre classes de composantes ainsi étiquetées sont enfin vérifiées manuellement pour corriger ces erreurs restantes. Au final, on otient une base d'apprentissage et une base de test dont les effectifs sont rapportés dans le tableau 4.1.

À l'issue de l'entraînement des deux classifieurs, les taux de reconnaissance moyens sur la base de test sont présentés dans la table 4.2, où MLP117 désigne

[1] On observe toutefois dans la pratique que certains chiffres liés sont plus fréquents : «00» et «06» principalement.

effectifs	R	D	S	DD	Total
Base d'apprentissage	7008	4968	522	334	12832
Base de test	3609	2559	268	171	6607

TAB. 4.1 – Effectifs des classes pour la base d'apprentissage et la base de test.

le MLP entraîné sur le vecteur statistique/structurel, et MLP128 désigne le MLP entrainé sur le vecteur du chaincode. Il semble que MLP128 donne les meilleures performances ; cependant il convient d'observer le comportement des classifieurs sur chaque classe.

taux de reconnaissance	Rang 1	Rang2
MLP128	0,69	0,92
MLP117	0,63	0,86

TAB. 4.2 – Taux de reconnaissance moyens en rang 1 et 2 pour les deux classifieurs

Les matrices de confusion des deux classifieurs sur la base de test sont présentées dans les tables 4.3 et 4.4, où les valeurs en «rang N» représentent la proportion d'éléments classés dans les N premières propositions du classifieur. On constate que chaque classifieur possède un comportement spécifique : le MLP128 présente les meilleurs résultats en rang 1 sur la classe Rejet, alors que MLP117 possède un meilleur taux de reconnaissance sur les trois autres classes. En revanche, MLP128 génère globalement moins de confusions. On retrouve des résultats similaires en rang 2. Il est raisonnable de penser que ces deux classifieurs sont complémentaires pour la discrimination des quatre classes : nous avons donc combiné leurs sorties afin d'obtenir la meilleure classification possible.

Rang 1	R	D	S	DD
R	0,65	0,16	0,09	0,09
D	0,10	0,72	0,01	0,17
S	0,02	0,03	0,92	0,03
DD	0,15	0,15	0,01	0,68

Rang 2	R	D	S	DD
R	0,93	0,50	0,13	0,43
D	0,54	0,91	0,03	0,51
S	0,79	0,11	0,94	0,16
DD	0,50	0,55	0,04	0,91

TAB. 4.3 – Matrices de confusion MLP128.

Plusieurs règles de combinaison peuvent être utilisées pour fournir la sortie de la combinaison [Rahman 03] : le maximum, le minimum, la médiane, le produit, la combinaison linéaire sont les plus couramment utilisés. Nous avons essayé deux méthodes de fusion : le produit et la moyenne (resp. «prod» et «mean»). Le tableau 4.5 donne les taux de reconnaissance (TR) en rang 1, 2 et 3 sur la base de test.

Ces résultats montrent la nette supériorité des combinaisons de classifieurs sur les classifieurs simples. L'opérateur produit semble donner les meilleurs résultats ;

Rang 1	R	D	S	DD
R	0,51	0,19	0,12	0,18
D	0,04	0,73	0,02	0,22
S	0,00	0,02	0,96	0,02
DD	0,06	0,13	0,01	0,80

Rang 2	R	D	S	DD
R	0,77	0,41	0,20	0,62
D	0,25	0,94	0,08	0,73
S	0,62	0,04	0,97	0,36
DD	0,34	0,66	0,02	0,98

TAB. 4.4 – Matrices de confusion MLP117.

Classifieur	TR1	TR2	TR3
MLP117	0,63	0,86	0,96
MLP128	0,69	0,92	0,99
prod(MLP117, MLP128)	0.76	0.95	0.99
mean(MLP117, MLP128)	0.74	0.92	0.99

TAB. 4.5 – Taux de reconnaissance (rangs 1, 2 et 3) pour les deux classifieurs et leur combinaison selon les opérateurs «prod» (produit) et «mean» (moyenne arithmétique).

nous conservons donc ce type de combinaison.

À l'issue de cette étape, nous disposons d'un système capable de discriminer les quatre classes. Pour chaque composante, le classifieur fournit une liste ordonnée des hypothèses de classification associées chacune à une mesure de confiance. La figure 4.6 montre un exemple de classification des composantes d'une ligne de texte par notre système. Remarquons la fréquence des erreurs de classification : les deux chiffres `0` et les composantes à rejeter `SA` et `E` sont classés en double digits, les lettres `D` et `S` sont reconnues comme des chiffres, et le tiret entre `SAINT` et `DENIS` est reconnu comme un séparateur. Ces erreurs montrent la difficulté de ce problème de classification. On peut en effet se demander comment le classifieur peut différencier le tiret d'un séparateur, ou le `S` d'un chiffre 5. Selon nous, la seule manière de lever les ambiguïtés est de prendre en compte le contexte, c'est-à-dire les observations voisines. C'est le rôle de l'étape suivante qui consiste à corriger les confusions du système de classification en alignant la séquence d'observations sur un modèle de syntaxe valide.

4.2.2 Analyseur syntaxique

Nous discutons dans cette partie de la construction et de l'apprentissage des modèles de Markov décrivant une ligne de texte pouvant contenir ou non un champ numérique.

Il nous faut dans un premier temps définir les états des modèles. Pour cela, prenons par exemple le cas d'un numéro de téléphone français à dix chiffres. Nous considérons dix états «Digit», ainsi que cinq états «Double Digit», correspondant aux

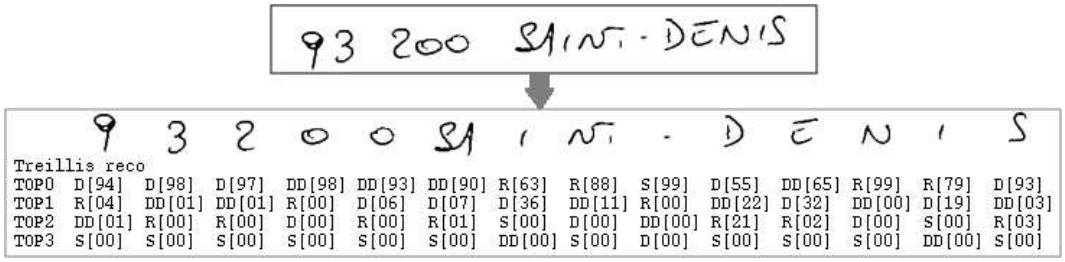


FIG. 4.6 – Hypothèses de classification des composantes d'une ligne de texte contenant un code postal

regroupements possibles entre deux chiffres consécutifs. Quatre états «séparateurs» sont également prévus pour les points et tirets potentiels entre les paires de chiffres. Dans la mesure où la majorité des lignes ne contiennent pas de champ, les modèles doivent pouvoir prendre en compte également les lignes composées exclusivement de rejet. Un seul état de rejet prenant en compte toutes les situations possibles est toutefois suffisant. Nous avons donc un nombre d'états $N = 10 + 5 + 4 + 1 = 20$.

Dans le cas où la structure du modèle est inconnue, on utilise généralement l'algorithme itératif non supervisé de Baum-Welch [Rabiner 90] qui détermine à la fois les états du modèle et les transitions entre ceux-ci. Ici, les états et la structure du modèle sont déjà connus, et seule la matrice $A = \{a_{ij}\}$ des probabilités de transition de l'état i vers l'état j est à déterminer. Les probabilités de transition sont ainsi obtenues par une simple estimation statistique sur la base d'apprentissage de 293 courriers contenant les trois types de champs. On peut ainsi représenter le modèle syntaxique d'une ligne de texte contenant respectivement un code postal, un numéro de téléphone et un code client sur les figures 4.7, 4.8 et 4.9, où les transitions sont probabilisées (par souci de lisibilité, ces probabilités ne sont toutefois pas indiquées).

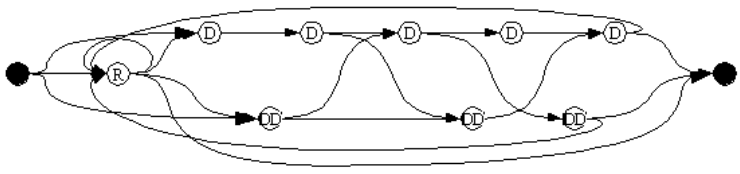


FIG. 4.7 – Modèle de Markov pour une ligne de texte contenant un code postal. Les probabilités de transition non nulles entre états sont représentées par des flêches.

Comme pour la matrice des probabilités de transition entre les états, la matrice des distributions des états initiaux $\pi = \{\pi_i\}$ est obtenue par estimation statistique sur la base étiquetée : il suffit de comptabiliser les étiquettes des premières composantes des lignes contenant un champ numérique.

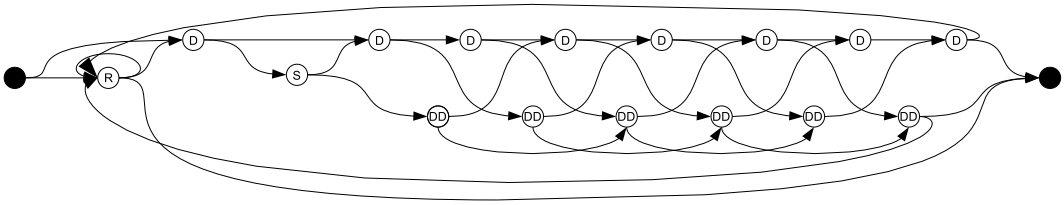


FIG. 4.8 – Modèle de Markov pour une ligne de texte contenant un numéro de téléphone.

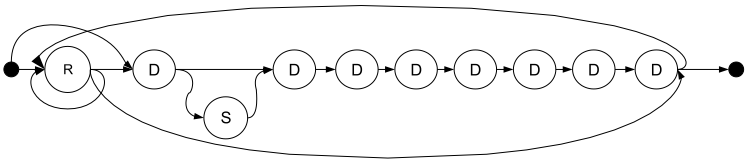


FIG. 4.9 – Modèle de Markov pour une ligne de texte contenant un code client.

Une alternative à cette mise en œuvre serait donc la détermination conjointe des états du modèle et des probabilités de transitions entre ces états par l'algorithme de Baum-Welsh. Dans ce cas, l'apprentissage du classifieur neuronal devrait être embarqué dans le processus d'apprentissage du modèle de Markov pour lui apprendre à reconnaître les différents états du modèle. Un aspect pratique de notre variante est que le classifieur ne comporte que quatre classes pour tous les types de champs. En phase de décision, les probabilités d'appartenance d'une composante à chacune des quatre classes sont recopiées sur tous les états relatifs à cette classe.

L'étape de classification produit un treillis d'hypothèses de reconnaissance, soumis à l'analyseur syntaxique qui donne les meilleurs alignements sur une syntaxe donnée. La figure 4.10 montre un exemple de recherche de code client dans une ligne de texte manuscrit : le meilleur chemin est mis en évidence, et les champs ainsi détectés sont encadrés dans le treillis. On peut constater que le code client est bien localisé. On remarque également que l'alignement proposé par l'analyseur localise un autre code client en fin de ligne, dans le numéro de téléphone. Cette «fausse alarme» s'explique par le fait que la syntaxe d'un code client est contenue dans celle d'un numéro de téléphone.

Réciproquement, la figure 4.11 montre la recherche d'un numéro de téléphone dans cette même ligne de texte. On peut constater que l'analyseur a correctement localisé le numéro de téléphone, en générant une fausse alarme au niveau du code client. Nous discuterons par la suite des moyens permettant de limiter ces fausses alarmes.

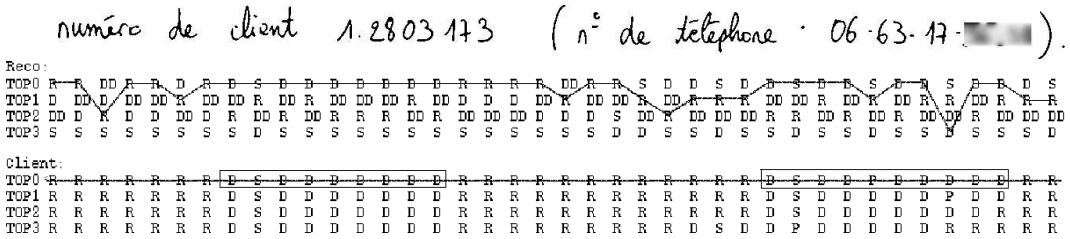


FIG. 4.10 – Alignement des hypothèses de classification des composantes d'une ligne de texte sur le modèle syntaxique d'une ligne contenant un code client.

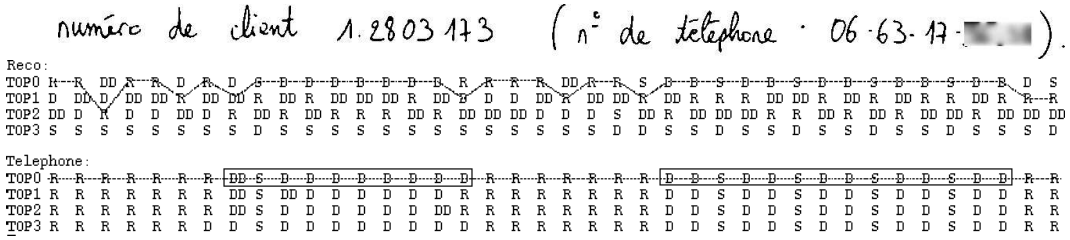


FIG. 4.11 – Alignement des hypothèses de classification des composantes d'une ligne de texte sur le modèle syntaxique d'une ligne contenant un numéro de téléphone.

Nous avons décrit la construction des modèles de Markov permettant l'extraction des champs numériques dans un document sans reconnaissance chiffre ni segmentation. Nous présentons maintenant les performances de localisation de cette approche.

4.2.3 Résultats à l'issue de la localisation des champs.

Les expérimentations ont été réalisées sur deux bases distinctes d'images de courriers entrants manuscrits provenant du service de réception du courrier d'une grande entreprise : la première (292 images) a été utilisée comme base d'apprentissage pour la classification des composantes connexes ainsi que pour déterminer les probabilités de transitions des modèles de Markov et pour paramétrer le système ; la seconde (293 documents) a servi à tester notre approche.

La détection des champs numériques est réalisée en effectuant l'analyse de chaque ligne d'un document. Le module d'extraction étiquette l'ensemble des composantes de la ligne en cours d'analyse. La séquence d'étiquettes peut alors être composée exclusivement de rejet, ou contenir un ou plusieurs champs. Un champ est considéré comme convenablement détecté si et seulement si *aucune composante du champ étiqueté n'est rejetée et si toutes les composantes connexes dans le champ détecté appartiennent au champ étiqueté*. On ne comptabilise donc que les champs parfaitement alignés (voir figure 4.12).

La figure 4.13 présente un exemple de détection de codes postaux, numéros de

FIG. 4.12 – Exemples de champs considérés comme bien et mal détectés. L'alignement du deuxième champ est faux puisqu'il manque la dernière composante : nous le comptabilisons comme une non détection, mais aussi comme une fausse alarme.

téléphone et codes client dans un document complet. On constate que les champs sont correctement localisés, et qu'un certain nombre de champs erronés sont extraits («fausses alarmes»). Néammoins, la majorité des composantes connexes du document est rejetée et seuls les champs extraits par l'extracteur sont susceptibles d'être soumis à un moteur de reconnaissance.

Le tableau 4.6 donne les taux de détection des champs en rang 1, 2 et 5. Ces résultats montrent des résultats intéressants puisque des taux de détection de 69, 75 et 81% sont obtenus en première proposition respectivement pour les codes postaux, les numéros de téléphones et les codes clients. Si l'on observe les 5 propositions, ces chiffres atteignent 87, 90 et 92%.

Type de champ	codes postaux	téléphones	codes client
Effectifs	328	240	150
Taux de détection RANG1/2/5	69 / 82 / 87	75 / 82 / 90	81 / 88 / 92

TAB. 4.6 – Taux de détection en rang 1/ rang 2/ rang 5.

On constate que les résultats sont meilleurs pour les champs qui possèdent une syntaxe plus contraignante tels que le numéro de téléphone et le code client (nombre de chiffres plus important, présence de séparateurs) que sur les champs faiblement contraints (codes postaux).

Nous observons également sur la table 4.6 que le taux de détection progresse très significativement entre la première et la deuxième proposition, et qu'il atteint près de 90% en considérant les cinq premières propositions. Sur la base de test, les résultats montrent que 90% des composantes connexes du document peuvent être rejetées si l'on ne considère que la première proposition. Seul 10% du document est donc à soumettre au processus de reconnaissance. Ces résultats prometteurs montrent le potentiel de l'approche à fournir les bonnes séquences de composantes tout en rejetant la majorité du document.

Les causes de non détection sont multiples. On peut les classer suivant deux types d'erreurs : les erreurs dues aux limites de la modélisation, et les erreurs dues à la classification syntaxique. Concernant le premier point, la seule véritable cause récurrente de non détection concerne les codes postaux contenant des triple digits. En effet, le système est actuellement incapable de prendre en compte ces composantes. Il faudrait pour cela introduire une nouvelle classe de composantes triple digit «DDD».

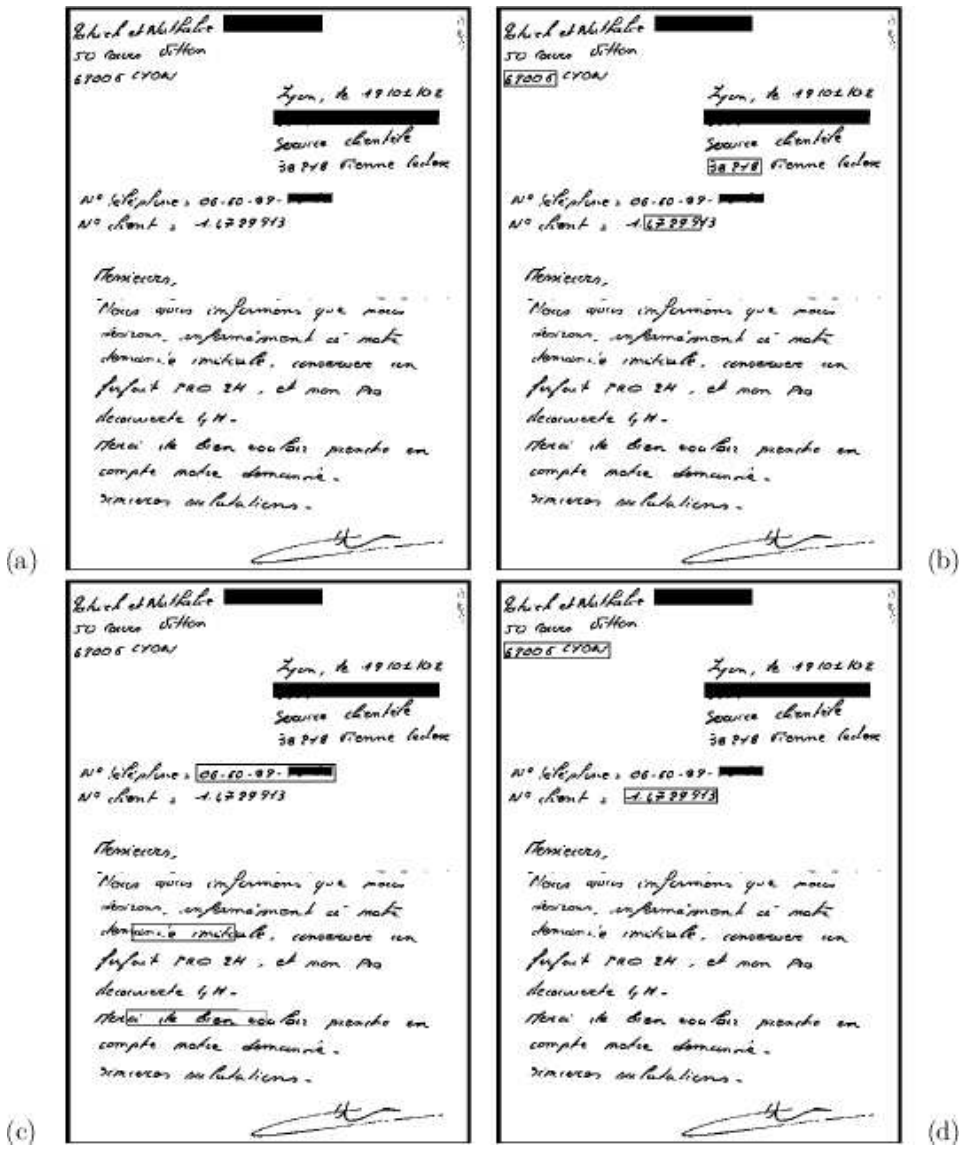


FIG. 4.13 – (a) Image originale ; (b) codes postaux détectés (séquences encadrées), (c) numéros de téléphone, (d) codes clients.

Les autres types de champs ne contenant pas de triple digit, ils ne sont pas affectés par cette limitation. En ce qui concerne les erreurs de classification, elles sont très diverses et il est difficile de dégager des causes récurrentes. Nous pouvons toutefois mentionner la tendance du système de classification à classer certaines formes de chiffre aplaties en tant que double digit, en particulier les '0', ou certains rejet reconnus comme des chiffres : confusion entre les « :» précédents un numéro de

téléphone ou un code client avec un 1 par exemple. La définition de caractéristiques dédiées aux spécificités de ces confusions pourront être mis en œuvre pour résoudre ces problèmes. I

Nous pouvons également présenter ces résultats sous la forme de compromis rappel-précision où un champ bien localisé est considéré comme bien extrait. Le rappel désigne donc ici la proportion de champs bien localisés (taux de détection). Ces résultats sont présentés en figure 4.14.

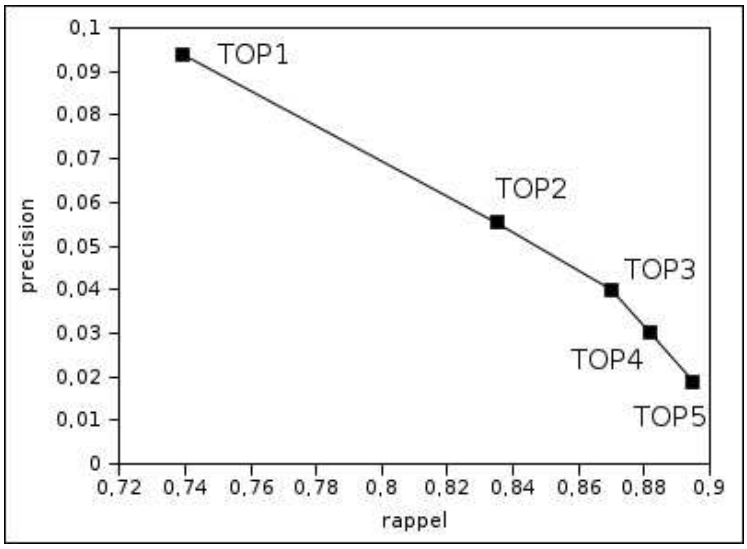


FIG. 4.14 – Courbe rappel/précision du système avant l'étape de reconnaissance des champs, en considérant un champs bien localisé comme correctement extrait.

Ces résultats sont intéressants puisqu'ils présentent également la précision du système. On peut constater que si les résultats en détection (rappel) sont très encourageants, la précision du système est peu élevée puisqu'elle est de 9,4% en première proposition et qu'elle chute en dessous des 2% lorsqu'on considère les 5 premières propositions. En conservant les N meilleurs choix de l'analyseur, le rappel augmente et la précision diminue. Nous détectons donc plus de champs mais nous générons aussi davantage de fausse alarme. L'augmentation de la fausse alarme est logique : parmi les N alignements proposés, il ne peut y avoir qu'une seule bonne proposition et qu'un seul alignement composé exclusivement de Rejet. Les N, $N-1$ ou $N-2$ alignements restants dans la liste de propositions contiennent donc une fausse alarme. La fausse alarme est ainsi directement proportionnelle au nombre de propositions considéré.

Nous avons présenté dans cette partie les résultats obtenus à l'issue de l'étape de localisation des champs. L'étape suivante consiste à soumettre ces hypothèses de localisation au module de reconnaissance de champs numériques.

4.3 Reconnaissance des champs

Dans cette section, nous nous focalisons sur la reconnaissance des champs localisés lors de l'étape précédente. Nous en présentons le principe, puis évaluons les performances de la méthode proposée sur une base de champs isolés. Nous donnons dans une troisième partie les résultats du système complet à l'issue de cette étape de reconnaissance des champs localisés.

4.3.1 Principe

Contrairement aux systèmes de reconnaissance de documents manuscrits où la localisation et la reconnaissance des informations sont intimement liées, l'exploitation de la connaissance *a priori* sur la syntaxe des champs ainsi que l'utilisation d'une méthode sans segmentation nous a permis de localiser les champs numériques sans les reconnaître. La reconnaissance intervient donc en fin de traitement et permet une vérification des hypothèses de localisation.

L'étape de reconnaissance des champs numériques s'appuie sur l'exploitation des hypothèses de classification fournies lors de l'étape de détection. En effet, nous bénéficions pour chaque champ extrait de l'hypothèse de classification «Digit», «Séparateur» ou «Double digit» des composantes. Il s'agit donc de déterminer l'hypothèse de classification *chiffre* pour chacune de ces composantes (voir figure 4.15).

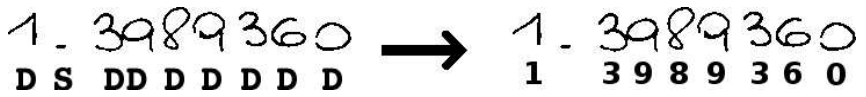


FIG. 4.15 – Détermination des hypothèses de classification *chiffre* à partir des hypothèses de classification *Digit, Séparateur, Double Digit.*

Pour les composantes dont l'hypothèse de classification est «Digit», il suffit de soumettre l'imagette à un classifieur chiffre qui déterminera la meilleure hypothèse de classification «chiffre». Les composantes «Séparateur» sont ignorées lors de cette étape, puisqu'elles n'interviennent pas dans la valeur numérique du champ à reconnaître. La reconnaissance des composantes classifiées comme «Double digit» est effectuée de la manière suivante : comme nous savons que la composante contient deux chiffres liés, il nous faut trouver la meilleure segmentation des deux chiffres, et les reconnaître. Afin d'effectuer la reconnaissance des chiffres isolés et des chiffres liés, nous utiliserons les outils dans la première stratégie. La reconnaissance de chiffres isolés s'effectue donc à l'aide du classifieur chiffre décrit dans le chapitre 3.4. Concernant la reconnaissance de chiffres liés, elle est effectuée par une stratégie de segmentation/reconnaissance à l'échelle de la composante : plusieurs hypothèses de segmentation sont générées et soumises au classifieur chiffre. La segmentation produisant les scores de confiance les plus élevées est conservée (voir figure 4.16 et le descriptif plus détaillé de la méthode en section 3.3.2).

Drop fall	ascendant gauche	ascendant droit	descendant gauche	descendant droit
chemin de coupure				
classifieur chiffres	0[98] 8[82]	2[27] 8[35]	0[73] 8[36]	0[92] 8[34]
produit des confiances	81	09	26	32

FIG. 4.16 – Exemple de reconnaissance d'une composante double digit. Plusieurs chemins de coupures sont générés et soumis au classifieur chiffre. L'hypothèse qui maximise le produit des confiances des deux premières propositions du classifieur chiffre est conservée.

4.3.2 Evaluation de la reconnaissance des champs numériques

Nous évaluons dans cette partie les performances de la méthode de reconnaissance de champs isolés développée. Nous rappelons pour cela les performances du classifieur chiffre sur lequel est repose la méthode : ses taux de bonne classification sans rejet sont de 98,72, 99,63 et 99,85% respectivement en TOP 1,2 et 3 (voir section 3.4.3). Nous détaillons maintenant les performances au niveau chiffres liés et au niveau champs.

Résultats de la reconnaissance des chiffres liés

La reconnaissance de chiffres liés est évaluée sur une base étiquetée d'environ 150 «double digit» extraits de séquences numériques. Une composante est comptabilisée comme bien reconnue si les deux chiffres qui la constituent sont bien classifiés. Le taux de reconnaissance obtenu sur cette base est de 91%. La figure 4.17 montre des exemples de doubles digits bien et mal reconnus. Les erreurs les plus fréquentes apparaissent dans les cas suivants :

- Il arrive que la segmentation échoue lorsque les chiffres liés possèdent une liaison prolongée non verticale, comme dans le cas des 00. Dans ce cas de figure, l'algorithme du drop fall commence la coupure puis continue sa course verticalement et sort des pixels noirs. Pour solutionner ce problème, nous aurions pu ajouter aux hypothèses de segmentation les variantes du drop fall décrites dans [Dey 99] qui prolongent la chute de la goutte selon l'orientation de la liaison.
- Une mauvaise reconnaissance des chiffres peut être due à la présence d'un trait de liaison relativement long (ligature) entre deux chiffres. Comme ce trait n'appartient à aucun des deux chiffres, il déforme les chiffres isolés et entraine parfois une mauvaise reconnaissance (cas du double digit 06 en bas à droite de la figure 4.17). Afin de résoudre ces problèmes, une méthode de reconnaissance de chiffres liés capable d'éliminer ces ligatures pourrait être testée. Dans [Suwa 04], une telle méthode est présentée, fondée sur une représentation en graphe de la composante.

– La mauvaise reconnaissance d'une composante double digit peut également provenir d'une erreur de classification du moteur de reconnaissance de chiffres sur une composante *a priori* bien segmentée.

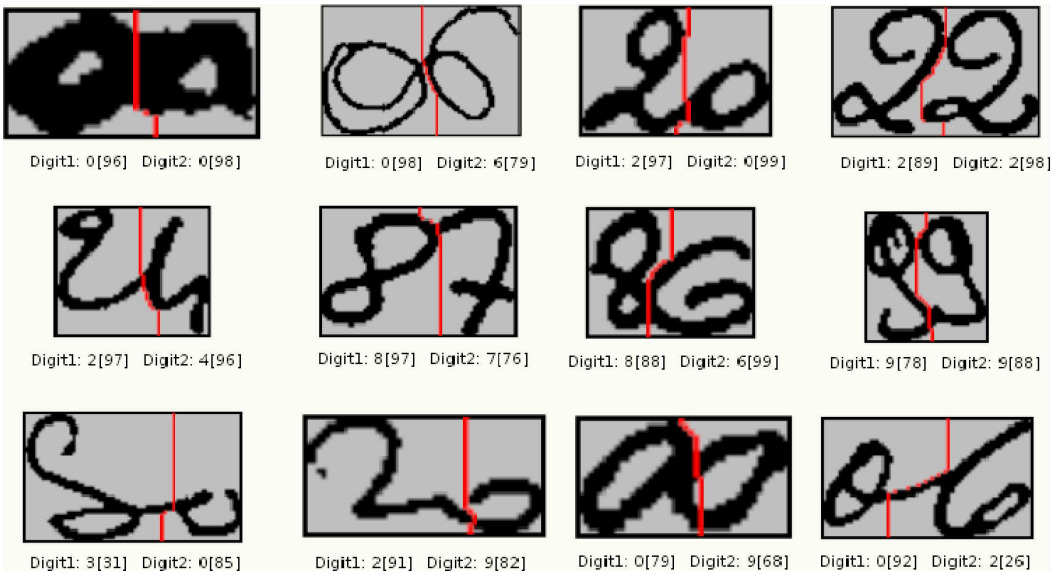


FIG. 4.17 – Exemples de reconnaissance de double digit. Les composantes des deux premières lignes sont correctement reconnues, celles de la dernière ligne sont des erreurs.

Résultats de la reconnaissance des champs isolés

Pour évaluer la reconnaissance des champs numériques, nous avons constitué une base d'environ 500 champs isolés disposant de l'étiquetage «syntaxique» (Digit, Séparateur, Double Digit), et annotés au niveau chiffre. La base provient de courriers entrants manuscrits réels, et les trois types de champs recherchés sont représentés (codes postaux, numéros de téléphone et codes clients). Nous ne comptabilisons comme bien reconnus que les champs dont toutes les composantes ont été bien reconnues au niveau chiffre. Le tableau 4.7 donne les taux de reconnaissance au niveau champs.

type de champ	codes postaux	téléphones	codes client	total
taux de reconnaissance	81,0	77,9	81,5	80,0

TAB. 4.7 – Taux de reconnaissance des champs isolés.

On constate que les taux de bonne reconnaissance au niveau champ varient assez peu suivant le type de champ. Pourtant, le nombre de chiffres est différent

suivant les champs : les codes postaux, numéros de téléphone et codes client sont respectivment composés de 5, 10 et 8 chiffres. On aurait ainsi pu s'attendre à obtenir un taux de reconnaissance pour les codes postaux nettement supérieur à celui obtenu sur les deux autres type de champ. Nous expliquons ces résultats par le fait que les codes postaux contiennent très souvent des chiffres liés, et en particulier des '00' sur lesquels le segmenteur fait parfois des erreurs (voir section précédente). Inversement, nous avons observé que les codes client contiennent peu de chiffres liés, ce qui explique leur bon taux de reconnaissance.

4.3.3 Résultats du système à l'issue de la reconnaissance

Nous présentons dans cette partie les résultats du système complet d'extraction des champs dans les courriers manuscrits, depuis l'extraction des lignes de texte jusqu'à la reconnaissance des séquences localisées.

Nous avons présenté dans la partie 4.2.3 les résultats obtenus à l'issue de l'étape de localisation des champs, nous donnons maintenant les résultats en rappel-précision du système complet, à l'issue de la reconnaissance.

Nous présentons dans cette section les résultats obtenus en fin de chaîne de traitement, à l'issue de la reconnaissance des champs. Les trois analyseurs syntaxiques sont passés successivement sur les documents afin d'en extraire les codes postaux, numéros de téléphone et code client. Nous obtenons donc en sortie du système une liste de champs reconnus de tout type. La figure 4.18 donne l'évolution de la courbe rappel/précision du système à l'issue de la reconnaissance.

On constate que le système est capable d'extraire et de reconnaître correctement de 58 à 69% des codes postaux, codes clients et numéros de téléphone suivant le nombre de propositions du système que l'on considère. Le rappel du système a donc baissé à l'issue de l'étape de reconnaissance, pour passer de 74% à l'issue de la localisation à 58% en première proposition, et de 89,5 à 69% en TOP5. Cette baisse est due aux champs correctement localisés mais mal reconnus. L'étape de reconnaissance a également fait chuter la précision, en particulier pour les premières propositions. En effet, on doit ajouter la fausse alarme due aux champs bien localisés mais mal reconnus aux fausses alarmes déjà mentionnées à l'issue de la localisation des champs.

Le problème de la faible précision est toutefois à relativiser puisque dans le cadre d'une application industrielle, plusieurs mesures simples peuvent permettre de solutionner les fausses alarmes. La première consiste à mettre en relation les champs extraits avec une base de données contenant les informations relatives aux clients de l'entreprise afin de filtrer les séquences numériques existantes. La seconde consiste à exploiter des connaissances *a priori* sur les champs recherchés (numéro de téléphone commencant par '06', code client commencant par '1', etc.). Afin d'améliorer les performances en précision du système, nous proposons en section 4.4 un module de vérification permettant d'accepter ou de rejeter les séquences de composantes reconnues.

Le temps de traitement est également un critère important pour évaluer le

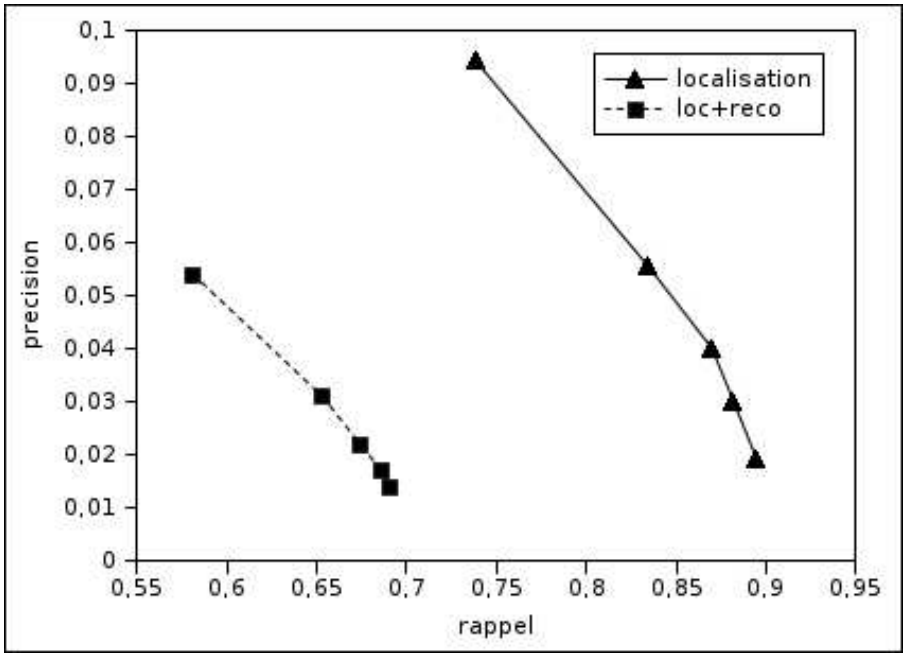


FIG. 4.18 – Courbe rappel/précision du système à l'issue de l'extraction (localisation) et à l'issue de l'étape de reconnaissance des champs (loc+reco).

système puisque notre approche se veut réaliser une extraction rapide des champs d'intérêts, afin de limiter les zones sur lesquelles la reconnaissance sera effectuée ultérieurement. Sur notre base de test de 293 images et sur une machine cadencée à 1,5GHz, il faut environ 3 secondes/image pour extraire et reconnaître les trois types de champ. Ces temps comprennent l'intégralité des traitements, depuis l'extraction des composantes connexes jusqu'à la reconnaissance des champs. Cette seconde approche est donc sensiblement plus rapide que la première approche (4,5 secondes /image). Ce résultat est logique puisque les nombreuses segmentations de la première approche requièrent un grand nombre d'appels au classifieur chiffre. Notons que comme pour la première approche, les traitements n'ont pas fait l'objet des différentes optimisations possibles en vue d'une industrialisation.

4.4 Vérification des hypothèses de champs numériques

Nous avons vu qu'à l'issue du processus d'extraction des champs, un certain nombre de fausses alarmes apparaissent parmi les solutions proposées par le système. Ces fausses alarmes ont plusieurs origines : il peut s'agir de séquences textuelles (détection d'un champ dans une zone de texte en présence notamment de caractères bâtons) ; numériques et textuelles (défaut d'alignement) ; ou même strictement numériques (détection d'un champ dans un autre, défaut d'alignement, ou

erreur lors de l'étape de reconnaissance chiffre sur un champ bien localisé).

Une grande partie de cette fausse alarme pourra être éliminée lors de la mise en relation des solutions fournies par le système à l'issue de la reconnaissance avec une base client : tous les champs proposés absents de la base seront ignorés. Notons qu'on dispose dans ce cas d'un «lexique» qui pourrait également être exploité afin de rattrapper certaines erreurs de reconnaissance chiffres dans le cas d'un champ bien localisé mais mal reconnu. Dans ce cas de figure, on se situe dans un cadre similaire aux approches de reconnaissance de mots non dirigées par le lexique.

Nous proposons toutefois une méthode de vérification dont le but est d'analyser les hypothèses de champs de manière à rejeter les fausses alarmes et à accepter les séquences numériques qui étaient effectivement à détecter. Ce module est basé sur l'interprétation d'un certain nombre d'informations obtenues tout au long de la chaîne de traitement, permettant d'accepter ou de rejeter ces hypothèses. L'étape de localisation fournit des scores d'alignement des séquences de composantes sur les modèles markoviens traduisant la qualité de l'alignement, l'étape de reconnaissance fournit des scores de confiance permettant de déceler les éventuelles composantes non numériques. Ces scores, auxquels nous avons rajouté des informations sur la régularité des boites englobantes des composantes, constituent les caractéristiques d'un vecteur soumis à un classifieur de type MLP, entraîné sur une base de champs numériques et de fausses alarmes. L'unique sortie du classifieur se prononce sur l'acceptation (sortie du MLP $> 0{,}5$) ou le rejet (sortie $< 0{,}5$) de l'hypothèse de champs. Le MLP a été entraîné sur une base de 17000 séquences de composantes (16800 fausses alarmes et 200 véritables champs).

4.4.1 Caractéristiques

Nous décrivons maintenant le vecteur de 14 caractéristiques provenant des trois familles : caractéristiques issues de la localisation, de la reconnaissance, et des boites englobantes des composantes.

Caractéristiques provenant de la localisation

Lors de l'étape de localisation, l'analyseur syntaxique fournit pour chaque ligne un score d'alignement des composantes sur les modèles (voir figure 4.19). Ce score est une indication précieuse sur la fiabilité de la localisation du champs et doit donc être retenu comme caractéristique dans notre vecteur. Lorsque le champs n'est pas proposé en première solution par l'analyseur syntaxique, nous remarquons que l'écart entre les scores est généralement faible avec les premiers alignements. Nous avons donc retenu comme caractéristiques les écarts entre le score de l'alignement du champs et les scores des autres alignements de la même ligne. L'expérience montre que la bonne proposition n'est jamais au delà de la cinquième proposition de l'analyseur. Nous avons ainsi retenu 6 caractéristiques issues de l'étape de localisation.

Caractéristiques provenant de la reconnaissance

Une autre famille de caractéristiques pour la discrimination des fausses alarmes provient de l'étape de reconnaissance. Partant de l'hypothèse selon laquelle une

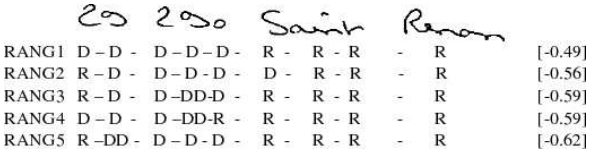


FIG. 4.19 – Les cinq premiers alignements proposés par l'analyseur syntaxique pour une ligne de texte contenant un code postal, avec les scores des alignements (scores logarithmiques).

séquence de composantes non numériques produit des confiances basses lors de l'étape de reconnaissance (voir figure 4.20), nous avons choisi d'intégrer dans le vecteur les trois caractéristiques suivantes :

- Les moyennes arithmétiques et géométriques des scores de la reconnaissance chiffre
- Parmi tous les chiffres du champs, le score minimum de la reconnaissance chiffre

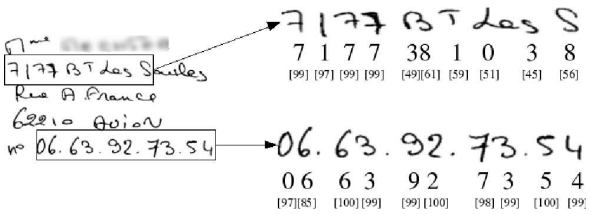


FIG. 4.20 – En-tête d'un document dans lequel l'analyseur a détecté deux numéros de téléphone. Les confiances associées à la fause alarme sont plus faibles que celle du véritable champ.

Caractéristiques morphologiques

L'observation d'un certain nombre de champs numériques et de fausses alarmes a montré que les boites englobantes des chiffres constituant un champs numérique présentent généralement des régularités que ne possèdent pas les fausses alarmes (voir figure 4.21).

Nous avons donc ajouté dans le vecteur 5 caractéristiques traduisant la régularité dans la succession des boites englobantes :

- L'écart à la moyenne des ordonnées minimum et maximum des chiffres
- L'écart à la moyenne des hauteur et largeur des chiffres
- L'écart à la moyenne entre les abscisses des centres de gravité des chiffres

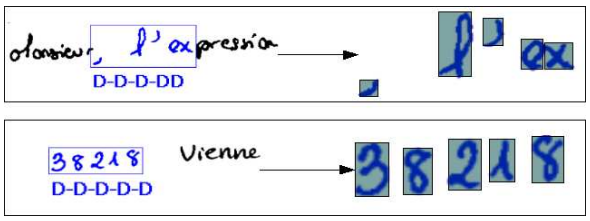


FIG. 4.21 – Boites englobantes d'une fausse alarme et d'un champs numérique. Les boites englobantes du champs numérique présentent généralement davantage de régularité (hauteur, largeur, position relative) que celles des fausses alarmes.

4.4.2 Evolution de la courbe rappel-précision

La figure 4.22 montre l'évolution du rappel et de la précision du système avant et après l'étape de vérification des hypothèses de champs numériques.

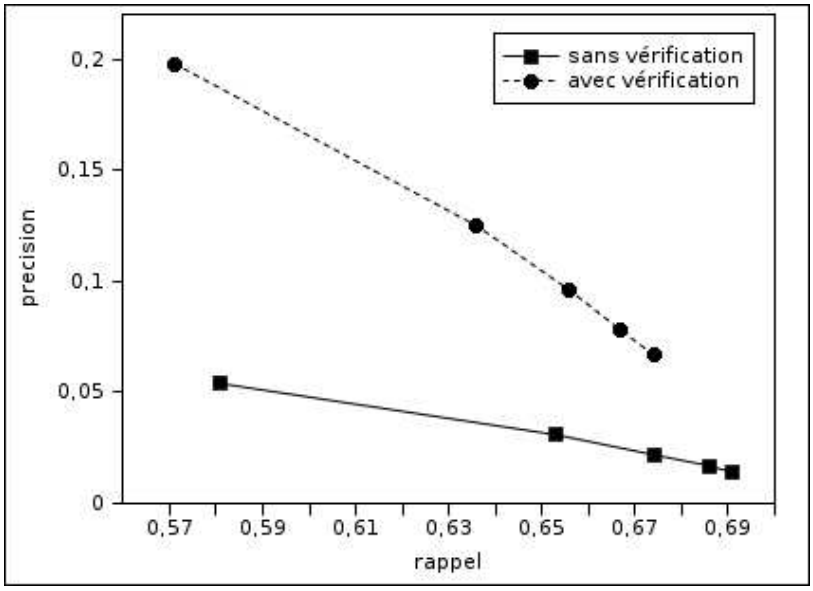


FIG. 4.22 – Courbe rappel/précision du système avant et après rejet des champs par le module de vérification.

Nous constatons que le système de rejet mis en place permet d'améliorer considérablement la précision du système, pour tous les rangs considérés. Le rappel du système est peu affecté par ce rejet pour les rangs faibles, et baisse sensiblement pour les rang plus élevés.

La figure 4.23 propose un comparatif des résultats obtenus par les deux méthodes d'extraction des champs : la stratégie de segmentation/reconnaissance/rejet et l'ap-

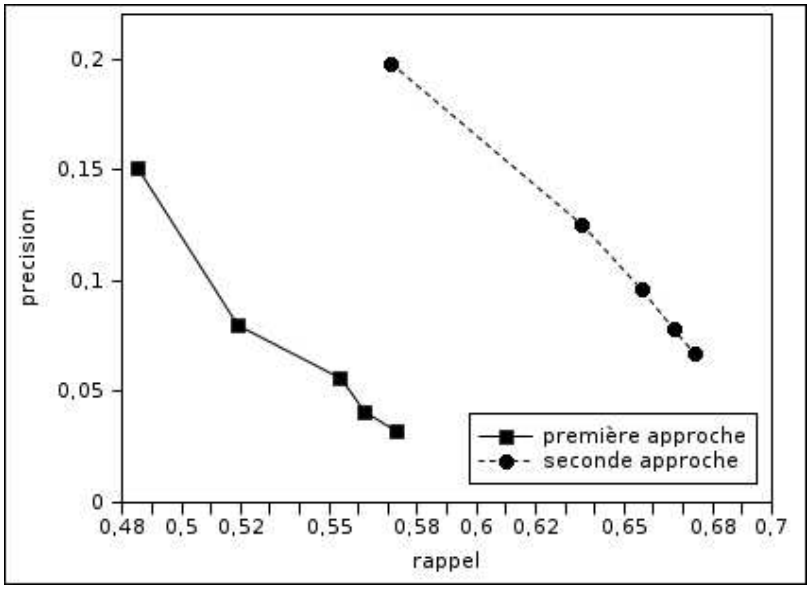


FIG. 4.23 – Comparaison des compromis rappel/précision obtenus avec la première et la seconde approche.

proche dirigée par la syntaxe. On peut constater que le rappel est nettement supérieur avec la stratégie dirigée par la syntaxe. Le problème de recherche du meilleur compromis rappel-précision étant un problème multi-objectif, on peut dire que tous les compromis obtenus avec l'approche dirigée par la syntaxe «dominent» les compromis de la première approche. Nous reviendrons sur les notions de multi-objectifs et de dominance dans le chapitre 5. Cette première évaluation va dans le sens de la seconde approche qui semble plus apte à faire ressortir les bonnes hypothèses de champ.

Il est également intéressant d'observer la répartition des différentes séquences proposées par cette seconde approche selon les 4 cas de figure identifiés dans le chapitre 3 :

- **a)** La séquence proposée correspond à un champ et sa valeur numérique est exacte.
- **b)** La séquence proposée correspond à un champ mais sa valeur numérique est fausse.
- **c)** La séquence proposée est mal alignée sur un champ.
- **d)** La séquence proposée provient d'une ligne ne contenant pas de champ.

La figure 4.24 présente la répartition des champs proposés par le système suivant les quatre catégories mentionnées ci-dessus.

Il est intéressant de constater que pour un nombre équivalent de séquences proposées, la seconde approche propose non seulement davantage de champs corrects, mais également moins d'alignements de champs manqués et moins de fausses alarmes

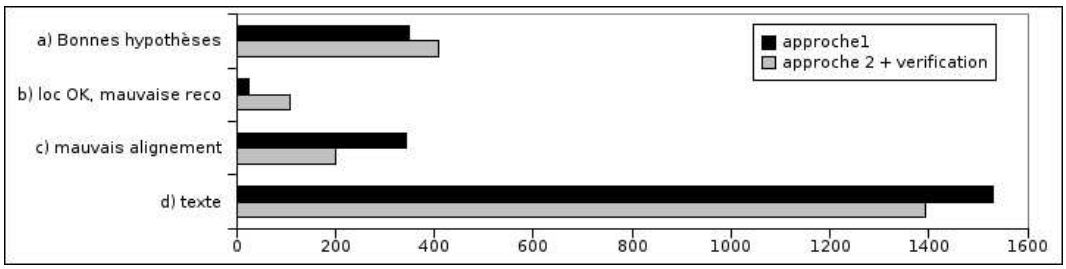


FIG. 4.24 – Nature des séquences extraites pour les deux stratégies d'extraction des champs.

dans des lignes de texte ne contenant pas de champs. Le nombre de séquences bien alignées mais mal reconnues est nettement plus important pour la seconde approche, ce qui montre encore une fois le potentiel de détection de la méthode. En optimisant la phase de reconnaissance des champs isolés, le rappel final semble de plus encore améliorable.

4.5 Conclusion

Nous avons présenté dans ce chapitre une méthode générique pour l'extraction des champs numériques dirigée par la syntaxe, et la méthode de reconnaissance associée. L'intérêt de la méthode réside dans le fait qu'elle utilise la syntaxe d'un champ numérique comme infomation *a priori* pour le localiser, sans pour autant procéder à la segmentation et la reconnaissance systématique des composantes lors de l'étape de localisation.

Nous avons pu constater que la classification des composantes en classes syntaxiques est une opération de discrimination difficile, qui produit un certain nombre d'erreurs de classification. L'analyseur syntaxique permet cependant de corriger ces confusions en faisant remonter les solutions syntaxiquement correctes dans le treillis de reconnaissance. Les résultats montrent que l'étape de localisation produit de très bons résultats. Le taux de détection d'un type de champ dépend de la complexité de sa syntaxe, c'est-à-dire du niveau de contrainte qu'elle impose à la séquence de composantes du champ. Les hypothèses de localisation sont ensuite soumises au processus de reconnaissance de champs qui dégrade sensiblement les résultats. Une majorité de champs peut toutefois être extraite, avec une précision relativement faible. Une étape de vérification des hypothèses de champs basée sur des caractéristiques extraites au cours des différentes étapes de traitement a donc permis d'augmenter significativement cette précision. Les résultats finaux en rappel-précision obtenus avec cette approche dépassent nettement ceux de la première approche.

Selon nous, cette méthode peut toutefois être améliorée sur trois points clefs.

Premièrement, les modèles de lignes de texte pourraient être enrichis d'états

supplémentaires en vue de la détection d'entités particulières d'avant champs, tels que les « : », ou la présence de mots particuliers («téléphone», «client»). Cette modélisation suppose toutefois la définition de nouveaux états qui ne faciliteront pas la discrimination des classes de composante. Un apprentissage non supervisé de la structure et des états des modèles par l'algorithme de Baum-Welsh pourrait alors être testé. Dans la même optique d'une exploitation d'information non numérique, une autre solution consisterait à coupler notre système avec celui présenté dans [Koch 06] visant à extraire des mots clefs.

Le deuxième point clef pouvant bénéficier d'améliorations concerne la reconnaissance des champs localisés. En effet, les erreurs lors de cette étape font à la fois baisser le rappel et la précision. On pourra améliorer la précision du classifieur chiffre, par exemple par l'ajout d'une seconde étape de décision par SVM lorsque le classifieur MLP hésite entre plusieurs hypothèses [Bellili 03]. De nouvelles méthodes de segmentation des chiffres liés adaptées au problème les plus fréquents pourraient également être mises en place.

Enfin le dernier point clef concerne la difficile classification Rejet-Digit-Séparateur-DoubleDigit. En effet, nous avons constaté qu'elle était particulièrement délicate et qu'elle génèrait un certain nombre d'erreurs, principalement à cause de la classe Rejet. Le processus de classification pourrait donc être amélioré, soit par l'ajout de nouvelles caractéristiques, soit en abordant le problème du Rejet différemment. C'est précisément ce que nous proposons dans le chapitre 5, où une stratégie de rejet en deux étapes vise à améliorer les capacités de rejet des deux approches existantes.

Chapitre 5

Gestion du rejet

5.1 Introduction

Dans les deux chaînes de traitement présentées dans les chapitres précédents, on cherche à modéliser une ligne de texte manuscrit pouvant contenir un champ numérique. Ce modèle de ligne est constitué d'un modèle de champ numérique (l'information recherchée) et d'un modèle de rejet permettant d'absorber l'information non pertinente. Si la modélisation des champs numériques diffère dans les deux approches, le rejet est quant à lui modélisé de la même manière par un état unique «Rejet». Cette modélisation volontairement grossière des éléments non pertinents permet de n'effectuer qu'une reconnaissance partielle du document. En revanche, elle impose de discriminer les formes appartenant à un champ numérique (chiffres, séparateurs) du reste du document (que nous appelerons «rejets»). Cette discrimination est une opération délicate pour deux raisons : (i) d'une part, l'extrême variabilité des formes n'appartenant pas à un champ numérique (mots ou fragments de mots, ponctuation, bruit, symboles, ratures, ainsi que toutes les formes mal segmentées, y compris les chiffres mal segmentés) rend difficile la modélisation d'une telle classe, (ii) d'autre part, la ressemblance entre certaines formes rejets et les entités appartenant à un champ numérique (chiffres/lettre ou fragment de mot, séparateur/bruit ou ponctuation) engendre un recouvrement entre les deux catégories d'entités. Ainsi, la capacité du module de reconnaissance à discriminer les formes appartenant à un champ numérique du reste du document joue selon nous un rôle central pour l'extraction de champs numériques dans les documents manuscrits. Dans ce chapitre, nous cherchons donc à améliorer les capacités de rejet vis-à-vis des formes non pertinentes dans les deux chaînes de traitement.

Rappelons comment est effectuée la discrimination entre les formes appartenant à un champ numérique et les formes à rejeter dans les deux approches développées :

- Dans la première approche, on procède à une segmentation/reconnaissance chiffre de toutes les composantes du document, et tout ce qui n'est pas un chiffre doit être rejeté (les séparateurs sont identifiés par la suite). La discrimination entre les chiffres et le reste du document est effectuée en exploitant les

capacités de rejet du classifieur chiffre par une analyse des scores de confiance de ses sorties. Cette méthode n'est pas optimale dans la mesure où le classifieur utilisé est discriminant (combinaison de MLP) et qu'on sait qu'il ne présente pas de bonnes capacités de rejet [Gori 98].

– Dans la seconde approche, on cherche à distinguer les composantes appartenant à un champ numérique du reste du document sans les segmenter. Le problème du rejet est donc différent de la première approche puisque les chiffres liés et séparateurs ne doivent pas être rejetés. Dans cette approche, la gestion du rejet est effectuée en considérant une classe de rejet dans un problème de classification à quatre classes : Rejet, Digit, Séparateur, Double Digit.

Dans ce chapitre, nous proposons d'améliorer les capacités de rejet des deux méthodes existantes. Après avoir présenté les méthodes de la littérature pour le rejet des entités non pertinentes dans la section 5.2.2, nous choisissons d'améliorer les capacités actuelles de rejet de nos systèmes à l'aide d'une étape de filtrage des rejets dits «évidents» en section 5.3. La mise en œuvre de la phase de filtrage nécessite une étape de classification où les coûts de mauvaise classification sont très déséquilibrés et inconnus. Nous proposons donc dans la section 5.4 une méthode originale d'apprentissage d'un ensemble de classifieurs SVM basée sur un algorithme évolutionnaire.

5.2 État de l'art sur la gestion du rejet

On distingue souvent deux types de rejet : le rejet dit d'*ambiguïté* concerne les formes qui se situent proches des frontières de deux ou plusieurs classes, et le rejet de *distance* qui concerne les formes qui sont éloignées de toutes les classes (voir figure 5.1).

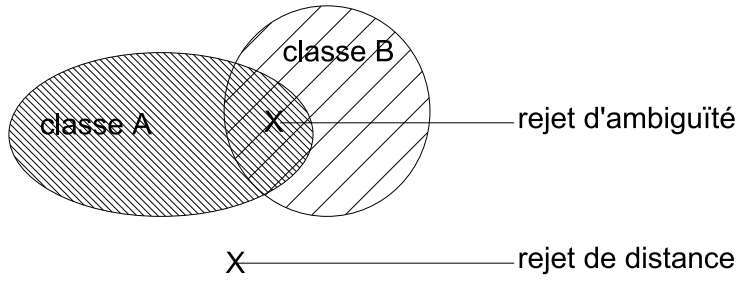


FIG. 5.1 – Rejet de distance et rejet d'ambiguïté.

Dans la littérature concernant la reconnaissance d'entités numériques, on rencontre plusieurs méthodes de gestion du rejet, plus ou moins aptes à traiter le rejet de distance et/ou d'ambiguïté. Pour un problème de classification de chiffres avec gestion du rejet, les entités très différentes des chiffres (mots, fragment de mots, bruit) relèveraient plutôt du rejet de distance, alors que les formes proches des chiffres (chiffres mal segmentés ou mal formés, lettres minuscule ou majuscule

isolés, certains fragments de mots, etc.) concernent plutôt le rejet d'ambiguïté. Nous présentons dans cette section un aperçu des solutions disponibles pour la gestion des deux types de rejet dans le cadre de la reconnaissance de chiffres manuscrits.

5.2.1 Revue des méthodes de gestion du rejet d'ambiguïté

Vérification des hypothèses de reconnaissance

Une manière de rejeter les formes ambigües consiste à mettre en œuvre une méthode de vérification en post-traitement d'un classifieur classique n'ayant appris qu'à discriminer les classes connues. Il s'agit d'accepter ou de rejeter les hypothèses de reconnaissance fournies par le classifieur. La vérification peut être effectuée en extrayant un jeu de caractéristiques provenant des scores de confiance du classifieur [Pitrelli 03, Gorski 97] analysées pour accepter ou rejeter la forme. C'est la méthode utilisée pour le traitement du rejet dans notre première stratégie (voir section 3.5). D'autres travaux utilisent des vecteurs de concavités [Oliveira 02b] ou des caractéristiques géométriques ou contextuelles [Zhou 02, Oliveira 02b] ensuite soumises à un classifieur à deux classes capable de rejeter les formes non numériques.

Apprentissage d'exemples de rejet

Il est possible d'entraîner des classifieurs avec une classe rejet afin de traiter les non-chiffres. Il s'agit de la méthode employée dans notre seconde approche (voir section 4.2.1). Dans [Bromley 93] et [Liu 02c], cette stratégie de rejet est appliquée à un réseau de neurones pour la reconnaissance de chiffres et mots manuscrits afin d'améliorer la resistance du classifieur aux formes mal segmentées. Dans [Liu 04], les auteurs présentent un système de reconnaissance de séquences numériques, reposant sur une stratégie de segmentation reconnaissance. Comme la segmentation génère des chiffres et des non-chiffres, le classifieur doit être résistant aux non-chiffres, c'est à dire que le classifieur doit avoir la capacité de fournir des scores bas pour toutes les classes lorsqu'un non-chiffre lui est présenté. La comparaison de plusieurs types de classifieurs entraînés avec et sans exemples de rejet montre que le taux de reconnaissance au niveau chiffre chute très légèrement lorsque des exemples de rejet sont présents dans la base d'apprentissage, quel que soit le type de classifieur, le MLP obtenant les meilleurs résultats sur la base CEDAR, le SVM à noyau RBF sur la base NIST. Les performances en discrimination sont peu affectées par l'ajout des rejets. En revanche, la comparaison des résultats au niveau séquence numérique montre des résultats systématiquement supérieurs pour les classifieurs ayant été entraînés avec du rejet.

5.2.2 Méthodes pour la gestion du rejet de distance

Classifieurs paramétriques modélisants

Les classifieurs paramétriques modélisants sont basés sur des estimateurs de densité de probabilité qui déterminent un modèle de chaque classe. En phase de décision, il est possible de calculer une distance entre la forme à reconnaître et les modèles de

classe (voir section 1.2.3.1). L'analyse de ces distances permet d'appliquer une règle de décision permettant un rejet de distance ; par exemple «rejet lorsque toutes les distances de la forme à reconnaître aux modèles de classes sont supérieures à un seuil T». Dans [Prevost 03, Arlandis 02], un seuillage est ainsi appliqué sur les distances des formes à classer aux modèles de classes pour traiter le rejet de distance dans le cadre de classification de caractères manuscrits. Signalons toutefois que l'estimation des densités de probabilité ou des modèles de classes demande un nombre d'exemples qui croit exponentiellement avec le nombre de caractéristiques du problème, ce qui rend ce type de classifieurs extrêmement sensible au problème de la dimensionalité.

Le concept de classe modulaire

Dans [Takahashi 03], les auteurs proposent un système basé sur un ensemble de classifieurs GLVQ (Generalized Learning Vector Quantization) avec apprentissage du rejet pour la reconnaissance de chiffres manuscrits. Plutôt que de considérer une classe de formes non numériques, les auteurs décomposent le problème de classification à 10 classes + rejet en 10 problèmes de classification à 2 classes. Chaque classifieur discrimine donc une classe de chiffre du rejet composé des 9 classes de chiffre restantes et des formes non numériques. Les 10 sorties des classifieurs sont alors combinées pour fournir le résultat final. Ce principe qui consiste à décomposer un problème à K classes en K problèmes à 2 classes est appelé «class-modular».

Dans [Takahashi 03], la méthode de combinaison consiste à choisir la classe i si le score de confiance est supérieur à un seuil et si tous les autres sont inférieurs à un autre seuil. Si aucune des classes de chiffre ne vérifie cette propriété, la forme est considérée comme rejet. Dans [Oh 02], la classe produisant la confiance la plus élevée est choisie.

Les résultats montrent que la courbe erreur/rejet obtenue avec 10 classifieurs à 2 classes est proche de la courbe obtenue avec un seul classifieur à 10 classes. En revanche, la méthode proposée permet d'améliorer considérablement la courbe faux rejet - fausse acceptation (courbe ROC). Les expériences de [Kapp 04] ont même montré que le concept «class-modular» permettait d'augmenter le taux de reconnaissance, tout en améliorant les capacités de rejet du classifieur. Dans [Oh 02], aucune règle de rejet n'est mise en œuvre, mais le taux de reconnaissance est supérieur en utilisant le concept «class-modular». Plus le nombre de classes est élevé, plus l'amélioration est importante.

One class classifiers

Le principe des classifieurs à une classe est de modéliser une classe de manière à rejeter les autres éléments. Contrairement aux classifieurs à deux classes, la classe rejet, par définition mal délimitée, n'est pas modélisée : aucun exemple de rejet n'est fourni au classifieur lors de l'apprentissage. Dans [Tax 01], les auteurs présentent plusieurs classifieurs à une classe : des estimateurs de densité (densités modélisées par une loi normale, mixture de gaussiennes, estimation des densités par la méthode de Parzen), et des classifieurs basés sur un modèle de la classe avec calcul d'une

distance (machine à vecteur de support, k-means, k-centre, réseau de neurones «autoencodeur»). Ces classifieurs sont entraînés sur des exemples de la classe à modéliser (target) sans exemples de rejet (outliers), et sont testés sur des exemples de la classe ainsi que sur du rejet avec plusieurs jeux de caractéristiques (profils, Fourier, morphologiques, pixel, moments de Zernike, etc.). Chaque classe de chiffre est modélisée, et les chiffres des autres classes constituent les rejets pendant la phase de reconnaissance. La méthode de Parzen donne les meilleurs résultats, et d'une manière générale les classifieurs qui estiment les densités de probabilités se comportent mieux que les classifieurs par distance. Ce type d'approche basé sur les «One Class Classifiers» (OCC) est intéressant puisqu'il ne nécessite pas d'exemples de rejet lors de l'apprentissage.

5.2.3 Combinaison des approches

Comme nous venons de le voir, les classifieurs discriminants sont souvent utilisés pour traiter le rejet d'ambiguïté, alors que les classifieurs modélisants permettent davantage le rejet de distance. Afin de traiter efficacement les deux types de rejets, de nombreux travaux ont cherché à combiner classifieurs modélisants et discriminants [Milgram 04, Prevost 03, Landgrebe 05]. Il s'agit généralement d'une combinaison séquentielle des approches où un classifieur modélisant traitant les rejets de distance est suivi d'un classifieur discriminant chargé de reconnaître les classes connues et d'identifier le rejet d'ambiguïté.

- Dans [Milgram 04], une combinaison séquentielle est mise en œuvre pour la discrimination de chiffres manuscrits avec gestion du rejet. La première étape est constituée d'une approche modélisante par hyperplan qui fournit des mesures d'appartenance d'un point aux différentes classes. Lorsqu'un conflit est détecté, un classifieur SVM est chargé de lever les ambiguïtés.
- Dans [Landgrebe 05], un système de reconnaissance basé sur la combinaison séquentielle d'un «détecteur» et d'un «classifieur» est présenté. Le premier niveau est un classifieur à une classe chargé d'identifier les formes à classer parmi un ensemble contenant des formes à rejeter ; le deuxième niveau est un classifieur multiclasse discriminant les formes qui ont été identifiées comme valides au premier niveau.
- Le même type de combinaison séquentielle est effectuée dans [Prevost 03] pour la reconnaisance de caractères manuscrits dans un problème à 62 classes (lettres minuscules, majuscules et chiffres).

Certains travaux ont également cherché à effectuer une combinaison séquentielle de classifieurs pour traiter les différents types de rejet, mais en utilisant des classifieurs discriminants pour la première phase de classification [Vuurpijl 03, Bellili 03]. Dans [Vuurpijl 03], l'analyse des scores d'un MLP appliqué en première phase permet de détecter les rejets de distance, puis un ensemble de classifieur SVM est appliqué pour effectuer la reconnaissance. On évite ainsi la difficile modélisation des classes en hautes dimensions lors de la première phase.

Comme l'a montré cette étude bibliographique, la combinaison séquentielle de classifieur semble une approche séduisante pour le problème de la gestion du rejet puisqu'elle permet de gérer relativement facilement les deux types de rejet. Nous avons ainsi choisi de mettre en œuvre une telle stratégie pour nos deux chaînes de traitement.

5.3 Une stratégie de rejet en deux étapes

Dans cette section, nous proposons une stratégie pour l'amélioration des capacités de rejet de nos deux approches. Rappelons que pour la première approche, le rejet consiste à rejeter tout ce qui n'appartient pas à l'une des dix classes de chiffres isolés (chiffres liés compris), alors que dans la deuxième approche, les chiffres liés et les séparateurs ne doivent pas être rejetés. Afin de visualiser le problème du rejet, observons une base de chiffres isolés et d'éléments à rejeter pour la première et la seconde approche (voir figures 5.2 et 5.3).

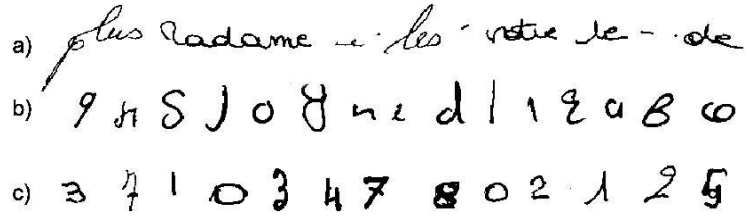


FIG. 5.2 – Exemples de formes rejets et chiffres dans la stratégie mise en œuvre au chapitre 3. La première ligne contient des rejets *évidents*, la dernière ligne contient des chiffres, et la ligne du milieu contient des rejets pouvant être qualifiés *d'ambigüs*.

Parmi les composantes à rejeter dans la première approche, on peut considérer deux catégories :

- Les rejets *évidents* possédant une forme très différente des chiffres isolés : bruit, fragments de mots ou mots entiers, traits, points, etc. (voir figure 5.2 a).
- Les rejets que nous pouvons qualifier d'*ambigüs* ont une forme proche des chiffres isolés. Il s'agit principalement de lettres, groupes de lettres ou fragments de lettres (voir figure 5.2 b). Cette deuxième catégorie est naturellement plus difficile à distinguer des chiffres (voir figure 5.2 c).

En ce qui concerne le rejet des entités pour la deuxième approche, le problème est similaire, excepté pour les composantes de type ponctuation et certains fragments de mots qui peuvent être confondus respectivement avec des séparateurs ou des chiffres liés (figure 5.3 b). Ces composantes deviennent par conséquent des rejets ambigüs, et les formes «rejets évidents» sont *a priori* moins nombreuses que pour la première approche.

Dans la suite de cette section, nous utiliserons le terme générique de «forme numérique» pour désigner les chiffres pour la première approche, et les chiffres,

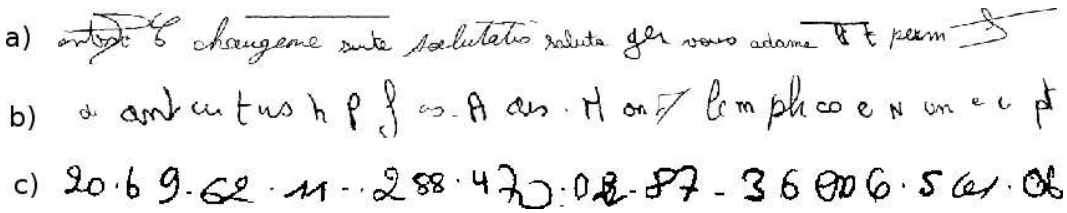


FIG. 5.3 – Exemples de formes rejets et chiffres dans la stratégie mise en œuvre au chapitre 4.

les séparateurs et les chiffres liés pour la seconde approche. De la même manière, nous appelerons «rejets» l'ensemble des formes non-chiffre contenant ou non les séparateurs et chiffres liés suivant que l'on se trouve dans la première ou la seconde approche.

En nous replaçant par rapport à l'étude bibliographique, les rejets dits «évidents» relèveraient plutôt du rejet de distance (les formes sont éloignées des classes de formes numériques), alors que les rejets ambigüs doivent être traités comme des rejets d'ambiguïté (les formes étant assez proches des classes de formes numériques, elles sont *a priori* proches des frontières de décision). Partant de ce constat, nous avons mis en place une stratégie séquentielle en deux étapes pour rejeter les formes non numériques.

Les phases de rejet existantes dans nos deux chaînes de traitement permettant plutôt de traiter le rejet d'ambiguïté, nous proposons d'ajouter une première étape qui vise à différencier les «rejets évidents» des formes numériques. Agissant comme un filtre, ce module produit une règle de décision binaire : acceptation ou rejet de la forme. Dans le cas d'un rejet, celui-ci est définitif puisque la confiance de la classe rejet est placée à 1, et celles des 10 classes de chiffre à 0, afin de verrouiller un certain nombre de décisions dans le treillis des hypothèses de reconnaissance. L'idée est donc de filtrer un maximum de rejet, tout en ne rejetant aucun chiffre (voir figure 5.4).

Afin de discriminer au mieux les formes numériques des rejets, nous utilisons un classifieur SVM connu pour ses performances très intéressantes (voir section 1.2.3.2). Nous décrivons maintenant la conception et l'apprentissage de cette étape de filtrage.

5.4 Filtrage des rejets évidents

Nous décrivons ici la réalisation du système de filtrage des rejets évidents par un classifieur SVM et un jeu de caractéristiques réduit. La stratégie pour le filtrage des rejets évidents est basée sur l'utilisation d'un classifieur SVM dont nous connaissons les bonnes capacités de discrimination [Vapnik 95, Liu 02a, Liu 04]. Nous abordons l'apprentissage de ce SVM comme un problème à deux classes : Digit/Rejet dans le cadre de la première approche ; Digit+Séparateur+DoubleDigit/Rejet dans le cadre

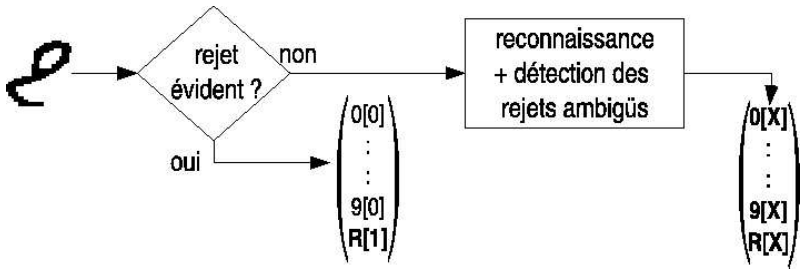


FIG. 5.4 – Une stratégie de rejet en deux étapes. La première étape élimine les rejets «évidents» de manière binaire, alors que la deuxième étape prend une décision plus douce pour discriminer les chiffres des rejets «ambigüs».

de la seconde. L'apprentissage d'un tel classifieur n'est toutefois pas évident dans le contexte de notre application car les coûts de mauvaise classification sont inconnus. En effet, rappelons que si une composante à rejeter reconnue comme un chiffre par le classifieur peut être rejeté lors de la seconde étape, le rejet d'une forme numérique est définitif et irrattrapable par le système. Dans ce cas, plusieurs cas de figure peuvent se produire :

- La forme numérique rejetée n'appartient pas à un champ numérique et le rejet ne porte pas à conséquence.
- La forme numérique rejetée appartient à un champ numérique et son rejet empêche la bonne localisation et la reconnaissance du champ.
- La forme numérique rejetée appartient à un champ numérique qui était déjà mal localisé et/ou mal reconnu. Son rejet n'aggrave donc pas la situation.

On constate ainsi que les coûts de mauvaise classification sont inconnus puisqu'il est impossible de mesurer l'incidence d'une *forme numérique rejetée* par rapport à l'incidence d'un *rejet accepté* sur les performances globales du système. Le problème d'apprentissage du classifieur SVM qui se pose comporte donc des coûts de mauvaise classification déséquilibrés *et* inconnus.

Ces coûts déséquilibrés et inconnus complexifient le problème du réglage des hyperparamètres d'un classifieur SVM : le paramètre de régularisation C et au moins un paramètre de noyau (classiquement le γ pour les SVM à noyaux gaussiens). La détermination de ces hyperparamètres est appelée «sélection de modèle» dans la littérature, et influe beaucoup sur les performances du classifieur [Ayat 05].

La sélection de modèle est souvent effectuée par des algorithmes «full search» en discrétisant l'espace des paramètres et en évaluant toutes les combinaisons possibles. Il a été montré que ce type d'approche était très gourmand en temps de calcul et fonctionnait mal [Hsu 02b, Lavalle 02].

Plus récemment, la détermination des hyperparamètres a été considérée comme une tâche d'optimisation. Un algorithme d'optimisation est alors mis en œuvre afin de trouver l'ensemble des hyperparamètres apportant les meilleures performances. Les méthodes de descente de gradient ont ainsi été appliquées pour l'optimisation des

hyperparamètres de classifieurs SVM [Chapelle 02, Chung 03, Gold 03, Keerthi 02]. Cependant, les méthodes à gradient imposent une dérivabilité du critère de performance et du noyau SVM par rapport aux hyperparamètres. De plus, il est connu que les résultats des méthodes de descente de gradient dépendent de leur initialisation et peuvent tomber dans des *minima* locaux.

Les algorithmes évolutionnaires ont ainsi été employés pour solutionner ces problèmes puisqu'ils ne nécessitent pas la dérivabilité du critère de performance. On peut citer les travaux présentés dans [Huang 06] et [Wu 06] basés sur des algorithmes génétiques, ou [Friedrichs 05] qui utilise une stratégie évolutionnaire. Dans les deux cas, l'algorithme d'optimisation est utilisé pour optimiser C et γ relativement à un critère de performance tel que le taux de bonne classification.

Dans tous ces algorithmes d'optimisation, un critère de performance unique est utilisé, ce qui peut être très réducteur, particulièrement dans le cas où les données sont mal balancées (la base d'apprentissage et/ou de test contient des effectifs par classe disproportionnés) ou quand le problème de classification comporte des coûts de mauvaise classification déséquilibrés. Ces cas de figure sont très fréquents dans les problèmes réels, et on doit alors prendre en compte les probabilités *a priori* et les coûts de mauvaise classification dans le calcul du critère de performance. Dans le contexte de notre étude, nous avons vu que ces coûts de mauvaise classification sont difficiles à estimer. Dans ce contexte, la courbe ROC (Receiver Operating Characteristics [Bradley 97]) est un meilleur indicateur de performance. Elle représente le compromis entre les taux de faux rejet (FR) et de fausse acceptation (FA), aussi appelés respectivement sensitivité et spécificité. Ainsi dans le cas d'un problème de classification à deux classes, deux critères doivent être minimisés à la place d'un unique taux de bonne classification.

La sélection de modèle d'un SVM pour notre problème de discrimination forme numérique/rejet peut donc être vue comme un problème d'optimisation multiobjectifs. Nous proposons ainsi d'appliquer un algorithme évolutionnaire multiobjectifs pour optimiser les hyperparamètres du SVM relativement aux deux critères FA et FR. Une telle stratégie permet d'obtenir à chaque itération un ensemble de classifieurs distincts et optimaux du point de vue des deux critères. L'ensemble de ces classifieurs couvre ainsi un large éventail de compromis FA/FR optimaux. Une fois l'apprentissage évolutionnaire effectué, il sera alors possible de choisir parmi l'ensemble de classifieurs produisant un compromis FA/FR optimal celui qui donnera les meilleurs résultats pour l'application de localisation et de reconnaissance de champs numériques.

Dans la suite de cette partie, nous présentons une introduction aux SVM et à leur hyperparamètres avant de discuter du choix des critères de performance à optimiser (section 5.4.1). Dans la section 5.4.2 nous dressons un panorama des méthodes d'optimisation multiobjectif évolutionnaires, décrivons l'algorithme choisi et son application au problème d'optimisation des hyperparamètres d'un SVM. Enfin les résultats expérimentaux sont présentés dans la section 5.4.4.

5.4.1 Description du problème

5.4.1.1 Classifieurs SVM et leurs hyperparamètres

Comme définis dans [Osuna 97], les SVM peuvent prendre en compte des coûts de mauvaise classification déséquilibrés par l'intermédiaire de deux paramètres distincts de pénalité : C_- et C_+. Dans ce cas, pour un ensemble de m exemples d'apprentissage x_i dans $\Re^n$ appartenant à la classe u_i :

$$(x_1, u_1) \dots (x_m, u_m), x_i \in \Re^n, u_i \in \{-1, +1\}$$

la maximisation du lagrangien dual par rapport aux α_i devient :

$$Max_\alpha \Big\{ \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j u_i u_j K(x_i, x_j) \Big\}$$

$$\text{sous les contraintes : } \begin{cases} 0 \leq \alpha_i \leq C_+ \quad pour \quad u_i = -1 \\ 0 \leq \alpha_i \leq C_- \quad pour \quad u_i = +1 \\ \sum_{i=1}^{m} \alpha_i u_i = 0 \end{cases}$$

où α_i désigne les multiplicateurs de Lagrange et $K(.)$ le noyau. Dans le cas d'un noyau gaussien (RBF), $K(.)$ est défini de la manière suivante :

$$K(x_i, x_j) = exp\left(-\gamma \times \|x_i - x_j\|^2\right)$$

Ainsi, dans le cas de coûts de mauvaise classification différents, trois paramètres doivent être déterminés :

– Le paramètre du noyau RBF : γ.
– Les paramètres de pénalité introduits ci-dessus : C_- and C_+.

Dans la suite de cette partie, un *ensemble d'hyperparamètres* désigne donc un ensemble particulier de trois valeurs γ, C_- et C_+.

5.4.1.2 Critères pour la détermination des hyperparamètres

Considérer la sélection de modèle comme un problème d'optimisation sous-entend le choix d'un ou de plusieurs critères à optimiser. Comme nous l'avons vu précédemment, la courbe ROC donne un meilleur indicateur de performance qu'un taux de bonne classification dans le cas d'un problème à deux classes avec des coûts de mauvaise classification déséquilibrés. L'utilisation de la courbe ROC implique l'optimisation de deux critères (FA et FR), ce qui est un problème plus complexe que l'optimisation d'un seul critère.

Plusieurs approches ont été proposées dans la littérature pour optimiser la courbe ROC en réglant les paramètres intrinsèques d'un classifieur. Ce type d'approche est généralement basé sur la réduction des deux critères FA et FR en un seul, tel que l'aire sous la courbe ROC (AUC : Area Under the ROC Curve) ou la F-mesure

(FM). C'est le cas des travaux présentés par l'un des membres de notre équipe dans [Rakotomamonjy 04], où un critère d'AUC est utilisé pour entrainer un classifieur SVM. Dans ces travaux, les supports vecteurs et les α associés sont déterminés par minimisation du critère AUC. Cette approche ayant donné de bons résultats, nous les comparons avec les nôtres dans la partie 5.4.4, et referrons à [Rakotomamonjy 04] pour les détails concernant le calcul de l'AUC et le processus d'optimisation de la méthode. Signalons que les approches reposant sur un critère AUC ont également été proposées dans [Ferri 02] et [Mozer 02] dans le cas d'autres classifieurs, et qu'une approche similaire basée sur la F-mesure est proposée dans [Musicant 03].

Dans tous ces travaux, le but est de réaliser un classifieur optimal au sens du critère de performance choisi (AUC ou FM). Cependant, ces critères de performance ne sont que des indicateurs réducteurs de la courbe ROC. Ainsi, pour une valeur de FA donnée (respectivement FR), les classifieurs entrainés avec ce type de critère ne sont pas capables de produire le classifieur avec la valeur de FR optimale (respectivement FA). Ce qui signifie qu'un classifieur optimisant l'aire sous la courbe ROC ne garantie pas d'être le classifieur optimal pour une valeur donnée de FA (respectivement FR). Cette remarque est illustrée sur la figure 5.5.

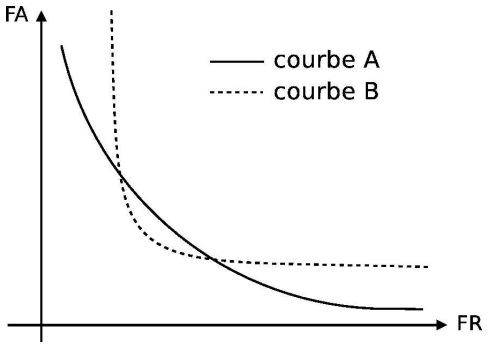


FIG. 5.5 – La courbe A minimise l'aire sous la courbe ROC mais en certains points la courbe B donne un meilleur compromis FA/FR.

La méthode que nous proposons ici est basée sur l'optimisation de la courbe ROC par la recherche d'un ensemble de classifieurs optimaux au sens des deux paramètres FA et FR, à l'aide d'une véritable optimisation multicritère. Cela implique la mise en œuvre d'un algorithme d'optimisation multiobjectif pour la recherche des ensembles d'hyperparamètres, chaque ensemble d'hyperparamètres optimisant un compromis FA/FR. La dimension de l'espace des objectifs étant supérieur à 1, le concept de dominance employé dans le domaine de l'optimisation multiobjectif doit être introduit pour comparer les performances de deux classifieurs.

5.4.1.3 Le concept de dominance de Pareto pour la sélection de modèles SVM

Le concept de dominance a été proposé par Vilfredo Pareto au 19ème siècle. On dit qu'un vecteur $\overrightarrow{u}$ (dans notre cas, un ensemble donné (C_+, C_-, γ))) domine un autre vecteur $\overrightarrow{v}$ si $\overrightarrow{u}$ n'est pas pire que $\overrightarrow{v}$ pour n'importe lequel des objectifs (FA et FR) et si $\overrightarrow{u}$ est meilleur que $\overrightarrow{v}$ pour au moins un objectif. La notation est la suivante : $\overrightarrow{u} \prec \overrightarrow{v}$. Plus formellement, un vecteur $\overrightarrow{u} = (u_1, u_2, \ldots, u_k)$ domine un vecteur $\overrightarrow{v} = (v_1, v_2, \ldots, v_k)$ si et seulement si :

$$\forall i \in \{1, \ldots, k\}, u_i \leq v_i \wedge \exists j \in \{1, \ldots, k\} : u_j < v_j$$

Étant donné le concept de dominance, l'objectif d'un algorithme d'optimisation multiobjectif est de chercher l'*ensemble de Pareto*, défini comme l'ensemble des solutions dans l'espace des paramètres engendrant des solutions non dominées dans l'espace des objectifs :

$$\text{Ensemble de Pareto} = \left\{ \overrightarrow{u} \in \vartheta / \neg \exists \overrightarrow{v} \in \vartheta, \overrightarrow{f(v)} \prec \overrightarrow{f(u)} \right\}$$

où ϑ désigne l'espace des paramètres où les contraintes sont satisfaites, et $\overrightarrow{f}$ désigne le vecteur d'objectifs. Du point de vue de la sélection de modèle SVM, l'ensemble de Pareto correspond à l'ensemble optimal de vecteurs d'hyperparamètres produisant tous les compromis FA/FR optimaux. Dans l'espace des objectifs, cet ensemble de compromis optimaux est appelé *front de Pareto*. Remarquons que dans le cadre de la sélection de modèles SVM, le front de Pareto pourrait être comparé à la courbe ROC qui décrirait le meilleur ensemble de compromis FA/FR. Dans notre cas, le front de Pareto correspond toutefois aux compromis FA/FR obtenus à l'aide d'un ensemble de classifieurs, alors que la courbe ROC est obtenue à l'aide d'un seul classifieur. Nous discutons de la relation entre le front de Pareto et la courbe ROC dans la section 5.4.4.

L'approche proposée cherche donc à approximer l'ensemble optimal de Pareto d'un classifieur SVM à deux classes à l'aide d'une optimisation multiobjectif évolutionnaire. Nous dressons maitenant un bref panorama des méthodes d'optimisation multiobjectif évolutionnaire, et décrivons l'algorithme choisi ainsi que son application à la sélection de modèles SVM.

5.4.2 Optimisation multiobjectif évolutionnaire

Nous recherchons l'ensemble de classifieurs SVM décrivant l'ensemble des compromis FA/FR optimaux. Les classifieurs sont paramétrés par les hyperparamètres (C_+, C_-, γ). Du point de vue de l'optimisation multiobjectif, cet ensemble peut être vu comme un ensemble de Pareto. L'ensemble des compromis FA/FR associés à ces classifieurs forme le front que nous recherchons. Les algorithmes évolutionnaires sont bien adaptés à la recherche de ce front car ils sont capables grâce à leur parallélisme implicite de dégager des solutions optimales en une seule itération.

5.4.2.1 Panorama des approches existantes

Depuis les premiers travaux de [Schaffer 85] au milieu des années 80, un certain nombre d'approches d'optimisation multiobjectif évolutionnaire a été proposé : MOGA [Fonseca 93], NSGA [Srinivas 94], NPGA [Horn 94], SPEA [Zitzler 99], NSGA II [Deb 00], PESA [Corne 00] ou encore SPEA2 [Zitzler 01]. Dans une étude comparative, [Khare 02] compare les performances des trois algorithmes les plus populaires : SPEA2, PESA et NSGA-II. Ces trois approches sont élitistes, c'est-à-dire que toutes les solutions non dominées trouvées sont sauvegardées dans une archive afin d'assurer la préservation de bonnes solutions. Cette étude comparative a été menée sur différents problèmes, avec pour mesure de qualité les deux critères importants pour un algorithme multiobjectif évolutionnaire : se rapprocher le plus possible du front de Pareto et obtenir une bonne dispersion des solutions sur ce front. Les résultats de cette étude (qui ont été confirmés dans [Zitzler 01] et [Bui 04]) montrent qu'aucun des algorithmes ne dominait les autres au sens de Pareto. SPEA2 et NSGA-II offrent des performances similaires en terme de convergence et de diversité. Leur convergence est inférieure à celle de PESA mais la diversité des solutions est meilleure. L'étude montre également que NSGA-II est plus rapide que SPEA2.

Dans le contexte de la sélection de modèles SVM, le calcul des fonctions objectifs prend beaucoup de temps puisqu'il faut entrainer puis évaluer le classifieur pour chaque ensemble d'hyperparamètres. De plus, une bonne diversité des solutions est nécessaire puisqu'on ne connait pas le point de fonctionnement sur le front de Pareto. Nous avons donc choisi l'algorithme NSGA-II. Nous donnons dans la partie suivante une description de cet algorithme.

5.4.2.2 NSGA-II

NSGA-II est une version modifiée de l'algorithme NSGA [Srinivas 94]. C'est une approche rapide, élitiste et sans paramètres qui manipule une population de solutions et utilise un mécanisme de préservation de la diversité explicite.

Initialement, une population parent P_0 de N solutions (ou individus) est créée aléatoirement. Cette population est triée sur une base de non-dominance à l'aide d'un algorithme rapide. Ce tri associe un rang de dominance à chaque individu. Les individus non dominés ont un rang de 1 et constituent le front $\mathcal{F}_1$. Les autres fronts $\mathcal{F}_i$ sont ensuite définis récursivement en ignorant les solutions les moins bien classées. Ce tri est illustré sur la figure 5.6 (à gauche) dans le cas d'un problème à deux objectifs (f_1,f_2).

À l'aide des résultats de la procédure de tri, on associe à chaque individu une mesure correspondant à son degré de non-dominance. Les opérateurs de croisement, de recombinaison et de mutation (voir [Goldberg 89] et [Deb 00] pour plus de détails) sont ensuite utilisés pour créer une population fille Q_0 de même taille que P_0. À l'issue de cette première étape, l'algorithme 2 est itéré durant M générations. À chaque itération, t désigne le numéro de génération courante, $\mathcal{F}$ désigne le résultat de la procédure de tri, $\mathcal{F}_i$ désigne le i^{eme} front de $\mathcal{F}$, P_t et Q_t désignent respectivement la

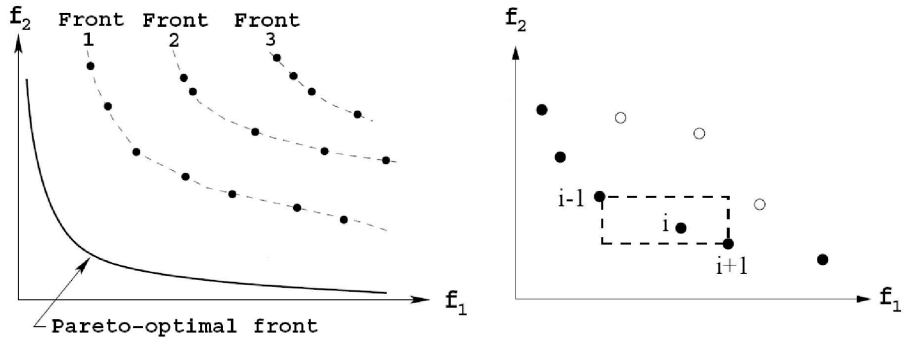


FIG. 5.6 – Illustration du concept $\mathcal{F}_i$. Les points noirs sont les vecteurs dominants, les points blancs sont dominés.

population et la progéniture à la génération t, et R_t est une population temporaire.

Algorithme 2 Algorithme NSGA-II

$t \leftarrow 0$
tant que $t < M$ **faire**
  $R_t \leftarrow P_t \cup Q_t$
  $\mathcal{F} \leftarrow$ tri-selon-non-dominance (R_t)
  $P_{t+1} \leftarrow \emptyset$
  $i \leftarrow 0$
  tant que $|P_{t+1}| + |\mathcal{F}_i| \geq N$ **faire**
    $P_{t+1} \leftarrow P_{t+1} \cup \mathcal{F}_i$
    assigner-crowding-distance $(\mathcal{F}_i)$
    $i \leftarrow i + 1$
  fin tant que
  Trier $(\mathcal{F}_i, \prec_n)$
  $P_{t+1} \leftarrow P_{t+1} \cup \mathcal{F}_i[1 : (N - |P_{t+1}|)]$
  $Q_{t+1} \leftarrow$ creer-nouvelle-population (P_{t+1})
  $t \leftarrow t + 1$
fin tant que

Remarquons que la première itération de l'algorithme débute avec une fusion des populations parent P_t et fille Q_t pour construire R_t. Cette population de $2N$ solutions est triée à l'aide de la procédure de tri de non-dominance pour construire la population P_{t+1}. Durant cette étape, un autre critère de tri est mis en œuvre pour conserver l'effectif de P_{t+1} à une taille constante durant l'intégration des $\mathcal{F}_i$ successifs. Son but est de prendre en compte la contribution des solutions pour la diversité et la dispersion de la fonction objectif dans la population. Le tri des individus est effectué selon une mesure de dispersion appelée *crowding distance* [Deb 00]. Cette mesure est basée sur le calcul de la distance moyenne aux deux points de part et

d'autre de l'individu considéré selon les deux objectifs (voir figure 5.6 droite). Plus la surface autour de l'individu considéré est grande, plus la solution est bonne du point de vue de la diversité. Les solutions de R_t contribuant le plus à la diversité sont ainsi favorisées dans la construction de P_{t+1}. Cette étape est désignée dans l'algorithme 5.4.2.2 par : trier($\mathcal{F}_i$,$\prec_n$), où $\prec_n$ designe une relation d'ordre partiel basée à la fois sur la dominance et sur la *crowding distance*. Selon cette relation, une solution i est meilleure qu'une solution j si ($i_{rank} < j_{rank}$) ou si ($i_{rank} = j_{rank}$) et ($i_{distance} > j_{distance}$).

Grâce à cet algorithme, la population P_t converge nécessairement vers un ensemble de points du front de Pareto puisque les solutions non dominées sont préservées à travers les générations. De plus, le critère de dispersion (crowding distance) garantit une bonne diversité dans la population [Deb 00].

5.4.2.3 Application de NSGA-II à la sélection de modèles SVM

Dans cette section, nous présentons l'application de l'algorithme NSGA-II au problème de sélection de modèle SVM. Pour cela, deux points particuliers doivent être précisés :

- **Le codage des individus** : rappelons que trois paramètres sont impliqués dans l'apprentissage des classifieurs SVM avec des coûts de mauvaise classification déséquilibrés : C_+, C_- et γ. Ces trois paramètres constituent l'espace des paramètres de notre problème d'optimisation. Chaque individu de la population doit donc coder ces trois valeurs réelles. Nous avons choisi un codage réel des paramètres afin d'être le plus précis possible.
- **La procédure d'évaluation** : chaque individu de la population correspond à un ensemble de trois hyperparamètres. Afin d'évaluer la qualité de cet individu, un apprentissage SVM classique piloté par l'ensemble d'hyperparamètres encodé est lancé. Ce classifieur SVM est ensuite évalué sur une base de test à l'aide des critères FA et FR.

5.4.3 Caractérisation du rejet

Pour mener à bien cette tâche de classification, nous avons développé un vecteur de caractéristiques spécifique, adapté à la discrimination de formes numériques/non numériques. Ce vecteur est constitué des 8 caractéristiques suivantes dont chacune est dédiée à la détection d'un certain type de rejets :

- (f_1) ratio hauteur/largeur : les chiffres et chiffres liés n'étant souvent pas très alongés, cette caractéristique permet la détection des formes alongées (traits, mots longs).
- (f_2) densité de pixels noirs : permet de rejeter les composantes comportant une trop forte ou une trop faible densité de pixels noirs (rature, bruit, signature).
- (f_3) nombre de *water reservoirs* (voir page 97 pour plus de détails concernant les water reservoirs) : les chiffres ne contenant généralement pas plus de 1 ou 2 water reservoirs, cette caractéristique est dédiée à la détection des composantes

complexes (mots ou groupement de lettres).
- (f_{4-6}) nombre d'intersections avec deux sondes horizontales et une sonde verticale : ces caractéristiques permettent la détection des composantes complexes générant de nombreuses intersections telles que les mots.
- (f_7) nombre de fin de traits : les chiffres possèdent un nombre de fin de trait limité ; cette caractéristique est donc destinée à détecter les composantes complexes.
- (f_8) nombre d'occlusions : le nombre maximum d'occlusions dans les composantes numériques étant relativement réduit, cette caractéristique est pertinente pour la détection de groupements de lettres ou de mots contenant de nombreuses boucles.

L'observation de la disposition des composantes connexes d'une page de document nous a également conduit à ajouter un certain nombre de caractéristiques extraites des boîtes englobantes des composantes. En effet, comme on peut le constater sur la figure 5.7, la disposition des boîtes englobantes des composantes numériques respecte une certaine régularité que l'on ne retrouve pas dans les composantes à rejeter. Nous avons donc choisi d'extraire un ensemble supplémentaire de 8 caractéristiques «contextuelles» permettant de traduire cette régularité. Il s'agit des même caractéristiques que pour l'identification des séparateurs (mis à part le ratio hauteur/largeur déjà présent ; voir partie 3.6.2) que nous rappelons : si C désigne la composante considérée, C_{-1} et C_{+1} ses voisines gauche et droite, H_C, W_C, G_{C_x} et G_{C_y} respectivement les hauteur, largeur, l'abscisse et l'ordonnée du centre de gravité de la composante C, les 8 caractéristiques permettant d'identifier la régularité/irrégularité dans la taille et le positionnement des composantes sont :

$(f_9) = \frac{H_{C_{-1}}}{H_C}$; $(f_{10}) = \frac{H_{C_{+1}}}{H_C}$; $(f_{11}) = \frac{W_{C_{-1}}}{W_C}$; $(f_{12}) = \frac{W_{C_{+1}}}{W_C}$; $(f_{13}) = \frac{G_{C_X} - G_{C_{X-1}}}{W_C}$; $(f_{14}) = \frac{G_{C_X} - G_{C_{X+1}}}{W_C}$; $(f_{15}) = \frac{G_{C_Y} - G_{C_{Y-1}}}{H_C}$; $(f_{16}) = \frac{G_{C_Y} - G_{C_{Y+1}}}{H_C}$

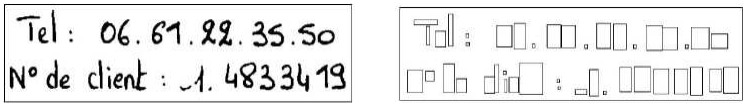


FIG. 5.7 – Boîtes englobantes de composantes textuelles et numériques non segmentées. Remarquons la régularité dans la taille et le positionnement des boîtes englobantes constituant un champ numérique.

Ces caractéristiques ne peuvent cependant pas être utilisées pour la première chaîne de traitement puisque la segmentation des composantes «casse» la régularité des composantes. Au final, un vecteur de 8 ou 16 caractéristiques est donc utilisé pour le filtrage des formes non numériques suivant que l'on se situe dans la première ou la seconde chaîne de traitement.

5.4.4 Résultats

Nous présentons dans cette section les résultats expérimentaux obtenus à l'aide de l'approche proposée, sur les problèmes de discrimination Digits *vs* Rejets et Digits + Séparateurs + DoubleDigits *vs* Rejets. Les deux systèmes basés sur NSGA-II sont entrainés sur une base d'apprentissage de 7129 formes contenant 1/3 de chiffres et 2/3 de rejets, testés sur une base de test de 7149 formes et évalués sur une base de validation de 5000 formes[1].

Les plages de valeurs des hyperparamètres sont données dans la table 5.1. Une précision de 10^{-6} est utilisée pour ces paramètres. En ce qui concerne les paramètres de NSGA-II, nous avons employé les valeurs classiques proposées dans [Deb 00]. Parmi celles-là, notons que la taille de la population a été fixée à 40 afin d'obtenir suffisamment de points sur le front de Pareto.

Hyperparamètres	γ	C_-	C_+
Plages de valeurs (précision 10^{-6})	$0-1$	$0-1000$	$0-10000$

TAB. 5.1 – Plages de valeurs pour les hyperparamètres du SVM

Ainsi, le front de Pareto obtenu avec une optimisation multiobjectif peut être vue comme l'enveloppe de toutes les courbes ROC de la population gérée par NSGA-II. Ce concept est illustré sur la figure 5.8. L'obtention de cette enveloppe est intéressante puisqu'elle permet de choisir le classifieur proposant le compromis souhaité. La figure 5.9 donne les enveloppes de courbe ROC pour les deux problèmes de filtrage.

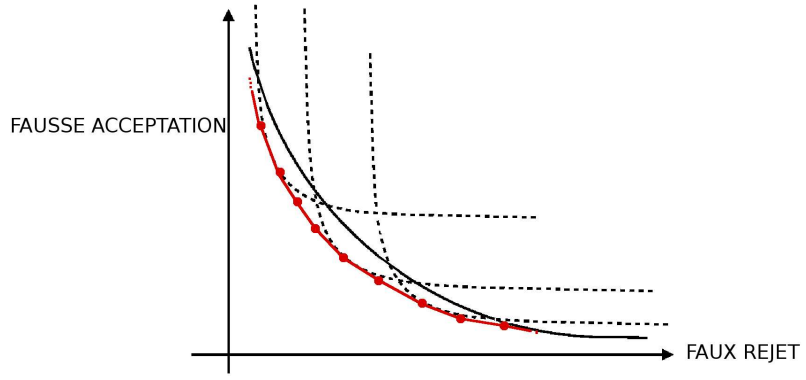


FIG. 5.8 – Exemple d'enveloppe obtenue dans le contexte de la sélection de modèle SVM par un algorithme d'optimisation multiobjectif.

[1]Les trois bases respectent les mêmes proportions de chiffres (1/3) et de rejets (2/3).

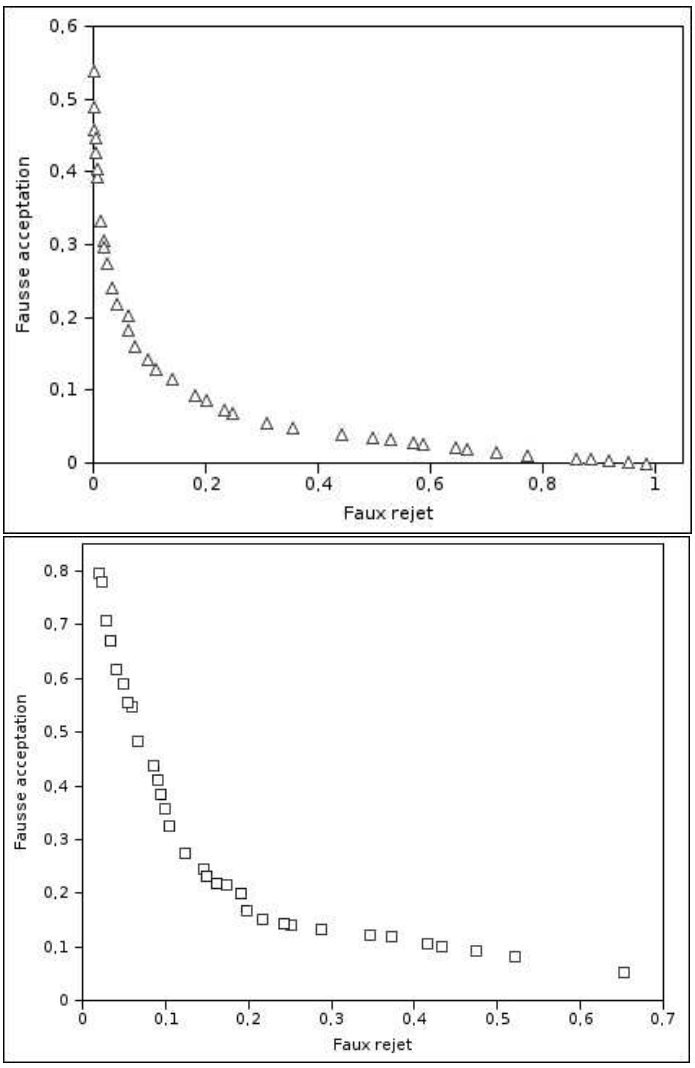


FIG. 5.9 – Enveloppes de courbe ROC pour les deux problèmes de filtrage Digits/Rejets (en haut) et Digits + Séparateur + DoubleDigit/Rejet (en bas).

Pour le premier problème (Digits *vs* Rejets), on peut constater que l'enveloppe de courbe ROC obtenue forme un coude important offrant un «break-even point» (point de fonctionnement où FA=FR) aux alentours de 12%. On atteint un faux rejet nul pour des valeurs de fausse acceptation comprises entre 50 et 60 %. Autrement dit, il est possible de rejeter entre 50 et 60% des formes non pertinentes d'un document en ne rejetant aucun chiffre à l'aide d'un SVM appliqué sur un vecteur réduit de 8 caractéristiques. Remarquons également la bonne dispersion des solutions sur l'ensemble des intervalles FA et FR obtenue grâce à la *crowding distance* de NSGA-II. En ce qui concerne le deuxième problème (Digits + Separateurs +

DoubleDigit *vs* Rejets), le coude est légèrement moins prononcé, et le *break-even point* donne FA=FR=19%. Au vu de ces résultats, il semble qu'un faux rejet nul ne puisse être atteint malgré un nombre de caractéristiques plus important que pour le premier problème (16 caractéristiques, contre 8 pour le premier problème).

Afin de mieux visualiser le comportement de notre approche aux alentours de la zone critique (FR proche de 0), nous avons appliqué des contraintes sur les objectifs telles que le permet un algorithme d'optimisation multiobjectif évolutionnaire. Cette possibilité peut être très utile dans le contexte de notre application puisqu'elle permet de ne considérer qu'une partie de la courbe ROC. En effet, nous sommes particulièrement intéréssés par l'obtention d'un faux rejet le plus faible possible puisque nous savons qu'il a des conséquences *a priori* graves sur la localisation des champs numériques. Pourtant, obtenir un faux rejet nul n'est peut être pas la meilleure solution dans la mesure où un faux rejet nul implique une fausse acceptation importante pour les deux problèmes. En permettant quelques erreurs sur le faux rejet, le taux de fausse acceptation peut baisser considérablement. Nous avons donc choisi de limiter le faux rejet à 3% sur le premier problème afin d'obtenir un compromis FA/FR intéressant. Pour le second problème, le faux rejet n'est limité qu' à 10% car il est plus difficile d'obtenir un faux rejet faible. La figure 5.10 présente les courbes obtenues en limitant FR à 3 et 10%. Remarquons que les résultats obtenus sont similaires à ceux obtenus sans contrainte, mais une plus forte diversité est obtenue sur la zone d'intérêt choisie.

On remarque que pour le premier problème, un faux rejet nul est obtenu pour une fausse acceptation de 55%. Pour le second problème, la restriction de FR à 10% confirme qu'on ne peut obtenir un faux rejet nul autrement qu'avec une fausse acceptation de 100% (absence de filtrage). Ces résultats rappellent la difficulté déjà observée en section 4.2.1.1 du problème de classification lorsque rejets et chiffres liés sont deux classes distinctes.

Afin d'évaluer notre approche, nous comparons nos résultats à ceux de l'algorithme présenté dans [Rakotomamonjy 04], où un classfieur SVM unique est entrainé avec un critère d'aire sous la courbe ROC (AUC). La comparaison est effectuée sur le problème de discrimination chiffres/rejets (voir figure 5.11).

Au vu de ces résultats, remarquons que tous les points obtenus avec le SVM unique entrainé avec un critère AUC sont dominés par au moins un point de la courbe FA/FR obtenue avec NSGA-II. Ce résultat s'explique par le fait qu'avec l'algorithme d'optimisation multiobjectif évolutionnaire, chaque classifieur SVM est spécifique à un point de fonctionnement FA/FR. Le compromis FA/FR est ainsi forcément meilleur que celui obtenu par un classifieur unique entrainé pour optimiser tous les compromis FA/FR possibles.

Conclusion sur la sélection de modèle SVM par NSGA-II

Notons que pour un ensemble d'hyperparamètres donné, les paramètres intrinsèques des classifieurs SVM (les positions et poids des vecteurs de support)

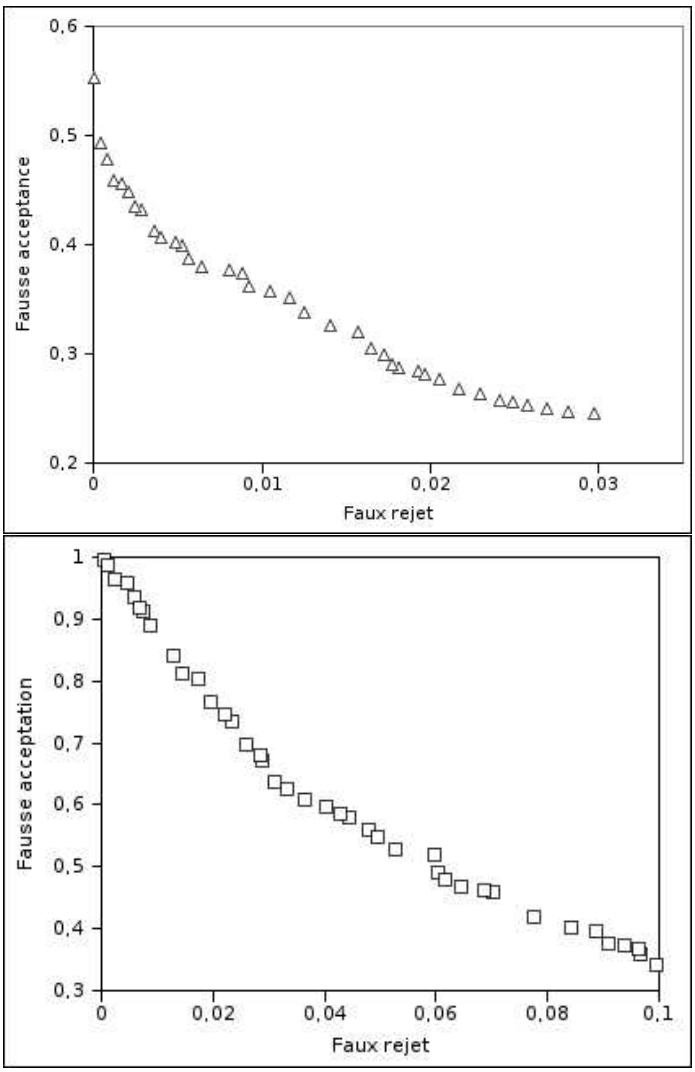


FIG. 5.10 – Enveloppes de courbe ROC pour les deux problèmes de filtrage Digits/Rejets avec contrainte $FR < 3\%$ (en haut) et Digits + Séparateur + Double-Digit *vs* Rejet avec contrainte $FR < 10\%$ (en bas).

sont déterminés à l'aide d'une optimisation mono-objectif adaptée à cette tâche. Ainsi, l'algorithme évolutionnaire se concentre sur le choix des hyperparamètres. Cette approche diffère donc des autres travaux mettant en œuvre des algorithmes évolutionnaires pour régler à la fois les paramètres intrinsèques et les hyperparamètres. Nous pouvons en particulier mentionner les travaux de [Kupinski 99, Anastasio 98, Fieldsend 04, Everson 06]. Tous ces travaux sont limités à des classifieurs très simples (c'est-à-dire possédant un faible nombre de paramètres in-

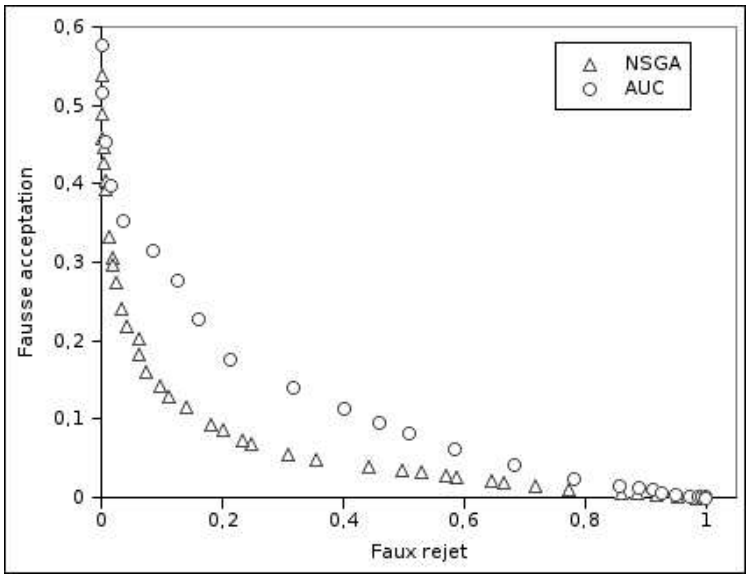


FIG. 5.11 – Courbes FA/FR obtenues par les deux approches : un ensemble de classifieurs SVM avec NSGA, et un classifieur unique entrainé avec un critère AUC.

trinsèques) à cause de l'impossibilité pour un algorithme évolutionnaire de traiter un nombre élevé de paramètres. Dans un contexte mono-objectif, une telle limitation a été contournée en développant des méthodes spécifiques telles que la maximisation du lagrangien pour les SVM ou la rétropropagation du gradient pour les MLP. Dans un contexte multiobjectif, l'utilisation de la maximisation du lagrangien pour le réglage des paramètres intrinsèques couplée à l'algorithme évolutionnaire en charge des hyperparamètres en nombre plus réduit constitue ainsi une solution intéressante.

Nous disposons donc désormais pour nos deux problèmes d'un ensemble de classifieurs SVM possédant les meilleurs compromis FA/FR. Il nous reste donc à faire notre choix parmi cet ensemble de classifieurs pour effectuer la tâche de discrimination formes numériques / non numériques. Comme nous allons le montrer dans la section suivante, nous choisirons simplement le compromis apportant les meilleurs résultats en terme de localisation et de reconnaissance des champs numériques.

5.5 Résultats

Dans cette section, nous montrons l'effet du filtrage des rejets évidents sur les résultats en rappel-précision des deux chaînes de traitement.

5.5.1 Intégration de l'étape de filtrage des rejets dans les deux systèmes

Rappelons qu'à l'issue de l'apprentissage de la méthode de filtrage des rejets évidents, un ensemble de classifieurs SVM produisant chacun un compromis faux rejet/fausse acceptation différent a été généré. Le meilleur compromis étant inconnu, tous les classifieurs doivent être essayés afin de retenir le meilleur au sens de l'application de localisation et de reconaissance des champs.

Premier système

La courbe 5.12 donne les courbes rappel-précision obtenues avec plusieurs classifieurs offrant différents taux de faux rejet pour le premier système.

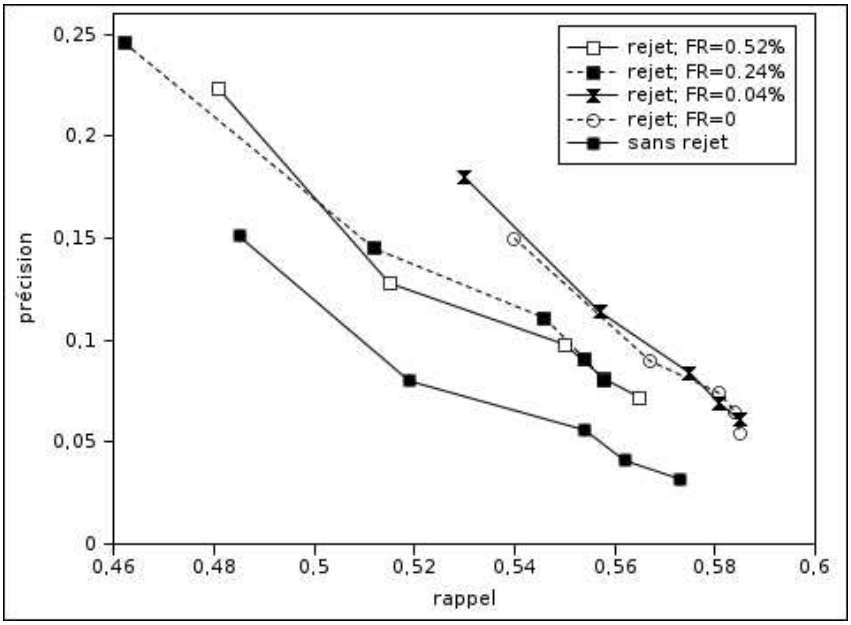


FIG. 5.12 – Rappel-précision du premier système avec et sans l'étape de filtrage des rejets évidents.

On peut constater l'effet bénéfique de l'étape de filtrage des rejets évidents sur la précision du système, pour tous les classifieurs SVM considérés. En ce qui concerne le rappel, celui-ci augmente lorsque les valeurs de FR sont suffisamment faibles : FR=0 et FR=0,04%. À partir de FR=0.24%, le rappel du système est affecté : certains chiffres appartenant aux champs numériques sont rejetés. Remarquons que pour les deux valeurs de FR les plus faibles, l'étape de filtrage permet d'obtenir des compromis en rappel-précision qui dominent totalement les compromis du système initial, pour tous les rangs considérés. Ceci montre l'intérêt d'une telle étape de filtrage.

Il est assez difficile de déterminer quel est le classifieur SVM optimal pour le filtrage des rejets évidents. Les classifieurs au faux rejet trop important sont abandonnés puisqu'ils font chuter le rappel du système. En revanche, les classifieurs «FR=0» et «FR=0,04%» offrent tous les deux des résultats intéressants. Selon nous, le classifieur «FR=0,04%» semble fournir plus de compromis non dominés. Au final, on obtient en TOP1 un rappel de 53% pour une précision de 18%. En TOP5, le rappel atteint 58,5% pour une précision de 6%.

Second système

Pour le second système, rappelons que le taux de faux rejet obtenu par l'étape de filtrage ne pouvait être atteint avant FA=100%. En intégrant les classifieurs dans le second système, tous font chuter le rappel, sans pour autant améliorer significativement la précision du système. Il semble donc qu'ancun rejet ne soit «évident» dans le cas d'une discrimination formes numériques/rejet lorsqu'on considère les chiffres liés. La cause de cette difficulté à discriminer les composantes numériques (Digits, Séparateurs, Double Digits) des rejets vient certainement de la présence des Double Digits. En effet, leur forme et leur structure très variable les rend difficile à distinguer des rejets.

Les capacités de rejet de cette seconde approche pourraient toutefois être améliorées en mettant en place une stratégie alternative du type «classe modulaire» [Takahashi 03] (voir section 5.2.2) où 3 classifieurs «un contre tous» (D,S,DD) pourraient être mis en place et combinés afin de rejeter efficacement les formes non numériques. On pourrait pour cela utiliser la méthode d'optimisation proposée dans ce chapitre afin de mener conjointement l'apprentissage de ces trois classifieurs.

Nous présentons maintenant les résultats finaux des deux approches développées.

5.5.2 Comparaison finale des deux systèmes

Les expériences ont montré que l'étape de filtrage des rejets évidents améliore largement les résultats du premier système, alors qu'elle n'est pas pertinente pour le second. Par ailleurs, nous avons montré dans les chapitre 3 et 4 que la seconde approche donnait de meilleurs résultats pour le problème d'extraction des champs. On peut ainsi se demander si l'amélioration des capacités de rejet de la première approche a permis de rattrapper ses lacunes. La figure 5.13 compare les résultats finaux en rappel-précision des deux systèmes.

On peut constater que même sans l'étape de filtrage des rejets évidents, la seconde approche donne toujours de meilleurs résultats. En effet, tous les points obtenus avec la première approche et un filtrage des rejets évidents sont dominés par les compromis obtenus par la seconde approche. Pour toutes les propositions considérées, on observe une précision légèrement meilleure, et un rappel largement supérieur pour la seconde approche. On constate que l'écart entre les résultats a tendance à s'accentuer lorsqu'on considère plusieurs propositions. Nous expliquons cette supériorité pour les rangs supérieurs par la plus faible combinatoire du treillis dans la seconde approche. Dans le premier système, la combinatoire est très élevée à cause des différents ni-

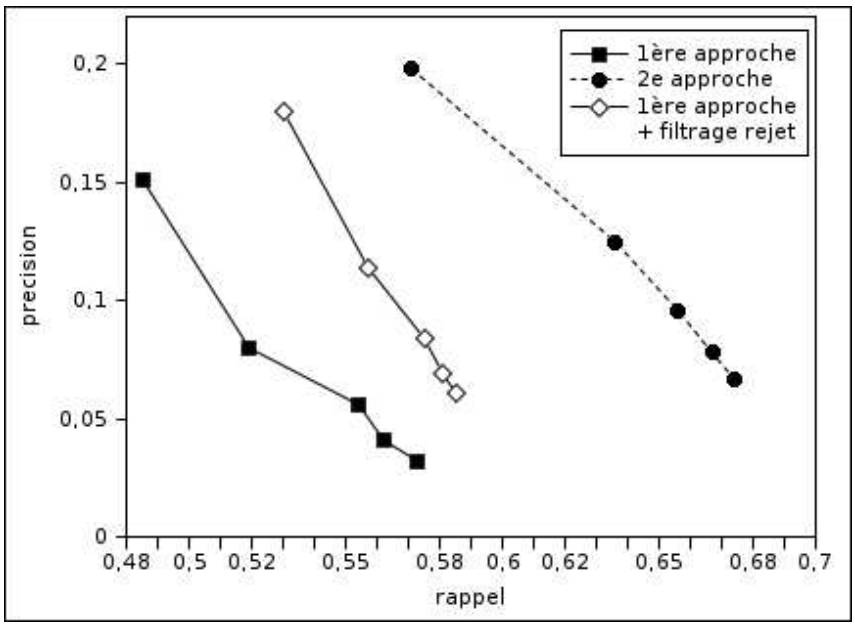


FIG. 5.13 – Comparaison finale des deux systèmes.

veaux de segmentation et des 10 hypothèses de reconnaissance chiffre ; il est donc plus difficile de faire remonter les solutions dans les premières propositions.

En ce qui concerne la répartition des fausses alarmes, la figure 5.14 donne la nature des séquences produites par les deux approches. On constate sur cette figure que malgré le filtrage des rejets évidents, la première approche propose toujours plus de champs mal alignés que la seconde.

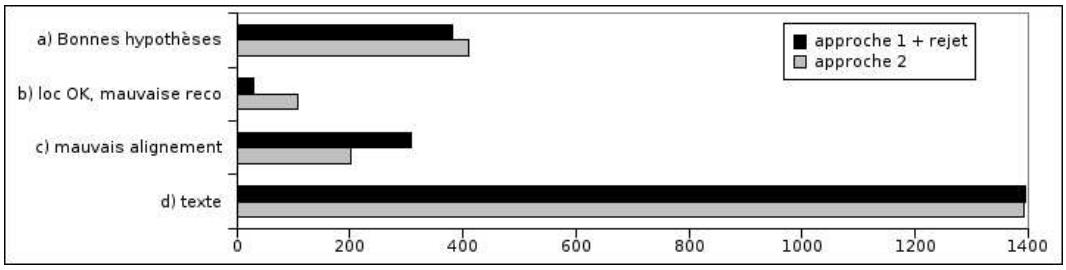


FIG. 5.14 – Comparaison de la nature des séquences numériques obtenues avec les deux approches (TOP1 uniquement).

Finalement, la seconde approche présente les meilleurs résultats en rappel comme en précision, et la figure 5.14 montre également qu'elle génère un nombre non négligeable de champs bien localisés mais mal reconnus, ce qui montre le poten-

tiel supérieur de la seconde approche pour l'extraction des champs.

5.6 Conclusion

Dans ce chapitre, nous avons abordé le problème central du rejet des composantes non numériques dans les deux chaînes de traitement proposées. Pour cela, nous avons dressé un panorama des méthodes existantes et montré que les approches mettant en œuvre une combinaison séquentielle de classifieurs étaient intéressantes puisqu'elles permettaient de traiter correctement les rejets d'ambiguïté comme les rejets de distance.

Nous avons ainsi appliqué une telle approche à nos deux chaînes de traitement afin d'améliorer leur capacité de rejet. L'idée est d'effectuer une première étape de rejet de distance par un filtrage des composantes qualifiées de «rejets évidents», avant de procéder au rejet d'ambiguïté par les méthodes de rejet déjà en place. Nous nous sommes donc concentrés sur le filtrage des rejets évidents, pour lequel nous avons mis en place un vecteur de caractéristiques dédié et un classifieur SVM. L'apprentissage du SVM pose toutefois problème puisqu'au-delà du réglage des hyperparamètres des SVM, nous sommes confrontés à des coûts de mauvaise classification à la fois fortement déséquilibrés et inconnus.

Pour faire face à ce problème, nous avons proposé une méthode d'optimisation multiobjectif basée sur un algorithme génétique, visant à réduire les taux de fausse alarme et de fausse acceptation du classifieur en jouant à la fois sur les hyperparamètres du classifieur SVM et sur ses coûts de mauvaise classification. Le résultat de cette optimisation produit un ensemble de classifieurs SVM proposant différents compromis FA/FR, parmi lequel le meilleur classifieur sera choisi en vue d'optimiser le système final.

Les résultats montrent que cette méthode permet d'améliorer considérablement les capacités de rejet de la première approche, mais qu'il n'apportait aucun bénéfice pour la seconde approche en raison de la difficulté à discriminer les chiffres liés des rejets. L'intégration de cette méthode au premier système complet a permis d'obtenir des compromis rappel-précision nettement plus intéressants, en particulier sur la précision. Malgré cette amélioration, la seconde approche reste toutefois la plus performante, aussi bien pour le rappel que pour la précision du système. Nous focaliserons donc nos prochains travaux sur cette seconde approche, en cherchant en particulier à améliorer l'étape de classification du 1er niveau (4 classes). Une méthode de reconnaissance alternative et plus performante pourra permettre d'augmenter encore le rappel et/ou la précision du système.

Conclusion générale

Dans le cadre du traitement automatique de courriers entrants, nous avons présenté dans cette thèse la mise en place d'un système d'extraction de séquences numériques dans des documents manuscrits quelconques. En effet, si la reconnaissance d'entités manuscrites isolées peut être considérée comme un problème résolu, l'extraction d'informations dans des images de documents aussi complexes et peu contraints que les courriers manuscrits libres est à ce jour un réel défi dans le domaine de la reconnaissance de l'écriture. Ce problème nécessite aussi bien la mise en œuvre de méthodes classiques de reconnaissance d'entités manuscrites que des méthodes issues du domaine de l'extraction d'information dans des documents électroniques. L'étude de ces deux domaines de recherche a permis de concevoir et d'implémenter deux chaînes de traitement fondées sur des stratégies différentes : la première applique intuitivement les différentes étapes de reconnaissance permettant une extraction des champs numériques de manière naturelle ; la seconde est basée sur des choix plus originaux qui se révèlent également plus pertinents.

Au cours de cette thèse, nous avons passé en revue les différentes techniques de reconnaissance de l'écriture manuscrite. Nous avons pu constater que la reconnaissance d'écriture a connu ces dernières années des progrès très importants, permettant désormais de faire face à la variabilité de l'écriture, et ainsi de reconnaître de manière fiable des entités manuscrites isolées : caractères (chiffre, lettre minuscule ou majuscule), séquences numériques ou mots. Au-delà de leur reconnaissance, c'est désormais la localisation automatique des entités manuscrites dans les documents qui pose problème à la communauté scientifique.

Nous avons ainsi étudié les différents systèmes de lecture de documents manuscrits et en particulier l'étape de localisation des entités qui nous intéresse plus particulièrement. Actuellement, les seules applications industrielles mettant en œuvre une reconnaissance d'écriture concernent des documents contraints créés spécialement dans l'optique d'une lecture automatique. C'est le cas des formulaires, des chèques bancaires ou des adresses postales, pour lesquels la localisation des entités est guidée par des connaissances *a priori* fortes sur les documents. Le problème est largement différent lorsque l'on aborde des documents plus faiblement contraints puisque l'on ne peut pas bénéficier d'un modèle de document suffisamment contraint pour faciliter la localisation des entités. Cette difficulté explique l'absence d'applications indus-

trielles relatives à des documents libres. Afin d'aborder le problème différemment, nous avons décrit les techniques d'extraction d'information dans les documents textuels.

Notre première contribution concerne ainsi la mise au point de stratégies générales pour l'extraction d'information dans des images de documents. Ces stratégies sont basées sur l'utilisation de techniques utilisées en reconnaissance de l'écriture (segmentation, classification de caractères, rejet des informations non pertinentes), mais aussi des méthodes issues de l'extraction d'information dans des documents électroniques (modélisation de séquences intégrant une description fine de l'information pertinente et une description grossière de l'information non pertinente). Deux stratégies alternatives ont ainsi été dégagées.

Bien qu'assez évidente, la mise en œuvre de la première stratégie qui consiste à localiser et à reconnaître les chiffres dans un document pour localiser les champs nécessite toutefois un certain nombre de modules de traitement. Notre seconde contribution concerne donc la conception des différentes briques de la chaîne dont chacune a été optimisée pour réduire les erreurs cumulées en fin de chaîne de traitement :

- Analyse de la structure du document par une segmentation du document en lignes de texte.
- Stratégie de segmentation/reconnaissance/rejet capable de localiser et reconnaître les chiffres dans un document quelconque. Ce module a nécessité le développement d'un classifieur chiffre aux performances satisfaisantes, d'un segmenteur efficace adapté aux formes numériques et d'une méthode de rejet des formes non numériques.
- Analyse syntaxique faisant intervenir les contraintes connues relatives à la syntaxe des champs (nombre de chiffres, position des séparateurs), et réalisant la localisation finale des séquences recherchées.

L'enchaînement de ces étapes constitue un système générique qui peut être adapté à d'autres types de champs (numéros de sécurité sociale, identifiant particulier, etc.) et d'autres types de documents. De plus, les temps de traitements tout à fait raisonnables autorisent son industrialisation.

Nous avons ensuite décrit la mise en œuvre de la seconde stratégie. Plus originale, elle repose sur une localisation et une reconnaissance des champs dissociée. La méthode de localisation constitue une contribution intéressante puisqu'elle est directement inspirée des méthodes d'extraction d'information dans des documents électroniques. Elle permet de localiser les champs numériques sans toutefois procéder à une reconnaissance chiffre ni à une segmentation des composantes qui sont autant de sources d'erreurs. Seule une étape de classification en classes syntaxiques est utilisée avant de procéder à l'extraction des champs. Nous avons également développé la méthode de reconnaissance des champs isolés adaptée à cette approche. Afin d'améliorer la pertinence des résultats fournis par ce système, nous avons mis au point une méthode de vérification des hypothèses de champs localisés et reconnus. Les résultats ont montré que cette seconde stratégie orientée extraction d'informa-

tions se révèle plus performante et moins coûteuse en temps de calcul que la première méthode orientée reconnaissance.

Nous avons enfin présenté la mise en place d'une stratégie de rejet des composantes numériques plus efficace pour les deux approches. Une méthode en deux étapes a été developpée, visant à traiter le rejet de distance et le rejet d'ambiguïté. Dans ce contexte, une contribution originale et générique a été apportée pour l'apprentissage de systèmes de classification dans les cas où les coûts de mauvaise classification sont déséquilibrés et inconnus. Basée sur un algorithme multiobjectif évolutionnaire, la méthode a été appliquée avec succès à un classifieur SVM pour le filtrage des rejets de distance. L'intégration de ce classifieur dans les systèmes complet a permis d'améliorer les résultats de la première approche, mais pas ceux de la seconde. Malgré cela, la seconde stratégie «dirigée par la syntaxe» procure toujours les meilleurs résultats, et sera donc conservée dans nos travaux futurs.

Malgré la réalisation d'un système complet efficace, générique et rapide, plusieurs améliorations et perspectives sont envisageables.

Rappelons que la seule connaissance *a priori* exploitée dans les approches présentées concerne la syntaxe des champs recherchés. L'intégration de ces systèmes en milieu industriel pourra bénéficier d'un certain nombre de connaissances *a priori* supplémentaires spécifiques aux types de champs recherchés afin d'en améliorer les performances. Premièrement, il s'agit de connaissances concernant la valeur des champs : par exemple, un numéro de téléphone commence toujours par un '0', un numéro de téléphone portable par '06', les codes clients commencent toujours par un '1', etc. Deuxièmement, il est possible d'exploiter certaines conniassances *a priori* sur la position la plus probable des champs : les codes postaux se trouvent par exemple souvent dans la partie supérieure gauche du document. Enfin l'industrialisation du système pourra également bénéficier d'une base de données contenant les informations des clients. En s'inspirant des stratégies de reconnaissance de mots, l'exploitation de ce «lexique» pourra se faire soit par une approche *non dirigée par le lexique* (vérification des hypothèses de reconnaissance), soit par une stratégie *dirigée par le lexique*.

Concernant les aspects modélisation des lignes de texte, plusieurs améliorations sont possibles. Actuellement, une analyse syntaxique est lancée pour chaque type de champ, ce qui entraine certaines fausses alarmes dues à la détection d'un type de champ dans un autre : code postal dans un code client ou dans un numéro de téléphone par exemple. Une mise en parallèle des différentes analyses syntaxiques pourra être effectuée afin de mettre en concurrence les hypothèses de localisation/reconnaissance et de limiter ainsi la fausse alarme. Une mise en parallèle des deux approches présentées pourra également être réalisée afin de fiabiliser les résultats. Nous avons présenté une implémentation de chaque stratégie, mais des mises en œuvre alternatives pourront également être expérimentées. Nous avons par exemple mentionné les approches basée sur une segmentation implicite, reposant sur

un classifieur dynamique de type HMM ou réseau de neurones récurrent.

Par ailleurs, le système est actuellement développé pour extraire les champs strictement numériques ; on pourra le faire évoluer afin de traiter les champs alpha-numériques tels que les dates, les numéros de plaques minéralogiques, certains numéros de client, etc. Il faudra dans ce cas intégrer les informations textuelles dans les modèles, ce qui suscite un certain nombre de questions, notamment en ce qui concerne la mise au point d'une méthode de segmentation et de reconnaissance adaptée à la fois aux lettres et aux chiffres.

En prolongeant cette idée d'intégration d'information textuelle dans les modèles, une perspective intéressante est la combinaison du système présenté dans cette thèse avec les travaux effectués en parallèle par Koch [Koch 06], visant à extraire des mots clés dans des documents semblables aux nôtres. Il existe selon nous deux moyens d'effectuer cette combinaison. La première pourrait être une simple combinaison parallèle des approches, qui devrait permettre d'associer, et donc de fiabiliser mutuellement les résultats des deux chaînes de traitement par le biais de la mise en concurrence des hypothèses de champs numériques et de mots détectés dans les mêmes zones du document. La deuxième pourrait être mise en œuvre en fusionnant les modèles d'extractions d'information textuelles et numériques dans un même modèle, permettant de détecter et d'associer les différents champs extraits (mots clés et champs numériques). Si ces perspectives soulèvent de nombreux problèmes de mise en œuvre, leur réalisation permettrait d'obtenir un système d'extraction d'information de haut niveau dans des images de documents.

Travaux de l'auteur

[1] Chatelain (C.), Heutte (L.) et Paquet (T.), « Discrimination between digits and outliers in handwritten documents applied to the extraction of numerical fields », *International Workshop on Frontiers in Handwriting Recognition, La Baule, France*, 2006, p. 475–480.

[2] Chatelain (C.), Heutte (L.) et Paquet (T.), « Discrimination chiffre/rejet pour l'extraction de champs numériques dans des documents manuscrits », *Colloque International Francophone sur l'Ecrit et le Document, Fribourg, Suisse*, 2006, p. 55–60.

[3] Chatelain (C.), Heutte (L.) et Paquet (T.), « A two-stage outlier rejection strategy for numerical field extraction in handwritten documents », *International Conference on Pattern Recognition, Hong Kong, China, IEEE proceedings*, vol. 3, 2006, p. 224–227.

[4] Chatelain (C.), Koch (G.), Heutte (L.) et Paquet (T.), « Une méthode dirigée par la syntaxe pour la detection de champs numériques dans des documents manuscrits », *revue Traitement du Signal*, vol. 23(2), 2006, p. 179–198.

[5] Chatelain (C.), Heutte (L.) et Paquet (T.), « Segmentation-driven recognition applied to numerical field extraction from handwritten incoming mail documents », *Document Analysis System, Nelson, New Zealand, LNCS 3872, Springer*, 2006, p. 564–575.

[6] Chatelain (C.), Heutte (L.) et Paquet (T.), « Une méthode d'extraction automatique de champs numériques dans des documents manuscrits », *6e journée francophone d'Extraction et Gestion de Connaissance, Lille, France*, vol. 1, 2006, p. 23–34.

[7] Chatelain (C.), Heutte (L.) et Paquet (T.), « Reconnaissance de champs numériques dans des documents manuscrits libres », *Reconnaissance de Forme et Intelligence Artificielle, Tours, France*, 2006, p. 45–63 (Actes sur CDROM).

[8] Chatelain (C.), Heutte (L.) et Paquet (T.), « A syntax-directed method for numerical field extraction using classifier combination », *9th International Workshop on Frontiers in Handwriting Recognition, Tokyo, Japan*, 2004, p. 93–98.

[9] Chatelain (C.), Adam (S.), Lecourtier (Y.) *et al.*, « SVM model selection using a multi-objective algorithm », *Pattern Recognition Letters, article soumis.*

Bibliographie

[Amin 97] A. Amin, S. Iyer & W. Wilson. *Recognition of hand-printed latin characters based on a structural approach with a neural network classifier.* Journal of Electronic Imaging, vol. 6, pages 303–310, 1997.

[Amin 98] A. Amin. *Off-Line Arabic Character-Recognition : The State of the Art.* Pattern Recognition, vol. 31, no. 5, pages 517–530, 1998.

[Anastasio 98] M. Anastasio, M. Kupinski & R. Nishikawa. *Optimization and FROC analysis of rule-based detection schemes using a multiobjective approach.* IEEE Trans. Med. Imaging, vol. 17, pages 1089–1093, 1998.

[Angluin 87] D. Angluin. *Queries and concept learning.* Machine Learning, vol. 2, pages 319–342, 1987.

[Annick 94] L. Annick. *Lexicon reduction based on global features for on-line handwriting.* IWFHR, page 12, 1994.

[Appelt 99] D.E. Appelt & D.J. Israel. *Introduction to Information Extraction Technology.* A Tutorial Prepared for IJCAI-99, 1999.

[Arica 01] N. Arica & F.T. Yarman-Vural. *An overview of character recognition focused on off-line handwriting.* IEEE Trans. on SMC - part C, vol. 31, no. 2, pages 216–233, 2001.

[Arlandis 02] J. Arlandis, J.C. Perez-Cortes & J. Cano. *Rejection strategies and confidance measures for a k-NN classifier in an OCR task.* ICPR, vol. 1, pages 576–579, 2002.

[Ayat 02] N.E. Ayat, M. Cheriet & C.Y. Suen. *Empirical error based optimization of SVM kernels : Application to digit image recognition.* IWFHR, pages 292–297, 2002.

[Ayat 05] N.E. Ayat, M. Cheriet & C.Y. Suen. *Automatic model selection for the optimization of SVM kernels.* Pattern Recognition, vol. 38, no. 10, pages 1733–1745, 2005.

[Bayer 97] T. Bayer, U.Bohnacker & I.Renz. *Information Extraction from Paper Documents.* Handbook of Character Recognition and Document Image Analysis, 1997.

[Bellili 01] A. Bellili, M. Gilloux & P. Gallinari. *An Hybrid MLP-SVM Handwritten Digit Recognizer.* ICDAR, pages 28–32, 2001.

[Bellili 03] A. Bellili, M. Gilloux & P. Gallinari. *An MLP-SVM combination architecture for offline handwritten digit recognition : Reduction of recognition errors by Support Vector Machines rejection mechanisms.* IJDAR, vol. 5, no. 4, pages 244–252, 2003.

[Bengio 95] Y. Bengio, Y. Le Cun, C. Nohl & C. Burges. *LeRec : A NN-HMM hybrid for on-line handwriting recognition.* Neural Computation, vol. 7, no. 6, pages 1289–1303, 1995.

[Bikel 99] D. M. Bikel, R. L. Schwartz & R.M. Weischedel. *An Algorithm that Learns What's in a Name.* Machine Learning, vol. 34, no. 1-3, pages 211–231, 1999.

[Bippus 97] R.D. Bippus. *1-Dimensional and Pseudo 2-Dimensional HMMS for the Recognition of German Literal Amounts.* ICDAR, vol. 2, pages 487–490, 1997.

[Bishop 95] C.M. Bishop. Neural networks for pattern recognition. Oxford University Press, 1995.

[Bottou 94] L. Bottou, C. Cortes, J.S. Denker, H. Drucker, I. Guyon, L.D. Jackel, Y. LeCun, U.A. Muller, E. Sackinger, P. Simard & V. Vapnik. *Comparison of classifier methods : a case study in handwritten digit recognition.* ICPR, vol. 2, pages 77–82, 1994.

[Bozinovic 89] R.M. Bozinovic & S.N. Srihari. *Off-Line Cursive Script Word Recognition.* IEEE Trans. on PAMI, vol. 11, no. 1, pages 68–83, 1989.

[Bradley 97] A.P. Bradley. *The use of the area under the ROC curve in the evaluation of machine learning algorithms.* Pattern Recognition, vol. 30, pages 1145–1159, 1997.

[Breiman 84] L. Breiman, J. Friedman, R. Olshen & C. Stone. Classification and Regression Trees. Wadsworth and Brooks, Monterey, CA, 1984.

[Bridle 90] J. S. Bridle. *Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition.* Neurocomputing : Algorithms, Architectures and Applications, pages 227–236, 1990.

[Britto 00] A. Britto, R. Sabourin, E. Lethelier, F. Bortolozzi & C. Suen. *Improvement handwritten numeral string recog-*

nition by slant normalization and contextual information. IWFHR, pages 323–332, 2000.

[Britto 02] A.S. Britto, R. Sabourin, F. Bortolozzi & C.Y. Suen. *A String Length Predictor to Control the Level Building of HMMs for Handwritten Numeral Recognition.* ICPR, vol. 4, pages 31–34, 2002.

[Britto 03] A. Britto, R. Sabourin, F. Bortolozzi & C. Suen. *Recognition of handwritten numeral strings using a two-stage HMM-Based method.* IJDAR, vol. 5, pages 102–117, 2003.

[Bromley 93] J. Bromley & J.S. Denker. *Improving rejection performance on handwritten digits by training with rubbish.* Neural computation, vol. 5, pages 367–370, 1993.

[Bui 04] L.T. Bui, D. Essam, H.A. Abbass & D. Green. *Performance analyis of multiobjective evolutionary methods in noisy environnments.* APS 2004, pages 29–39, 2004.

[Burges 97] C.J. C. Burges & B. Schölkopf. *Improving the Accuracy and Speed of Support Vector Machines.* NIPS, vol. 13, pages 500–506, 1997.

[Cai 99] J. Cai & Z.Q. Liu. *Integration of structural and statistical Information for Unconstrained Handwritten Numeral Recognition.* IEEE Trans. on PAMI, vol. 21, no. 3, pages 263–270, 1999.

[Califf 03] M.E. Califf & R.J. Mooney. *Bottom-up relational learning of pattern matching rules for information extraction.* JMLR, vol. 4, pages 177–210, 2003.

[Cao 97] J. Cao, M. Ahmadi & M. Shridhar. *A Hierarchical Neural-Network Architecture for Handwritten Numeral Recognition.* Pattern Recognition, vol. 30, no. 2, pages 289–294, 1997.

[Casey 96] R. Casey & E. Lecolinet. *A survey of methods and strategies in character segmentation.* IEEE Trans. on PAMI, vol. 18, no. 7, pages 690–706, 1996.

[Cavalin 06] P.R. Cavalin, A.Britto, F.Bortolozzi, R. Sabourin & L.Oliveira. *An implicit segmentation-based method for recognition of handwritten strings of characters.* ACM symposium on Applied computing, pages 836–840, 2006.

[Chang 01] C.C. Chang & C.J. Lin. *LIBSVM : a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[Chapelle 02] O. Chapelle, V. Vapnik, O. Bousquet & S. Mukherjee. *Choosing multiple parameters for support vector machines.* Machine Learning, vol. 46, no. 1, pages 131–159, 2002.

[Chen 00] Y. Chen & J.F. Wang. *Segmentation of Single- or Multiple-Touching Handwritten Numeral String Using Background and Foreground Analysis.* IEEE Trans. on PAMI, vol. 22, no. 11, pages 1304–1317, 2000.

[Chi 96] Z.R. Chi, M. Suters & H. Yan. *Handwritten Digit Recognition Using Combined ID3-Derived Fuzzy Rules and Markov-Chains.* Pattern Recognition, vol. 29, no. 11, pages 1821–1833, 1996.

[Cho 95] W. Cho, S.W. Lee & J.H. Kim. *Modelling and recognition of cursive words with hidden markov models.* Pattern Recognition, vol. 28, no. 12, pages 1941–1953, 1995.

[Chow 70] C.K. Chow. *On Optimum Recognition Error and Reject Tradeoff.* IEEE Trans. in Information Theory, vol. 16, no. 1, pages 41–46, 1970.

[Chung 03] K.M. Chung, W.C. Kao, C.L. Sun & C.J. Lin. *Radius margin bounds for support vector machines with RBF kernel.* Neural comput., vol. 15, no. 11, pages 2643–2681, 2003.

[Cohen 91] E. Cohen, J.J. Hull & S.N. Srihari. *Understanding handwritten text in a structured environment : determining ZIP codes from addresses.* Character and handwriting recognition - Expanding frontiers, vol. 30, pages 221–264, 1991.

[Cohen 94] E. Cohen, J.J. Hull & S.N. Srihari. *Control structure for interpreting handwritten addresses.* IEEE Trans. on PAMI, vol. 16, no. 10, pages 1049–1055, 1994.

[Cohn 05] T. Cohn & P. Blunsom. *Semantic Role Labelling with Tree Conditional Random Fields.* Ninth Conference on Computational Natural Language, pages 169–172, 2005.

[Collobert 02] R. Collobert, S. Bengio & J. Mariéthoz. *Torch : a modular machine learning software library.* Technical Report IDIAP-RR 02-46, 2002.

[Congedo 95] G. Congedo, G. Dimauro, S. Impedovo & G. Pirlo. *Segmentation of numeric strings.* ICDAR, vol. 2, pages 1038–1041, 1995.

[Connell 02] S.D. Connell & A.K. Jain. *Writer Adaptation for Online Handwriting Recognition.* IEEE Trans. on PAMI, vol. 24, no. 3, pages 329–346, 2002.

[Corne 00] D.W. Corne, J.D. Knowles & M.J. Oates. *The pareto envelope-based selection algorithm for multiobjective optimization.* Parallel problem solving from nature, pages 839–848, 2000.

[Cornuéjols 02] A. Cornuéjols & L. Miclet. Apprentissage artificiel, concepts et algorithmes. Eyrolles, 2002.

[Cortes 95] C. Cortes & V. Vapnik. *Support vector networks.* Machine Learning, vol. 20, pages 273–297, 1995.

[Cowie 96] J. Cowie & W. Lehnert. *Information extraction.* Communications of the ACM, vol. 39, no. 1, pages 80–91, 1996.

[CoxIII 82] C.H. CoxIII, P. Coueignoux, B.B.Blesser & M. Eden. *Skeletons : A link between theoretical and physical letter descriptions.* Pattern Recognition, vol. 15, no. 1, pages 11–22, 1982.

[Cracknell 98] C. Cracknell, A.C. Downton & L. Du. *An object-oriented descriptive language to facilitate advanced handwritten form processing.* IVC, vol. 16, no. 12-13, pages 843–853, 1998.

[Crettez 98] J.P. Crettez & G. Lorette. *Reconnaissance de l'´criture manuscrite.* Techniques de l'Ingénieur, traité Informatique, vol. H1 358, 1998.

[de Waard 94] W. P. de Waard. *Neural techniques and postal code detection.* Pattern Recognition Letters, vol. 15, no. 2, pages 199–205, 1994.

[Deb 00] K. Deb, S. Agrawal, A. Pratap & T. Meyarivan. *A fast elitist nondominated sorting genetic algorithm for multiobjective optimization : NSGA-II.* In Parallel problem solving from nature, pages 849–858. 2000.

[Devijver 82] P.A. Devijver & J. Kittler. Pattern recognition, a statistical approach. Englewood Cliffs, London, 1982.

[Dey 99] S. Dey. Adding feedback to improve segmentation and recognition of handwritten numerals. Master's thesis, Massachusetts Institute of Technology, 1999.

[Djeziri 97] S. Djeziri, Fathallah Nouboud & Rejean Plamondon. *Extraction of Items From Checks.* ICDAR, vol. 2, pages 749–752, 1997.

[Duin 02] R. Duin. *The Combining Classifier : To Train or Not to Train ?* ICPR, vol. 2, pages 765–770, 2002.

[Dzuba 97a] G. Dzuba, A. Filatov, D. Gershuny, I. Kil & V. Nikitin. Check amount recognition based on the cross validation of courtesy and legal amount fields, pages 177–194. Automatic Bank Check Processing. World Scientific, 1997.

[Dzuba 97b] G. Dzuba, A. Filatov & A. Volgunin. *Handwritten ZIP Code Recognition.* ICDAR, pages 766–770, 1997.

[E.Ashraf 03] E.Ashraf & A. Reda. *Segmentation of connected handwritten numeral strings.* Pattern Recognition, vol. 36, no. 3, pages 625–634, 2003.

[E.Charniak 93] E.Charniak. *Statistical Language Learning.* MIT - Press, Cambridge, MA, 1993.

[El Hajj 05] R. El Hajj, L. Likforman Sulem & C. Mokbel. *Arabic handwriting recognition using baseline dependant features and hidden Markov modeling.* ICDAR, vol. 2, pages 893–897, 2005.

[El-Yacoubi 99] A. El-Yacoubi, M. Gilloux, R. Sabourin & C. Y. Suen. *An HMM Based Approach for Off-line Unconstrained Handwritten Word Modeling and Recognition.* IEEE Trans. on PAMI, vol. 21, no. 8, pages 752–760, 1999.

[El-Yacoubi 02] A. El-Yacoubi, M. Gilloux & J.-M. Bertille. *A Statistical Approach for Phrase Location and Recognition within a Text Line : An Application to Street Name Recognition.* IEEE Trans. on PAMI, vol. 24, no. 2, pages 172–188, 2002.

[Everson 06] R.M. Everson & J.E. Fieldsend. *Multi-class ROC analysis from a multi-objective optimisation perspective.* Pattern Recognition Letters, page in press, 2006.

[Favata 96] J.T. Favata & G. Srikantan. *A multiple feature/resolution approach to handprinted digit and character recognition.* International Journal of Imaging Systems and Technology, vol. 7, no. 4, pages 304–311, 1996.

[Ferri 02] C. Ferri, P. Flach & J. Hernandez-Orallo. *Learning Decision Trees Using the Area Under the ROC Curve.* Proceedings of the 19th International Conference on Machine Learning, pages 139–146, 2002.

[Fieldsend 04] J.E. Fieldsend & R.M. Everson. *ROC optimisation of safety related systems.* Proceedings of ROCAI 2004, pages 37–44, 2004.

[Filatov 95] A. Filatov, A. Gitis & I. Kil. *Graph-based handwritten digit string recognition.* ICDAR, vol. 02, pages 845–848, 1995.

[Foggia 99] P. Foggia, C. Sansone, F. Tortorella & M. Vento. *Combining statistical and structural approaches for handwritten character description.* IVC, vol. 17, no. 9, pages 701–711, 1999.

[Fonseca 93] C.M. Fonseca & P.J. Flemming. *Genetic algorithm for multiobjective optimization : formulation, discussion and generalization.* Proceedings of ICGA 1993, pages 416–423, 1993.

[Forney 73] G.D. Forney. *The Viterbi Algorithm.* Proc. IEEE, vol. 61, pages 268–278, 1973.

[Francesconi 01] E. Francesconi, M.Gori, S. Marinai & G.Soda. *A serial combination of connectionist-based classifiers for OCR.* IJDAR, vol. 3, pages 160–168, 2001.

[Freitag 99] D. Freitag & A.K. McCallum. *Information extraction with HMMs and shrinkage.* AAAI-99 Workshop on Machine Learning for Information Extraction, pages 31–36, 1999.

[Friedrichs 05] F. Friedrichs & C. Igel. *Evolutionnary tuning of multiple SVM parameters.* Neurocomputing, vol. 64, pages 107–117, 2005.

[Fujisawa 92] H. Fujisawa, Y. Nakano & K. Kurino. *Segmentation Methods for Character Recognition : From Segmentation to Document Structure Analysis.* Proc. of the IEEE, vol. 80, no. 7, pages 1079–1092, 1992.

[Fukunaga 89a] K. Fukunaga & R.R. Hayes. *The Reduced Parzen Classifier.* IEEE Trans. on PAMI, vol. 11, no. 4, pages 423–425, 1989.

[Fukunaga 89b] K. Fukunaga & D. M. Hummels. *Leave-One-Out Procedures for Nonparametric Error Estimates.* IEEE Trans. PAMI, vol. 11, no. 4, pages 421–423, 1989.

[Gader 96] P.D. Gader & M.A. Khabou. *Automatic Feature Generation for Handwritten Digit Recognition.* IEEE Trans. on PAMI, vol. 18, no. 12, pages 1256–1261, 1996.

[Gader 97] P.D. Gader, M.A. Mohamed & J.H. Chiang. *Handwritten Word Recognition with Character and Inter-Character Neural Networks.* SMC - part B, vol. 27, no. 1, pages 158–164, 1997.

[Giacinto 00] G. Giacinto, F. Roli & G. Fumera. *Selection of Classifiers Based on Multiple Classifier Behaviour.* SSPR/SPR, pages 87–93, 2000.

[Gilloux 95] M. Gilloux, B. Lemari & M. Leroux. *A hybrid radial basis function network/hidden markov model handwritten word recognition system.* ICDAR, pages 394–397, 1995.

[Giusti 02] N. Giusti, F. Masulli & A. Sperduti. *Theoretical and experimental analysis of a two-stage system for classification.* IEEE Trans. on PAMI, vol. 24, pages 893–904, 2002.

[Gold 67] E.M. Gold. *Language identification in the limit.* Information and Control, vol. 10, pages 447–474, 1967.

[Gold 03] C. Gold & P. Sollich. *Model selection for support vector machine classification.* Neurocomputing, vol. 55, pages 221–249, 2003.

[Goldberg 89] David E. Goldberg. Genetic algorithms in search, optimization and machine learning. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[Gori 98] M. Gori & F. Scarselli. *Are multilayer perceptrons adequate for pattern recognition and verification ?* IEEE Trans. on PAMI, vol. 20, pages 1121–1132, 1998.

[Gorski 97] N. Gorski. *Optimizing error-reject trade off in recognition systems.* ICDAR, vol. 2, pages 1092–1096, 1997.

[Govindan 90] V. Govindan & A. Shivaprasad. *Character recognition- a review.* Pattern Recognition, vol. 23, no. 7, pages 671–683, 1990.

[Guermeur 00] Y. Guermeur, A. Eliseeff & H. PaugamMoisy. *A new multi-class svm based on a uniform convergence result.* IJCNN, vol. IV, pages 183–188, 2000.

[Guigue 05] V. Guigue. *Méthodes à noyaux pour la représentation et la discrimination de signaux non-stationnaires.* PhD thesis, INSA de Rouen, 2005.

[Guyon 02] I. Guyon, J. Weston, S. Barnhill & V. Vapnik. *Gene Selection for Cancer Classification using Support Vector Machines.* Machine Learning, vol. 46, no. 3, pages 389–422, 2002.

[Guyon 03] I. Guyon & A. Elisseeff. *An introduction to variable and feature selection.* JMLR, vol. 3, pages 1157–1182, 2003.

[Ha 98] T.M. Ha, M. Zimmermann & H. Bunke. *Off-line Handwritten Numeral String Recognition by Combining Segmentation-based and Segmentation-free Methods.* Pattern Recognition, vol. 31, no. 3, pages 257–272, 1998.

[Hao 97] H.W. Hao, X.H. Xiao & R.W. Dai. *Handwritten Chinese character recognition by metasynthetic approach.* Pattern Recognition, vol. 30, no. 7, pages 1321–1328, 1997.

[Heutte 94] L. Heutte. *Reconnaissance de caractères manuscrits : Application à la lecture des chèques et des enveloppes postales.* Thèse de doctorat, Université de Rouen, 1994.

[Heutte 97] L. Heutte, P. Barbosa-Perreira, O. Bougeois, J.V. Moreau, B. Plessis, P. Courtellemont & Y. Lecourtier. *Multi-bank check recognition system : consideration on the numeral amount recognition module.* IJPRAI, vol. 11, pages 595–618, 1997.

[Heutte 98] L. Heutte, T. Paquet, J.V. Moreau, Y. Lecourtier & C. Olivier. *A structural/statistical feature based vector for handwritten character recognition.* Pattern Recognition Letters, vol. 19, pages 629–641, 1998.

[Ho 94] T.K. Ho, J.J. Hull & S.N. Srihari. *Decision Combination in Multiple Classifier Systems*. IEEE Trans. on PAMI, vol. 16, no. 1, pages 66–75, 1994.

[Horn 94] J. Horn, N. Nafpliotis & Goldberg D.E. *A niched pareto genetic algorithm for multiobjective optimization*. Proceedings of IEEE-WCCC, pages 82–87, 1994.

[Hsu 02a] C. Hsu & C. Lin. *A comparison of methods for multi-class support vector machines*. IEEE Trans. on Neural Networks, vol. 13, pages 415–425, 2002.

[Hsu 02b] C.W. Hsu & C.J. Lin. *A simple decomposition method for support vector machine*. Machine Learning, vol. 46, pages 219–314, 2002.

[Hu 62] M.K. Hu. *Visual pattern recognition by moment invariants*. IRE Trans. on Information Theory, vol. 8, pages 179–187, 1962.

[Hu 98] J.M. Hu & H. Yan. *Structural Primitive Extraction and Coding for Handwritten Numeral Recognition*. Pattern Recognition, vol. 31, no. 5, pages 493–509, 1998.

[Huang 95] Y. S. Huang & C. Y. Suen. *A Method of Combining Multiple Experts for the Recognition of Unconstrained Handwritten Numerals*. IEEE Trans. on PAMI, vol. 17, no. 1, pages 90–94, 1995.

[Huang 06] C-L. Huang & C-J. Wang. *A GA-based feature selection and parameters optimization for support vector machine*. Expert systems with application, vol. 31, pages 231–240, 2006.

[Hwang 97] Y.S. Hwang & S.Y. Bang. *Recognition of unconstrained handwritten numerals by a radial basis function neural network classifier*. Pattern Recognition Letters, vol. 18, no. 7, pages 657–664, 1997.

[Impedovo 97a] S. Impedovo, G. Pirlo & A. Salzo. A new engineered system, pages 5–42. Automatic Bank Check Processing. World Scientific, 1997.

[Impedovo 97b] S. Impedovo, P.S.P. Wang & H. Bunke, editeurs. Automatic bankcheck processing, volume 28 of *Series in Machine Perception and Artificial Intelligence*. World Scientific, 1997.

[Ishitani 01] Y. Ishitani. *Model-Based Information Extraction Method Tolerant of OCR Errors for Document Images*. ICDAR, pages 908–915, 2001.

[Jacobs 91] R. A. Jacobs, M. I. Jordan, S. J. Nowlan & G. E. Hinton. *Adaptive mixture of local experts*. Neural Computation, vol. 3, pages 79–87, 1991.

[Jain 92] Anil K. Jain & Sushil K. Bhattacharjee. *Address block location on envelopes using Gabor filters.* Pattern Recognition, vol. 25, no. 12, pages 1459–1477, 1992.

[Jain 00] A.K. Jain, R.P.W. Duin & J. Mao. *Statistical pattern recognition : A review.* IEEE Trans. on PAMI, vol. 22, no. 1, pages 4–37, 2000.

[Jarousse 98] C. Jarousse & C. Viard-Gaudin. *Localisation du code postal par réseau de neurones sur bloc adresse manuscrit non contraint.* CIFED, pages 72–81, 1998.

[Joachims 99] T. Joachims. *Making large-Scale SVM Learning Practical.* Advances in Kernel Methods - Support Vector Learning, MIT Press, 1999.

[Jung 00] K. Jung & H.J. Kim. *On-line recognition of cursive Korean characters using graph representation.* Pattern Recognition, vol. 33, no. 3, pages 399–412, 2000.

[J.Weston 00] J.Weston, S. Mukherjee, O. Chapelle, M. Pontil, T.Poggio & V. Vapnik. *Feature Selection for SVMs.* NIPS, pages 668–674, 2000.

[Kapp 04] M.N. Kapp, C. Freitas & R. Sabourin. *Handwritten brazilian month recognition : An analysis of two NN architectures and a rejection mechanism.* IWFHR, pages 209–214, 2004.

[Keeni 96] K. Keeni, H. Shimodaira, T. Nishino & Y. Tan. *Recognition of Devanagari Characters Using Neural Networks.* IEICE, vol. E79-D, no. 5, pages 523–528, 1996.

[Keerthi 02] S.S. Keerthi. *Efficient tuning of SVM parameters using radius/margin bound and iterative algorithms.* IEEE transaction on Neural Networks, vol. 13, no. 5, pages 1225–1229, 2002.

[Khare 02] V. Khare, X. Yao & K. Deb. *Performance scaling of multiobjective evolutionary algorithm.* Technical report - SCS, University of Birmingham, pages 1–70, 2002.

[Kim 97a] G. Kim & V. Govindaraju. Banckcheck recognition using cross validation between legal and courtesy amount, pages 195–212. Automatic Bank Check Processing,. World Scientific, 1997.

[Kim 97b] G. Kim & V. Govindaraju. *A Lexicon Driven Approach to Handwritten Word Recognition for Real-Time Applications.* IEEE Trans. on PAMI, vol. 19, no. 4, pages 366–378, 1997.

[Kim 98] G. Kim & V. Govindaraju. *Handwritten Phrase Recognition as Applied to Street Name Images.* Pattern Recognition, vol. 31, no. 1, pages 41–51, 1998.

[Kim 99] G. Kim, V. Govindaraju & S.N. Srihari. *An architecture for handwritten text recognition system.* IJDAR, vol. 2, pages 37–44, 1999.

[Kim 00] J.H. Kim, K.K. Kim, C.P. Nadal & C.Y. Suen. *A Methodology of Combining HMM and MLP Classifiers for Cursive Word Recognition.* IWFHR, vol. 2, page 2319, 2000.

[Kim 01] S. H. Kim, C. Y. Suen, S. Jeong & G.S. Lee. *Word Segmentation in Handwritten Korean Text Lines Based on Gap Clustering Techniques.* ICDAR, page 189, 2001.

[Kim 02a] Ho-Yon Kim, Kil-Taek Lim & Yun-Seok Nam. *Handwritten Numeral String Recognition Using Neural Network Classifier Trained with Negative Data.* IWFHR, page 395, 2002.

[Kim 02b] K.K. Kim, J.H. Kim & C. Y. Suen. *Segmentation-based recognition of handwritten touching pairs of digits using structural features.* Pattern Recognition Letters, vol. 23, no. 1, pages 13–24, 2002.

[Kim 04] K.M. Kim, J.J. Park, Y.G. Song, I.C. Kim & C.Y. Suen. *Recognition of Handwritten Numerals Using a Combined Classifier with Hybrid Features.* SSPR/SPR, pages 992–1000, 2004.

[Kimura 91] F. Kimura & M. Shridhar. *Handwritten numerical recognition based on multiple algorithms.* Pattern Recognition, vol. 24, no. 10, pages 969–983, 1991.

[Kimura 94] F. Kimura, S. Tsuruoka, Y. Miyake & M. Shridhar. *A Lexicon Directed Algorithm for Recognition of Unconstrained Handwritten Words.* IEICE Trans. on Information & Syst., vol. E77-D, no. 7, pages 785–793, 1994.

[Kittler 98] J. Kittler, M. Hatef, R.P.W. Duin & J. Matas. *On Combining Classifiers.* IEEE Trans. on PAMI, vol. 20, no. 3, pages 226–239, 1998.

[Knerr 97] S. Knerr, V. Asimov, O. Baret, N. Gorsky & J.C. Simon D. Price. The a2ia intercheque system : Courtesy amount and legal amount recognition for french checks, pages 43–86. Automatic bankcheck processing. World Scientific, 1997.

[Knerr 98] S. Knerr & E. Augustin. *A neural network-hidden markov model hybrid for cursive word recognition.* ICPR, vol. 2, pages 1518–1520, 1998.

[Koch 04] G. Koch, L. Heutte & T. Paquet. *Combination of Contextual Information for Handwritten Word Recognition.* IWFHR, pages 468–473, 2004.

[Koch 06] G. Koch. *Catégorisation automatique de documents manuscrits : application aux courriers entrants*. Thèse de doctorat, Université de Rouen, 2006.

[Koerich 02] A.L. Koerich, Y. Leydier, R. Sabourin & C.Y. Suen. *A Hybrid Large Vocabulary Handwritten Word Recognition System Using Neural Networks with Hidden Markov Models*. pages 99–105, 2002.

[Koerich 03a] A. L. Koerich. *Unconstrained Handwritten Character Recognition Using Different Classification Strategies*. ANNPR, page 5 pages, 2003.

[Koerich 03b] A.L. Koerich, R. Sabourin & C.Y. Suen. *Large vocabulary off-line handwriting recognition : A survey*. Pattern Analysis and Applications, vol. 6, pages 97–121, 2003.

[Koga 01] M. Koga, R. Mine, H. Sako & H. Fujisawa. *A recognition method of machine-printed monetary amounts based on the two-dimensional segmentation and the bottom-up parsing*. ICDAR, pages 968–971, 2001.

[Kohavi 97] R. Kohavi & G.H. John. *Wrappers for feature subset selection*. Artif. Intell., vol. 97, no. 1-2, pages 273–324, 1997.

[Koller 96] D. Koller & M. Sahami. *Toward Optimal Feature Selection*. ICML, pages 284–292, 1996.

[Kristjannson 04] T. Kristjannson, A. Culotta, P. Viola & A. McCallum. *Interactive information extraction with constrained conditional random fields*. AAAI, pages 412–418, 2004.

[Kupinski 99] M. Kupinski & M. Anastasio. *Multiobjective genetic optimization of diagnostic classifiers with implications for generating receiver operating characterisitic curves*. IEEE Trans. Med. Imaging, vol. 8, pages 675–685, 1999.

[Lafferty 01] J. Lafferty, A. McCallum & F. Pereira. *Conditional Random Fields : Probabilistic Models for Segmenting and Labeling Sequence Data*. Proc. 18th International Conf. on Machine Learning, pages 282–289, 2001.

[Landgrebe 05] T. Landgrebe, P. Paclik & D.M.J. Tax. *Optimising two-stage recognition systems*. MCS' 05, pages 206–215, 2005.

[Lavalle 02] S.M. Lavalle & M.S. Branicky. *On the relationship between classical grid search and probabilistic roadmaps*. International Journal of Robotics research, vol. 23, pages 673–692, 2002.

[Lecce 00] V. Di Lecce, G. Dimauro, A. Guerriero & A. Salzo S. Impedovo G. Pirlo. *Classifier Combination : the role of a priori knowledge*. IWFHR VII, pages 143–152, 2000.

[Lecun 87] Y. Lecun. *Modèles connexionnistes de l'apprentissage*. PhD thesis, Université P. et M. Curie (Paris 6), juin 1987.

[LeCun 89] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard & L. D. Jackel. *Backpropagation Applied to Handwritten Zip Code Recognition*. Neural Computation, vol. 1, no. 4, pages 541–551, 1989.

[LeCun 95] Y. LeCun, L. D. Jackel, L. Bottou, A. Brunot, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, U. A. Muller, E. Sackinger, P. Simard & V. Vapnik. *Comparison of learning algorithms for handwritten digit recognition*. ICANN, pages 53–60, 1995.

[LeCun 98] Y. LeCun, L. Bottou, Y. Bengio & P. Haffner. *Gradient-Based Learning Applied to Document Recognition*. Proceedings of the IEEE, vol. 86, no. 11, pages 2278–2324, 1998.

[Lee 96] S.W. Lee. *Off-Line Recognition of Totally Unconstrained Handwritten Numerals Using Multilayer Cluster Neural Network*. IEEE Trans. on PAMI, vol. 18, no. 6, pages 648–652, 1996.

[Lee 97] Luan Ling Lee, Miguel Gustavo Lizárraga, Natanael Rodrigues Gomes & Alessandro L. Koerich. *A Prototype for Brazilian Bankcheck Recognition.* IJPRAI, vol. 11, no. 4, pages 549–569, 1997.

[Leek 97] T. R. Leek. *Information extraction using hidden Markov models*. Master's thesis, UC San Diego, 1997.

[Lei 04] Y. Lei, C. S. Liu, X.Q. Ding & Q. Fu. *A Recognition Based System for Segmentation of Touching Handwritten Numeral Strings*. IWFHR, pages 294–299, 2004.

[Lemarie 93] B. Lemarie. *Practical Realization of a Radial Basis Function Network for Handwritten Digit Recognition*. IWANN, pages 131–136, 1993.

[Leroux 97] M. Leroux, E. Lethelier, M. Gilloux & B. Lemarie. Automatic reading of handwritten amounts on french checks, pages 157–176. Automatic bank check processing. World Scientific, 1997.

[Lethelier 95] E. Lethelier, M. Leroux & M. Gilloux. *An automatic reading system for handwritten numeral amounts on French checks*. ICDAR, vol. 1, pages 92–97, 1995.

[Lethelier 96] E. Lethelier. *Combinaison des concepts de segmentation et de reconnaissance pour l'écriture manuscrite hors ligne : application au traitement des montants numériques de chèques*. Thèse de doctorat, 1996.

[L.Hermes 00] L.Hermes & J.M. Buhmann. *Feature Selection for Support Vector Machines.* ICPR, vol. 2, pages 2712–2715, 2000.

[Lii 93] J. Lii, P. Palumbo & S. Srihari. *Address block location using character recognition and address syntax.* ICDAR, pages 330–335, 1993.

[Likforman-Sulem 95a] L. Likforman-Sulem & C. Faure. *Une methode de resolution des conflits d'alignements pour la segmentation des documents manuscrits.* Traitement du Signal, vol. 12, pages 541–549, 1995.

[Likforman-Sulem 95b] L. Likforman-Sulem, A. Hanimyan & C. Faure. *A Hough based algorithm for extracting text lines in handwritten documents.* ICDAR, pages 774–777, 1995.

[Liu 00] C.L. Liu & M. Nakagawa. *Precise Candidate Selection for Large Character Set Recognition by Confidence Evaluation.* IEEE Trans. on PAMI, vol. 22, no. 6, pages 636–641, 2000.

[Liu 02a] C.L. Liu, K. Nakashima, H. Sako & H. Fujisawa. *Handwritten Digit Recognition Using State-of-the-Art Techniques.* IWFHR, pages 320–325, 2002.

[Liu 02b] C.L. Liu, H. Sako & H. Fujisawa. *Performance evaluation of pattern classifiers for handwritten character recognition.* IJDAR, vol. 4, pages 191–204, 2002.

[Liu 02c] J. Liu & P. Gader. *Neural Networks with Enhanced Outlier Rejection Ability for Off-Line Handwritten Word Recognition Pattern Recognition.* Pattern Recognition, vol. 35, pages 2061–2071, 2002.

[Liu 04] C.L. Liu, H. Sako & H. Fujisawa. *Effects of Classifier Structures and Training Regimes on Integrated Segmentation and Recognition of Handwritten Numeral Strings.* IEEE Trans. on PAMI, vol. 26, pages 1395–1407, 2004.

[Liu 06] C.L. Liu & H. Sako. *Class-specific feature polynomial classifier for pattern classification and its application to handwritten numeral recognition.* Pattern Recognition, vol. 39, no. 4, pages 669–681, 2006.

[Lopresti 00] D. Lopresti. *String techniques for detecting duplicates in document databases.* IJDAR, vol. 2, pages 186–199, 2000.

[Lorette 92] G. Lorette & Y. Lecourtier. *Reconnaissance et interprétation de textes manuscrits hors-ligne : Un problème d'analyse de scènes ?* CNED'92, pages 109–135, 1992.

[Lorette 99] G. Lorette. *Handwriting Recognition or reading ? What is the situation at the dawn of the 3rd Millenium ?* IJDAR, vol. 2, no. 1, pages 2–12, 1999.

[Lu 99] Z. Lu, Z. Chi, W.C. Siu & P. Shi. *A background-thinning-based approach for separating and recognizing connected handwritten digit strings.* Pattern Recognition, vol. 32, pages 921–933, 1999.

[Madasu 03] V.K. Madasu, M.H.M. Yusof, M. Hanmandlu & K. Kubik. *Automatic extraction of signatures from bank cheques and other documents.* Biennial Australian Pattern Recognition Conference, vol. 2, pages 591–600, 2003.

[Madhvanath 93] S. Madhvanath & V. Govindaraju. *Holistic lexicon reduction.* IWFHR, Buffalo, New York, pages 71–81, 1993.

[Madhvanath 95] S. Madhvanath, V. Govindaraju, V. Ramanaprasad, D. S. Lee & S. N. Srihari. *Reading handwritten US census forms.* ICDAR, vol. 1, pages 82–87, 1995.

[Manke 96] S. Manke, M. Finke & A. Waibel. *A fast search technique for large vocabulary on-line handwriting recognition.* IWFHR, Colchester, England, pages 437–444, 1996.

[Marti 99] U. Marti & H. Bunke. *A Full English Sentence Database for Off-Line Handwriting Recognition.* ICDAR, pages 705–708, 1999.

[Marti 01a] U. Marti & H. Bunke. *Text Line Segmentation and Word Recognition in a System for General Writer Independent Handwriting Recognition.* ICDAR, page 159, 2001.

[Marti 01b] U.V. Marti & H. Bunke. *Using a statistical Language model to improve the performance of an HMM-based cursive handwriting recognition system.* IJPRAI, vol. 15, pages 65–90, 2001.

[Martin 93] C.L. Martin. *Centered-Object Integrated Segmentation and Recognition of Overlapping Handprinted Characters.* Neural Computation, vol. 5, pages 419–429, 1993.

[Matan 92] O. Matan, C.J.C. Burges, Y.Le Cun & J.S. Denker. *Multi-Digit Recognition Using a Space Displacement Neural Network.* NIPS, vol. 4, pages 488–495, 1992.

[Milewski 06a] R Milewski. *Automatic Recognition Of Handwritten Medical For Search Engines.* Rapport technique, Department of Computer Science and Engineering, University at Buffalo, 2006.

[Milewski 06b] R. Milewski & V. Govindaraju. *Extraction of Handwritten Text from Carbon Copy Medical Form Images.* DAS06, pages 106–116, 2006.

[Milgram 04] J. Milgram, R. Sabourin & M. Cheriet. *An hybrid classification system which combines model-based and discriminative approaches.* ICPR, vol. 1, pages 155–162, 2004.

[Milgram 05] J. Milgram, M. Cheriet & R. Sabourin. *Estimating accurate multi-class probabilities with support vector machines.* IJCNN, vol. 3, pages 1906–1911, 2005.

[Miller 00] D. Miller, S. Boisen, R. Schwartz, R. Stone & R. Weischedel. *Named entity extraction from noisy input : speech and OCR.* Proceedings of the sixth conference on Applied natural language processing, pages 316–324, 2000.

[Mohammed 96] M. Mohammed & P. Gader. *Handwritten Word Recognition Using Segmentation-Free Hidden Markov Modeling and Segmentation-Based Dynamic Programming Techniques.* IEEE Trans. on PAMI, vol. 18, no. 5, pages 548–554, 1996.

[Morgan 93] N. Morgan, H. Bourlard, S. Renls, M. Cohen & H. Franco. *Hybrid neural network/hidden Markov model systems for continuous speech recognition.* IJPRAI, vol. 7, no. 4, 1993.

[Morita 02] M. Morita, R. Sabourin, F. Bortolozzi & C.Y. Suen. *Segmentation and Recognition of Handwritten Dates.* IWFHR, pages 105–110, 2002.

[Morita 06] M. Morita, R. Sabourin, F. Bortolozzi & C.Y. Suen. *Segmentation and Recognition of Handwritten Dates : An HMM-MLP hybrid approach.* IJDAR, to appear, 2006.

[Mouchère 06] H. Mouchère, E. Anquetil & N. Ragot. *Etude et gestion des types de rejet pour l'optimisation de classifieurs.* RFIA, Actes sur CDROM, 2006.

[Mozer 02] M. C. Mozer, R. Dodier, M. D. Colagrosso, C. Guerra-Salcedo & R. Wolniewicz. *Prodding the ROC Curve : Constrained Optimization of Classifier Performance.* NIPS, pages 1409–1415, 2002.

[MUC 91] Proceedings third message understanding conference (darpa). San Francisco, Morgan Kaufmann Publishers, 1991.

[MUC 92] Proceedings fourth message understanding conference (darpa). San Francisco, Morgan Kaufmann Publishers, 1992.

[MUC 93] Proceedings fifth message understanding conference (darpa). San Francisco, Morgan Kaufmann Publishers, 1993.

[MUC 95] Proceedings sixth message understanding conference (darpa). San Francisco, Morgan Kaufmann Publishers, 1995.

[MUC 98] Proceedings seventh message understanding conference (darpa). San Francisco, Morgan Kaufmann Publishers, 1998.

[Muggleton 92] S. Muggleton. *Inductive Logic Programming.* Academic Press,London, pages 3–27, 1992.

[Mulbregt 98] P. Mulbregt, I. van, L. Gillick, S. Lowe & J. Yamron. *Text Segmentation and Topic Tracking on Broadcast News Via a Hidden Markov Model Approach.* ICSLP'98, vol. 6, pages 2519–2522, 1998.

[Musicant 03] D. Musicant, V. Kumar & A. Ozgur. *Optimizing F-Measure with Support Vector Machines.* FLAIRS Conference, pages 356–360, 2003.

[Nagy 00] G. Nagy. *Twenty years of document image analysis in PAMI.* IEEE Trans. on PAMI, vol. 22, pages 38–62, 2000.

[Niyogi 97] D. Niyogi, S. Srihari & V. Govindaraju. *Analysis of Printed Forms.* Handbook of Character Recognition and Document Image Analysis, pages 485–502, 1997.

[Nosary 02] A. Nosary. *Reconnaissance automatique de textes Manuscrits par adaptation du scripteur.* Thèse de doctorat, Université de Rouen, 2002.

[Oh 99] I.S. Oh, J.S. Lee & C.Y. Suen. *Analysis of Class Separation and Combination of Class-Dependent Features for Handwriting Recognition.* IEEE Trans. on PAMI, vol. 21, no. 10, pages 1089–1094, 1999.

[Oh 02] I. Oh & C.Y. Suen. *A class-modular feedforward neural network for handwriting recognition.* Pattern Recognition, vol. 35, no. 1, pages 229–244, 2002.

[Oliveira 00] L.E. Oliveira, E. Lethelier & F. Bortolozzi. *A New Segmentation Approach for Handwritten Digits.* ICPR, vol. 2, pages 323–326, 2000.

[Oliveira 02a] L. S. Oliveira, R. Sabourin, F. Bortolozzi & C. Y. Suen. *Feature Selection Using Multi-Objective Genetic Algorithms for Handwritten Digit Recognition.* ICPR, pages 568–571, 2002.

[Oliveira 02b] L.S. Oliveira, R. Sabourin, F. Botolozzi & C.Y. Suen. *Automatic Recognition of Handwritten Numeral Strings : A Recognition and Verification Strategy.* IEEE Trans. on PAMI, vol. 24, pages 1438–1454, 2002.

[Oliveira 03] L.S. Oliveira, R. Sabourin, F. Botolozzi & C.Y. Suen. *Impacts of Verification on a numeral string recognition system.* Pattern Recognition Letters, vol. 24, 2003.

[Oliveira 04] L.S. Oliveira & R. Sabourin. *Support Vector Machines for Handwritten Numerical String Recognition.* pages 39–44, 2004.

[Oliveira 06] L.S. Oliveira, M. Morita & R. Sabourin. *Feature Selection for Ensembles Applied to Handwriting Recognition.* IJDAR, vol. 18, pages 262–279, 2006.

[Omachi 00] S. Omachi, F. Sun & H. Aso. *A Noise-Adaptive Discriminant Function and Its Application to Blurred Machine Printed Kanji Recognition.* IEEE Trans. on PAMI, vol. 22, no. 3, pages 314–319, 2000.

[Oommen 97] B. Oommen & R. Loke. *Pattern recognition of strings with substitutions.* Pattern Recognition, vol. 30, pages 789–800, 1997.

[Osuna 97] E. Osuna, R. Freund & F. Girosi. Support vector machines : Training and applications. 1997.

[Pal 01] U. Pal, A. Belaïd & C. Choisy. *Water Reservoir Based Approach for Touching Numeral Segmentation.* ICDAR, pages 892–897, 2001.

[Pal 03] U. Pal, A. Belaïd & C. Choisy. *Touching Numeral Segmentation Using Water Reservoir Concept.* Pattern Recognition Letters, vol. 24, pages 261–272, 2003.

[Palacios 97] R. Palacios & A. Gupta. A system for processing handwritten bank checks automatically. rapport interne. 1997.

[Park 96] H.S. Park & S.W. Lee. *Off-Line Recognition of Large-Set Handwritten Characters with Multiple Hidden Markov-Models.* Pattern Recognition, vol. 29, no. 2, pages 231–244, 1996.

[Park 02] Jaehwa Park. *An Adaptive Approach to Offline Handwritten Word Recognition.* IEEE Trans. on PAMI, vol. 24, no. 7, pages 920–931, 2002.

[Pfister 00] Marcus Pfister, Sven Behnke & Raúl Rojas. *Recognition of Handwritten ZIP Codes in a Real-World Non-Standard-Letter Sorting System.* Appl. Intell., vol. 12, no. 1-2, pages 95–114, 2000.

[Pillet 00] V. Pillet. *Méthodologie d'extraction automatique d'information à partir de la littérature scientifique en vue d'alimenter un nouveau système d'information.* PhD thesis, Université d'Aix-Marseille III, 2000.

[Pinto 03] D. Pinto, A. McCallum, X. Wei & W. B. Croft. *Table extraction using conditional random fields.* Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, pages 235–242, 2003.

[Pitrelli 03] J.F. Pitrelli & M.P. Perrone. *Confidence-Scoring Post-Processing for Off-Line Handwritten-Character Recognition Verification.* ICDAR, vol. 1, pages 278–282, 2003.

[Plamondon 00] R. Plamondon & S.N. Srihari. *On-line and off-line handwriting recognition : A comprehensive survey.* IEEE Trans. on PAMI, vol. 22, no. 1, pages 63–84, 2000.

[Platt 99a] J.C. Platt. *Fast Training of Support Vector Machines using Sequential Minimal Optimization.* Advances in Kernel Methods - Support Vector Learning, MIT Press, pages 185–208, 1999.

[Platt 99b] J.C. Platt. *Probabilities for SV Machines.* Advances in Large Margin Classifiers, MIT Press, pages 61–74, 1999.

[Platt 00] J. Platt, N. Cristianini & J. Shawe-Taylor. *Large Margin DAGS for Multiclass Classification.* NIPS, pages 547–553, 2000.

[Poibeau 02] T. Poibeau. *Extraction d'information à base de connaissances hybrides.* PhD thesis, Université Paris 13 - Villetaneuse, 2002.

[Poisson 05] E. Poisson. *Architecture et apprentissage d'un système hybride neuro-markovien pour la reconnaissance de l'écriture manuscrite en-ligne.* PhD thesis, Université de Nantes, 2005.

[Powalka 94] Robert K. Powalka, Nasser Sherkat & Robert J. Whitrow. *The Use Of Word Shape Information For Cursive Script Recognition.* IWFHR, pages 67–76, 1994.

[Powalka 97] R.K. Powalka, N. Sherkat & R.J. Withrow. *Word shape analysis for hybrid recognition system.* Pattern Recognition, vol. 30, no. 3, pages 421–445, 1997.

[Prevost 98] L. Prevost & M. Milgram. *Coopération pour la Reconnaissance de Caractères Dynamique Isolés.* RFIA, vol. 3, pages 233–240, 1998.

[Prevost 03] L. Prevost, C. Michel-Sendis, A. Moises, L. Oudot & M. Milgram. *Combining model-based and discriminative classifiers : application to handwritten character recognition.* ICDAR, vol. 1, pages 31–35, 2003.

[Procter 98] S. Procter & A.J. Elms. *The recognition of handwritten digit strings of unknown length using Hidden Markov Models.* ICPR, pages 1515–1517, 1998.

[Pérez-Ortiz 01] Juan Antonio Pérez-Ortiz & Mikel L. Forcada. *Part-of-speech tagging with recurrent neural networks.* IJCNN, pages 1588–1592, 2001.

[Quinlan 93] J. R. Quinlan. C4.5 : programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[Rabiner 90] L. R. Rabiner. *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition.* In Readings in Speech Recognition, pages 267–296. Kaufmann, 1990.

[Rahman 97] A.F.R. Rahman & M.C. Fairhurst. *A New Hybrid approach in combining multiple experts to recognize handwritten numeral.* Pattern Recognition Letters, vol. 18, pages 781–790, 1997.

[Rahman 03] A.F.R. Rahman & M.C. Fairhurst. *Multiple classifier decision combination strategies for character recognition : A review.* IJDAR, vol. 5, pages 166–194, 2003.

[Rakotomamonjy 04] A. Rakotomamonjy. *Optimizing AUC with Support Vector Machine.* European Conference on Artificial Intelligence Workshop on ROC Curve and AI, pages 469–478, 2004.

[Ramdane 03] S. Ramdane, B. Taconet & A. Zahour. *Classification of forms with handwritten fields by planar hidden Markov models.* Pattern Recognition, vol. 36, no. 4, pages 1045–1060, 2003.

[Rosenfeld 00] R. Rosenfeld. *Two Decades Of Statistical Language Modeling : Where Do We Go From Here ?* Proceedings of the IEEE, vol. 88, pages 1270–1278, 2000.

[Sadri 04] J. Sadri, C. Y. Suen & T. D. Bui. *Automatic Segmentation of Unconstrained Handwritten Numeral Strings.* IWFHR, pages 317–322, 2004.

[Sadykhov 02] R. Sadykhov, O. Malenko, A.N. Klimovich & M.L. Selinger. *Hybrid System for Recognition of Handwritten Symbols on the Base of Structural Methods and Neural Networks.* VI02, page 223, 2002.

[Sayre 73] K.M. Sayre. *Machine recognition of handwritten words : A project report.* Pattern Recognition, vol. 5, pages 213–228, 1973.

[Schaffer 85] J.D. Schaffer & J.J. Grefenstette. *Multiobjective learning via genetic algorithms.* IJCAI, pages 593–595, 1985.

[Sekita 88] I. Sekita, K. Toraichi, R. Mori, K. Yamamoto & H. Yamada. *Feature Extraction of Handwritten Japanese Characters by Spline Functions for Relaxation Matching.* Pattern Recognition, vol. 21, no. 1, pages 9–17, 1988.

[Seni 94] G. Seni & E. Cohen. *External Word Segmentation of Off-Line Handwritten Text Lines.* Pattern Recognition, vol. 27, no. 1, pages 41–52, 1994.

[Seni 96] G. Seni, R.K. Srihari & N. Nasrabadi. *Large Vocabulary Recognition Of Online Handwritten Cursive Words.* IEEE Trans. on PAMI, vol. 18, no. 7, pages 757–762, 1996.

[Senior 98] A.W. Senior & A.J. Robinson. *An Off-Line Cursive Handwriting Recognition System.* IEEE Trans. on PAMI, vol. 20, no. 3, pages 309–321, 1998.

[Sha 03] F. Sha & F. Pereira. *Shallow Parsing with Conditional Random Fields.* Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics, pages 134–141, 2003.

[Shi 97] Z. Shi, S. N. Srihari, Y.-C. Shin & V. Ramanaprasad. *A System for Segmentation and Recognition of Totally Unconstrained Handwritten Numeral Strings.* ICDAR, page 455, 1997.

[Shridhar 84] M. Shridhar & A. Badreldin. *Recognition of isolated and connected handwritten numerals.* Proc. IEEE International Conference on Systems, Man and Cybernetics, page 142 146, 1984.

[Slavik 01] Petr Slavik & Venu Govindaraju. *Equivalence of Different Methods for Slant and Skew Corrections in Word Recognition Applications.* IEEE Trans. on PAMI, vol. 23, no. 3, pages 323–326, 2001.

[Soderland 94] S. Soderland & W.G. Lehnert. *Wrap-Up : a Trainable Discourse Module for Information Extraction.* Journal of Artificial Intelligence Research, vol. 2, pages 131–158, 1994.

[Srihari 93] Rohini K. Srihari & Charlotte M. Baltus. *Incorporating Syntactic Constraints in Recognizing Handwritten Sentences.* IJCAI, pages 1262–1267, 1993.

[Srihari 97a] Sargur N. Srihari & Edward J. Kuebert. *Integration of hand-written address interpretation technology into the United States Postal Service Remote Computer Reader system.* ICDAR, pages 892–896, 1997.

[Srihari 97b] S.N. Srihari & E.J. Keubert. *Integration of handwritten address interpretation technology into the united states postal service remote computer reader system.* ICDAR, pages 892–896, 1997.

[Srinivas 94] N. Srinivas & K. Deb. *Multiobjective optimization using nondominated sorting in genetic algorithms.* Evolutionnary Computation, pages 221–248, 1994.

[Suen 90] C. Y. Suen, C. C. Tappert & T. Wakahara. *The State of the Art in On-Line Handwriting Recognition.* IEEE Trans. on PAMI, vol. 12, no. 8, pages 787–808, 1990.

[Suwa 04] M. Suwa & S. Naoi. *Segmentation of Handwritten Numerals by Graph Representation.* IWFHR, pages 334–339, 2004.

[Taghva 04] K. Taghva, J.S. Coombs, R. Pereda & T.A. Nartker. *Address extraction using hidden Markov models.* Document Recognition and Retrieval XII. Proceedings of the SPIE, pages 119–126, 2004.

[Takahashi 03] K. Takahashi & D. Nishiwaki. *A class-modular GLVQ ensemble with outlier learning for handwritten digit recognition.* ICDAR, vol. 1, pages 268–272, 2003.

[Tappert 84] C. C. Tappert. *Adaptive on-line handwriting recognition.* ICPR, pages 1004–1007, 1984.

[Tax 01] D.M.J. Tax & Robert P. W. Duin. *Combining One-Class Classifiers.* MCS '01, pages 299–308, 2001.

[Taxt 90] T. Taxt, J. V. Olafsdottir & M. Doehlen. *Recognition of handwritten symbols.* Pattern Recognition., vol. 23, no. 11, pages 1155–1166, 1990.

[Trier 95] O. Trier & A.K. Jain. *Goal-Directed Evaluation of Binarization Methods.* IEEE Trans. on PAMI, vol. 17, no. 12, pages 1191–1201, 1995.

[Trier 96] O.D. Trier, A.K. Jain & T. Taxt. *Feature extraction methods for character recognition : A Survey.* Pattern Recognition, vol. 29, no. 4, pages 641–662, 1996.

[Tsang 98] C. Tsang & F.L. Chung. *Development of a Structural Deformable Model for Handwriting Recognition.* ICPR, vol. 2, pages 1130–1133, 1998.

[Tulyakov 03] S. Tulyakov & V. Govindaraju. *Postal Address Block Location by Contour Clustering.* ICDAR, pages 429–432, 2003.

[Vapnik 95] V. Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1995.

[Verschueren 84] W. Verschueren, B. Schaeken, Y. Rene de Cotret & A. Hermanne. *Structural Recognition of Handwritten Numerals.* ICPR, pages 760–762, 1984.

[Vinciarelli 04] A. Vinciarelli, S. Bengio & H. Bunke. *Offline Recognition of Unconstrained Handwritten Texts Using HMMs and Statistical Language Models.* IEEE Trans. on PAMI, vol. 26, no. 6, pages 709–720, 2004.

[Vuurpijl 03] L. Vuurpijl, L. Schomaker & M. Van Erp. *Architectures for detecting and solving conflicts : two-stage classification and support vector classifiers.* IJDAR, pages 213–223, 2003.

[Wang 88] C.H. Wang, P.W. Palumbo & S.N. Srihari. *Object Recognition in Visually Complex Environments : An Architecture for Locating Address Blocks on Mail Pieces.* ICPR, pages 365–367, 1988.

[Wang 99] X. Wang, V. Govindaraju & S. Srihari. *Multi-Experts for Touching Digit String Recognition.* ICDAR, pages 800–803, 1999.

[Wang 00] X. Wang, V. Govindaraju & S. Srihari. *Holistic recognition of handwritten character pairs.* Pattern Recognition, vol. 33, pages 1967–1973, 2000.

[Wermter 99] S. Wermter, G. Arevian & C.Panchev. *Recurrent Neural Network Learning for Text Routing.* ICANN-99, pages 898–903, 1999.

[Weston 99] J. Weston & C. Watkins. *Multi-class support vector machines for multi-class pattern recognition.* ESANN, pages 219–224, 1999.

[Wong 02] K.W. Wong, C.S. Leung & S.J. Chang. *Handwritten digit recognition using multi-layer feedforward neural networks with periodic and monotonic activation functions.* ICPR, vol. 3, pages 106–109, 2002.

[Wu 06] C-H. Wu, G-H. Tzeng, Y-J. Goo & W-C. Fang. *A real-valued genetic algorithm to optimize the parameters of support vector machine for predicting bankruptcy.* Expert systems with applications Article in Press, 2006.

[Xu 92] L. Xu, A. Kryzak, C.Y. Suen & K. Liu. *Method of combining multiple classifiers and their applications to handwriting recognition.* IEEE Trans. on SMC, vol. 22, no. 3, pages 418–435, 1992.

[Xu 03] Q. Xu, L. Lam & C.Y. Suen. *Automatic segmentation and recognition system for handwritten dates on Canadian bank cheques.* ICDAR, pages 704–708, 2003.

[Xue 06] H. Xue & V. Govindaraju. *Hidden Markov Models Combining Discrete Symbols and Continuous Attributes in Handwriting Recognition.* IEEE Trans. on PAMI, vol. 28, no. 3, pages 458–462, 2006.

[Ye 99] X. Ye, M. Cheriet, C.Y. Suen & K. Liu. *Extraction of bankcheck items by mathematical morphology.* IJDAR, vol. 2, no. 2–3, pages 53–66, 1999.

[Yeh 87] P.S. Yeh, S. Antoy, A. Litcher & A. Rosenfeld. *Address location on envelopes.* Pattern Recognition, vol. 20, no. 2, pages 213–227, 1987.

[Zaragoza 98] H. Zaragoza & P. Gallinari. *An automatic surface information extraction system using hierarchical IR and stochastic IE.* European Conference on Machine Learning : Workshop on Text Mining, 1998.

[Zhang 00] G.P. Zhang. *Neural Networks for Classification : A Survey.* IEEE Trans on Systems, Man and Cybernetics - Part C, vol. 30, no. 4, pages 641–662, 2000.

[Zhang 02] L.Q. Zhang & C.Y. Suen. *Recognition of courtesy amounts on bank checks based on a segmentation approach.* FHR02, pages 298–302, 2002.

[Zhao 00] B. Zhao, Y. Liu & S.W. Xia. *Support Vector Machine and its Application in Handwritten Numeral Recognition.* "ICPR", vol. 2, pages 720–723, 2000.

[Zhou 00] J. Zhou, Q. Gan, A. Krzyzak & C. Suen. *Recognition and verification of touching handwritten numerals.* IWFHR, pages 179–188, 2000.

[Zhou 02] J. Zhou, A. Krzyzak & C.Y. Suen. *Verification - A Method of Enhancing the Recognizers of Isolated and Touching Handwritten Numerals Pattern Recognition.* Pattern Recognition, vol. 35, no. 5, pages 1179–1189, 2002.

[Zhou 05] J. Zhou & C.Y. Suen. *Unconstrained Numeral Pair Recognition Using Enhanced Error Correcting Output Coding : A Holistic Approach.* ICDAR, pages 484–488, 2005.

[Zhu 99] X. Zhu, Y. Shi & S. Wang. *A New Distinguishing Algorithm of Connected Character Image Based on Fourier Transform.* icdar, vol. 00, page 788, 1999.

[Zitzler 99] E. Zitzler & L. Thiele. *Multiobjective evolutionary algorithms : A comparison case study and the strength pareto approach.* Evolutionary Computation, pages 257–271, 1999.

[Zitzler 01] E. Zitzler, M. Laumanns & L. Thiele. *SPEA2 : Improving the strength pareto evolutionary algorithm.* Technical report - Swiss Federal Institute of Technology, 2001.

[Zouari 02] H. Zouari, L. Heutte, Y. Lecourtier & A. Alimi. *Un panorama des méthodes de combinaison de classifieurs en reconnaissance de formes.* RFIA, vol. 2, pages 499–508, 2002.