Une methode dirigée par la syntaxe pour l'extraction de champs numériques dans les courriers entrants

Clément Chatelain, Guillaume Koch, Laurent Heutte, Thierry Paquet

Une méthode dirigée par la syntaxe pour l'extraction de champs numériques dans les courriers entrants
A syntax directed method for numerical field extraction in incoming mail documents

Clément Chatelain, Guillaume Koch,
Laurent Heutte, Thierry Paquet
Laboratoire PSI Université de Rouen
76821 Mt St Aignan Cedex
mail: {clement.chatelain, guillaume.koch,
laurent.heutte, thierry.paquet}@univ-rouen.fr

Résumé

Dans cet article, nous présentons une méthode générique d'extraction et de reconnaissance de champs numériques (numéro de téléphone, code postal, etc.) dans des courriers manuscrits non contraints. La méthode d'extraction exploite la syntaxe des champs comme information a priori pour les localiser. Un analyseur syntaxique à base de modèles de Markov filtre les séquences de composantes qui respectent la syntaxe d'un type de champ connu du système. Notre approche permet ainsi d'éviter la reconnaissance totale du document, opération délicate et coûteuse en temps de calcul, puisque seuls les champs localisés sont soumis à un système de reconnaissance. Nous montrons l'efficacité de la méthode sur une base de courriers manuscrits réels de type courrier entrant.

Abstract

In this article, we propose a generic method for the automatic localisation and recognition of numerical fields (phone number, ZIP code, etc.) in unconstrained handwritten incoming mail documents. The method exploits the syntax of a numerical field as an a priori knowledge to locate it in the document. A syntactical analysis based on Markov models filters the connected component sequences that respect a particular syntax known by the system. Once extracted, the fields are submitted to a numeral recognition process. Hence, we avoids an integral recognition of the document, which is a very tough and time consuming task. We show the efficiency of the method on a real incoming mail document database.

Mots clefs : reconnaissance de l'écriture manuscrite, extraction d'information, champ numérique, réseaux de neurones, modèles de Markov.

Keywords : handwriting recognition, information extraction, numerical field, neural networks, markovian model.

1 Introduction

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, ...), 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 temps de traitement. Dans certains cas, le nombre de documents traités dépassent 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 1) restent à ce jour extrêmement difficile à traiter.

En effet, si la lecture des documents dactylographiés peut être considérée comme un problème résolu, la lecture des documents manuscrits pose encore un certain nombre de problèmes à la communauté scientifique [19, 21]. La lec-

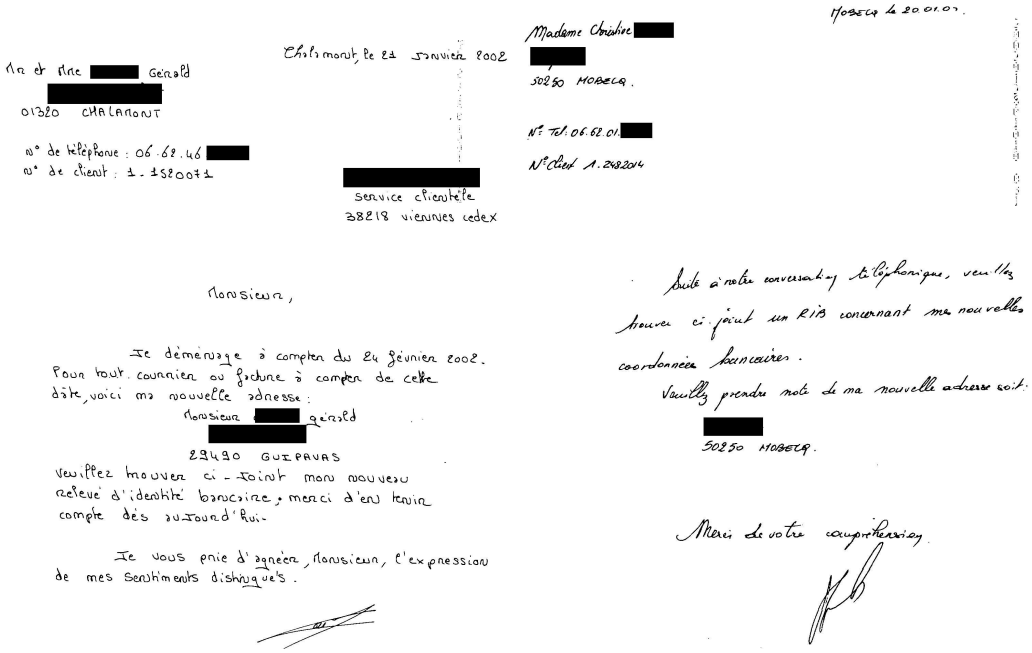

FIG. 1 – Exemples de courriers manuscrits libres.

ture automatique de documents manuscrits n'est possible que dans quelques cas applicatifs particuliers (lecture automatique de chèques ou d'adresses postales, reconnaissance des champs d'un formulaire, analyse de factures), car le contenu de ces documents est largement contraint. Pour les adresses postales, la structure de l'adresse est peu variable et contient généralement les champs prénom, nom, numéro et nom de rue, code postal et nom de ville. Pour les chèques, la position des montants est approximativement connue, et les deux montants (montant chiffre et montant littéral) sont redondants, ce qui permet une étape de vérification [13, 15] ; de plus, le vocabulaire est relativement réduit (30 mots pour le montant littéral des chèques). Le cas des formulaires ou des factures est également largement contraint puisqu'on impose au scripteur une écriture scripte et précasée ; on connait donc la position et la nature des champs d'intérêt. Dans toutes ces applications, le système de lecture possède donc une information *a priori* importante (redondance de l'information, position des zones d'intérêt, lexique limité, contenu attendu, etc.) qui permet de détecter les incohérences possibles du module de reconnaissance de symboles, ce qui autorise le traitement automatique de ces documents.

Peu de travaux abordent des problèmes de reconnaissance moins contraints

car il est plus difficile de bénéficier de moyens automatiques de vérification des hypothèses de reconnaissance. Contrairement aux applications précédemment citées, aucune information *a priori* n'est disponible, ce qui rend la tâche de lecture délicate. Si la lecture intégrale de ces documents est actuellement inenvisageable, il est cependant possible de considérer des problèmes de lecture partielle, visant à extraire l'information pertinente du document. C'est ce que nous envisageons dans cet article en proposant une méthode générique d'extraction et de reconnaissance de champs numériques (code postal, numéro de client, numéro de téléphone, etc.) dans des courriers manuscrits. Nous apportons ainsi des éléments de solution à la problématique du traitement automatique du «courrier entrant».

L'article est organisé de la manière suivante : la section 2 pose la problématique de la localisation des champs numériques dans les courriers entrants manuscrits libres. Nous justifions l'emploi de notre approche "dirigée par la syntaxe" par rapport à d'autre stratégies de reconnaissance. La section 3 est consacrée à la formalisation du problème. La section 4 donne une vue globale du système et introduit les différents modules de la chaîne de traitement : segmentation en ligne (section 5), classification des composantes (section 6), localisation des champs par analyse syntaxique (section 7), et reconnaissance des champs (section 8). Nous discutons dans la section 9 des résultats obtenus sur une base de 300 courriers manuscrits réels. Enfin nous concluons dans la section 10 sur les perpectives de notre approche.

2 Analyse du problème et justification de la méthode

2.1 Analyse du problème

On peut définir un champ numérique comme une séquence de chiffres apportant une information sur l'expéditeur du courrier. Par exemple, un numéro de téléphone peut permettre d'identifier l'expéditeur, un code postal donne une indication géographique, un code client permet de connaître le type de contrat et d'acheminer le courrier au service concerné dans l'entreprise. Nous souhaitons extraire ces champs numériques d'un document manuscrit de type courrier entrant (figure 2). La figure 3 montre quelques exemples de champs numériques. Notons que nous nous focalisons exclusivement sur les champs strictements numériques tels que les numéros de téléphone ou codes postaux ; nous ne nous intéressons donc pas à la détection de dates par exemple, pour lesquelles le mois peut être écrit en toutes lettres.

Les exemples présentés (figures 1, 2) montrent que les champs numériques

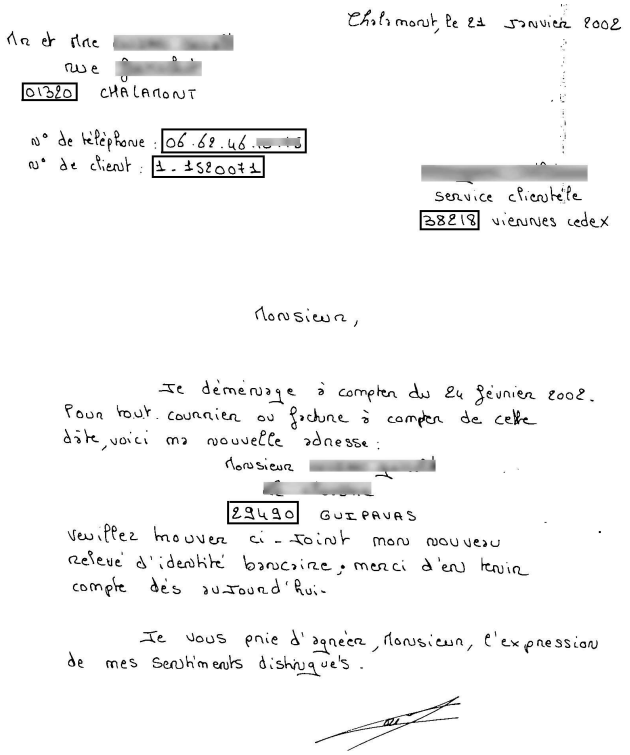

FIG. 2 – Extraction des champs numériques dans un courrier manuscrit.

que l'on souhaite extraire peuvent être situés à différents endroits dans le document : en-tête et corps du texte principalement. De ce fait, l'approche développée doit être capable de localiser des champs numériques dans le cas le plus général, c'est à dire dans des lignes de texte quelconques comportant à la fois des informations non numériques (textuelles) et numériques.

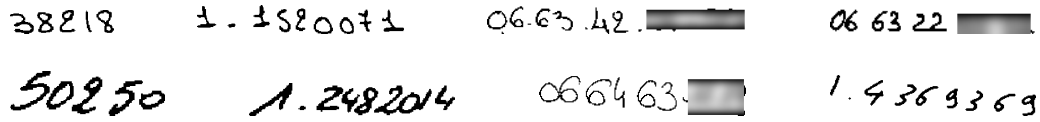

FIG. 3 – Exemples de champs numériques.

2.2 Justification de la méthode

Plusieurs méthodes pourraient permettre de localiser et reconnaître les champs numériques dans un document manuscrit non contraint : détection de tous les chiffres dans le document ou reconnaissance intégrale du courrier manuscrit. Nous considérons dans cette section ces deux approches et montrons qu'elles soulèvent des problèmes difficilement surmontables. Nous introduisons alors une troisième approche "dirigée par la syntaxe", que nous avons retenue.

2.2.1 Identification des composantes numériques à l'aide d'un classifieur chiffre.

Une première solution pour extraire les champs numériques consisterait à identifier tous les chiffres du document, sans procéder à une segmentation préalable des composantes. Les chiffres pourraient être identifiés en soumettant toutes les composantes connexes du document à un classifieur chiffre 10 classes (0..9) capable de rejeter les composantes non numériques. L'opération de rejet pourrait être effectuée en utilisant par exemple le score de confiance fourni par le classifieur [26, 14]. La figure 4 donne un exemple de résultat obtenu avec une telle méthode : un classifieur chiffre de type MLP est passé sur toutes les composantes du document, et un seuillage est effectué sur le score de la première proposition du classifieur. Les composantes dont le score est supérieur à un certain seuil sont considérées comme fiables et donc conservées comme hypothèse "chiffre". Les autres, dont le score est inférieur à ce seuil, doivent donc être rejetées (notées "R"). On constate que si les chiffres isolés sont en majorité correctement identifiés, les chiffres liés sont logiquement rejetés (Cf. le code postal de la figure 4).

L'incapacité de cette méthode à détecter la présence de chiffres liés interdit donc son utilisation pour notre problème puisque seule une segmentation des composantes peut permettre d'identifier tous les chiffres d'un document : chiffres isolés et chiffres liés.

2.2.2 Reconnaissance intégrale du document

Une deuxième méthode pour localiser et reconnaître les champs numériques consisterait à reconnaître l'intégralité du document. Les méthodes les plus fiables développées actuellement pour la reconnaissance d'une ligne de texte consistent à mettre en oeuvre une stratégie de segmentation-reconnaissance, pour aligner le treillis de reconnaissance sur un modèle de ligne plus ou moins contraint [22, 17, 23, 29]. Dans [22] par exemple, les auteurs proposent une modélisation par modèles de Markov cachés d'une ligne de texte dans une

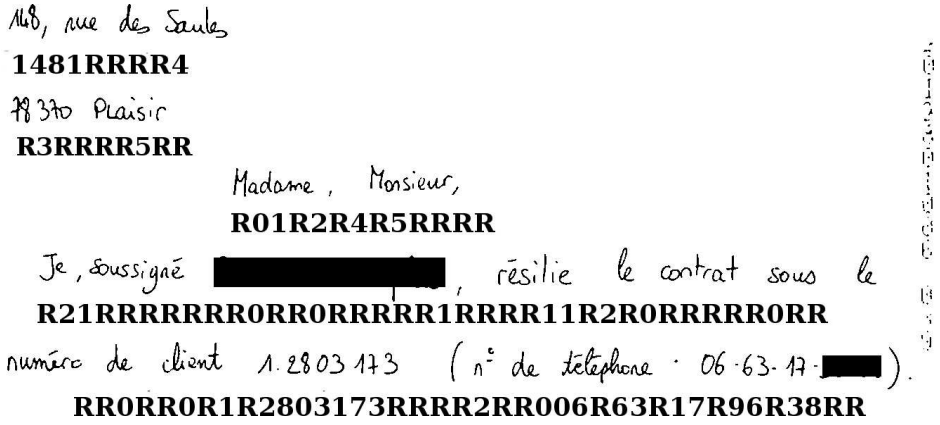

FIG. 4 – Exemple de résultats fournis par un classifieur chiffre appliqué sur toutes les composantes connexes d'un extrait de document. Au dessus d'un certain seuil de confiance de la première proposition du classifieur(ici 75%), les composantes sont considérées comme des chiffres, en dessous, elles sont considérées comme du rejet ('R').

adresse postale, en vue de localiser et reconnaître un nom de rue. Le modèle 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 telles que le numéro de rue, la nature de la voie, etc.

Imaginons la mise en place d'une telle stratégie pour notre problème. La première étape consisterait à effectuer une sur-segmentation des composantes en graphèmes de manière à constituer un treillis des graphèmes de niveau 1 à N (où un graphème de niveau N désigne l'aggrégation de N graphèmes élémentaires). Le treillis de graphèmes serait soumis à un moteur de reconnaissance qui devrait être capable d'identifier toutes les entités du document. On obtiendrait ainsi un treillis de reconnaissance comportant plusieurs hypothèses de segmentation. L'alignement du treillis de reconnaissance sur un modèle de ligne donné fournirait l'hypothèse de reconnaissance la plus probable. Dans notre cas, les documents ne sont pas contraints ; il serait donc nécessaire de définir un modèle de ligne suffisamment complexe pour absorber n'importe quelle information, numérique ou textuelle. Dans les approches citées ci-dessus, les modèles de lignes sont généralement obtenus par concaténation de modèles de mots, pouvant eux-mêmes être générés par concaténation de modèles de lettres. En intégrant dans ces modèles de lignes des états "chiffres" permettant d'absorber les composantes numériques, on pourrait modéliser l'intégralité du document. L'alignement d'une ligne de texte sur un tel modèle permettrait ainsi d'isoler les composantes numériques.

La mise en oeuvre d'une telle solution soulève toutefois un certain nombre de problèmes : premièrement, elle nécessite le développement d'un classifieur capable de reconnaître toutes les entités que l'on peut rencontrer dans un document manuscrit non contraint : lettres minuscules et majuscules, chiffres, ponctuation. Même si des travaux sur la réalisation d'un tel classifieur ont été menés [27], il est difficile d'obtenir des performances intéressantes. La réalisation d'un modèle de ligne générique capable de prendre en compte la très grande variabilité de l'écriture manuscrite semble également délicate puisque la modélisation des mots d'un dictionnaire de très grande taille est requise. Enfin signalons qu'une telle méthode est particulièrement gourmande en temps de calcul puisqu'un classifieur doit procéder à la reconnaissance de tous les regroupements de graphèmes.

2.2.3 Approche dirigée par la syntaxe

Contrairement aux deux premières approches où la localisation et la reconnaissance des champs sont étroitement liées, nous envisageons la mise en oeuvre d'une méthode de reconnaissance en deux étapes :

- La première étape est chargée de localiser les champs numériques sans faire appel à une segmentation des composantes connexes ni 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.

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 segmentation des composantes connexes ni à la reconnaissance des chiffres. En effet, on peut voir sur la figure 3 que chaque type de champs (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 (champs

numérique et rejet).

Supposons qu'une segmentation du document en lignes ait été effectuée, on dispose alors de la succession des composantes connexes de chaque ligne. C'est cette séquence que l'on va chercher à interpréter globalement 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 même 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 qu'on pourrait également chercher à détecter les chiffres liés comportant plus de deux chiffes, 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. La figure 5 montre un exemple d'étiquetage des composantes connexes d'une ligne de texte extraite d'un document.

FIG. 5 – Exemple d'étiquetage des composantes d'une ligne comprenant un code client. R : Rejet, D : Digit, S : Séparateur, DD : Double Digit.

Ces quatres classes constituent les états du modèle markovien, sur lequel on alignera les séquences de composantes reconnues, de manière à ne conserver que les séquences syntaxiquement correctes. En disposant d'un classifieur capable de discriminer ces 4 classes, l'extraction 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. La recherche du meilleur alignement dans le treillis est effectuée par l'algorithme de Viterbi [2, 4].

Une fois les champs localisés, un moteur de reconnaissance de champs est chargé de les reconnaître.

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.

3 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.

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 modelisée 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 fournissent 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.

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 7.

4 Description générale de la méthode

Comme nous venons de le voir, la méthode repose sur l'utilisation de la syntaxe d'un champ numérique comme information *a priori* pour le localiser. Pour cela, nous nous appuyons sur un modèle général d'une ligne d'écriture fondé sur un modèle markovien.

Nous avons vu que la mise en oeuvre de cette approche nécessitait les modules suivants (voir figure 6) :

Segmentation en lignes : le module de segmentation du document en ligne est le point d'entrée du système. Les lignes de texte sont extraites grâce à une approche de regroupement des composantes connexes. Nous décrivons cette étape dans la section 5.

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 caractérisation des composantes est réalisée à l'aide de deux jeux de caractéristiques présentés à deux classifeurs. Nous combinons ensuite les résultats. Cette étape est décrite dans la section 6.

Analyse syntaxique : cette dernière 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

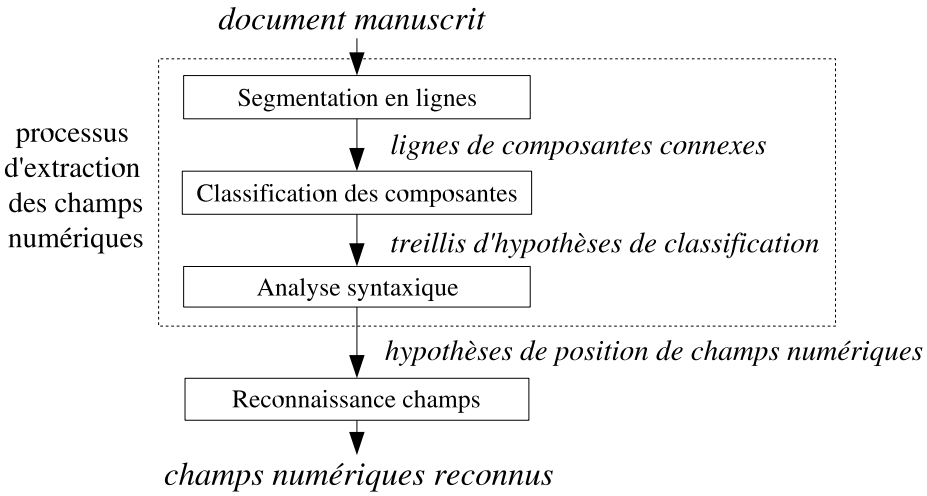

FIG. 6 – Chaine globale de traitement pour l'extraction et la reconnaissance de champs numériques.

en alignant les hypothèses 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 7.

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 performant et une méthode de segmentation de chiffres liés.

Nous décrivons maintenant ces quatre modules.

5 Segmentation en lignes

Bien que la détection des lignes de texte soit le point d'entrée du système, elle est basée sur une approche plutôt classique. Nous en décrivons sommairement les principales étapes. L'approche utilisée est inspirée de [10] où une méthode de détection des lignes de texte sans connaissance a priori de leur orientation est présentée. Comme nos documents sont relativement bien structurés, la méthode a été modifiée de manière à privilégier les alignements horizontaux. Les principales étapes sont les 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 7 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 7 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 7 c).

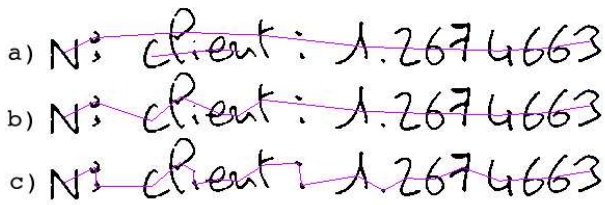

FIG. 7 – Etapes de la segmentation en ligne : a) regroupement initial des composantes connexes, b) fusion des alignements trop proches, c) attachement des composantes isolées à la ligne la plus proche.

6 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 reconnaître des chiffres, mais de classer les composantes selon les 4 classes définies dans la section 4 : Digit, Séparateur, Double Digit et Rejet («D», «S», «DD» et «R»). Cette tâche est relativement difficile puisqu'il s'agit d'un problème où les classes possèdent une grande variabilité intra classe. En effet, la classe "double digit" contient théoriquement les 100 doublons qu'on peut former avec 10 chiffres, ces chiffres étant liés de différentes manières (liaison haute, basse, double, etc., voir figure 8) ; la classe "rejet" contient également des éléments dont la forme est très variable puisqu'elle peut contenir des mots entiers ou des fragments de mot, des lettres isolées, etc. D'autre part, nous sommes confrontés à un recouvrement important entre certaines classes : il peut être difficile de discriminer certains fragments de mots des chiffres ou des chiffres liés.

FIG. 8 – Exemples de chiffres liés : remarquons la diversité des formes et des types de liaisons

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

6.1 Caractérisation des composantes

Parmi les nombreuses méthodes d'extraction de caractéristiques disponibles dans la littérature [12], 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 [7]. 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 9). Nous considérons un voisinage 8-connexe et un pavage $4 * 4$, ce qui fournit un vecteur à 128 caractéristiques.

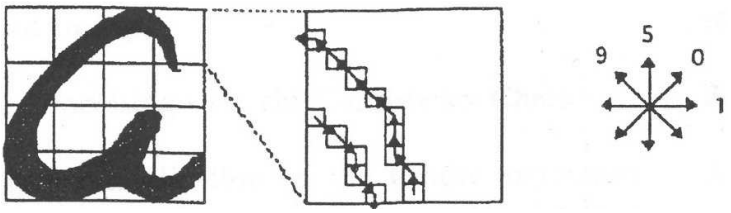

FIG. 9 – Image pavée, extraction du coutour et histogramme des directions du contour en 8-connexité sur un des pavé (l'image provient de [7]).

- Nous utilisons également le vecteur statistique/structurel développé dans nos travaux antérieurs [16]. 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 robuste de caractères manuscrits tels que les chiffres, lettres majuscules et même graphèmes [16].

6.2 Classification

Le Perceptron Multicouche ou MLP (MultiLayer Perceptron) [8] est un classifieur statistique de type "réseau de neurones" largement utilisé dans le domaine de la reconnaissance des caractères manuscrits [3]. Le MLP étant extrêmement rapide durant la phase de décision, cela nous autorise à utiliser un assez grand nombre de caractéristiques. Un autre avantage du MLP est qu'il fournit des estimations de probabilités *a posteriori* en sortie si celui ci est entraîné avec une fonction coût des moindres carrés [5]. C'est une propriété intéressante pour alimenter l'analyseur syntaxique qui analyse des séquences grâce à l'algorithme de Viterbi.

Nous avons donc conçu un MLP pour chaque vecteur de caractéristiques. Appelons "MLP128" le classifieur MLP entrainé sur le vecteur chaincode et "MLP117" le classifieur entraîné sur 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'entrees*+ *nombre de sorties*)/2 pour les deux réseaux.
- Une couche de sortie composée d'autant de neurones que de classes, soit 4.
- La fonction d'activation utilisée est une sigmoïde.

Les deux MLP sont entraînés avec l'algorithme itératif de rétropropagation du gradient, avec un pas adaptatif de type «line search» [8] qui minimise l'erreur à chaque itération. Le tableau 1 donne les effectifs des classes pour la base d'apprentissage et la base de test.

effectifs	R	D	S	DD	Total
Base d'apprentissage	7008	4968	522	334	13000
Base de test	3609	2559	268	171	6000

TAB. 1 – Effectifs des classes pour la base d'apprentissage et la base de test.

Les taux de reconnaissance moyens sur la base de test sont présentés dans la table 2 pour les deux classifieurs. Il semble que MLP128 donne les meilleures performances ; cependant il convient d'observer le comportement des classifieurs sur chaque classe.

Les matrices de confusion des deux classifieurs sur la base de test sont présentées dans les tables 3 et 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

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

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

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 4 classes : nous avons donc combiné leur 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. 3 – Matrices de confusion MLP128.

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 – Matrices de confusion MLP117.

6.3 Combinaison de classifieurs

Nous disposons de deux classifieurs pour discriminer les quatre classes Rejet - Digit - Séparateur - Double Digit. Comme nous avons pu le constater dans la section précédente, ces deux classifieurs ont des performances intéressantes, mais aussi des comportements complémentaires. Il semble donc naturel de les combiner afin de tirer le meilleur parti de leurs spécificités.

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 [24, 28]. Il existe plusieurs méthodes de combinaison de classifieurs, applicables en fonction de la nature de l'information à combiner [6] : 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 [28] : 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 5 donne les taux de reconnaissance (TR) en rang 1, 2 et 3 sur la base de test.

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. 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).

Ces résultats montrent la nette supériorité des combinaisons de classifieurs sur les classifieurs simples. En revanche, il est difficile de déterminer la meilleure méthode de fusion d'après ces résultats. Nous conservons donc la combinaison des classifieurs MLP117 et MLP128 comme moyen pour reconnaître les composantes. Le choix de la meilleure méthode de fusion sera fait lors de l'étape de test d'extraction des champs (voir section 9).

A l'issue de cette étape, nous disposons d'un système capable de discriminer les 4 classes. Pour chaque composante, le classifieur fournit une liste ordonnée des hypothèses de classification associées chacune à une mesure de confiance. La figure 10 montre un exemple de classification des composantes d'une ligne de texte par notre système. Il est évident que ce classifieur n'est pas parfait, et qu'il génère un certain nombre de confusions. Le rôle de la prochaine étape va donc être de corriger ces confusions en alignant la ligne de composantes sur un modèle de syntaxe valide.

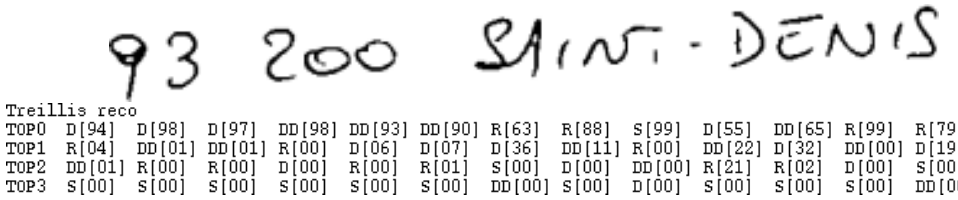

FIG. 10 – Hypothèses de classification des composantes d'une ligne de texte contenant un code postal

7 Analyseur syntaxique

Nous discutons dans cette partie de la construction et de l'apprentissage des modèles de Markov cachés 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 code postal français à cinq chiffres. Nous considérons cinq états "Digit", ainsi que quatre états "Double Digit", correspondant aux regroupements possibles entre deux chiffres consécutifs. 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 $N = 10$. Le nombre d'observations M correspond aux étiquettes que peuvent prendre les composantes : $M = 4$.

La construction de la matrice $A = \{a_{ij}\}$ des probabilités de transitions de l'état i vers l'état j est généralement obtenue par un apprentissage effectué grâce à l'algorithme itératif de Baum-Welch [4]. Ici, les transitions sont déterminées par une estimation statistique sur une base annotée. Afin de limiter le nombre de transitions des modèles, seules les transitions non négligeables sont conservées. Par exemple, le deuxième état "Double Digit" peut être supprimé du modèle d'un code postal car un code postal est généralement écrit en laissant un espace entre le deuxième et le troisième chiffre. Le nombre d'états conservés dans le modèle final est donc 9. On peut ainsi représenter le modèle syntaxique d'une ligne de texte contenant un code postal (voir figure 11).

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 (il suffit de comptabiliser les étiquettes des premières composantes des lignes contenant un champ numérique).

Les modèles de Markov décrivant une ligne de texte contenant un numéro

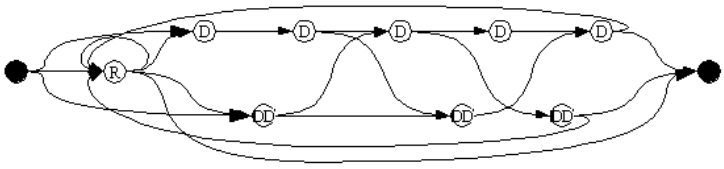

FIG. 11 – Modèle de Markov simple pour une ligne de texte contenant un code postal. Les probabilités de transition non nulles entre états sont représentées par des flêches.

de téléphone et un code client sont obtenus de la même manière (voir figures 12 et 13).

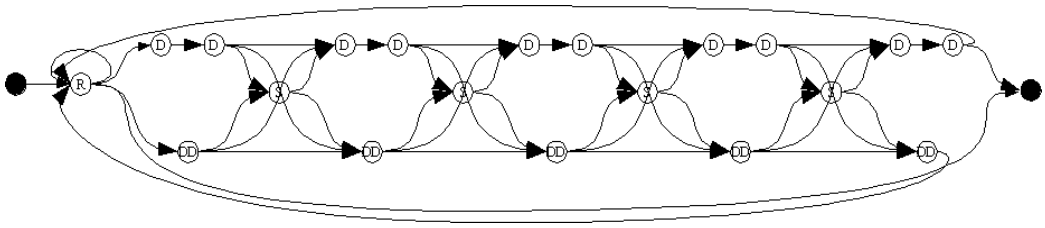

FIG. 12 – Modèle de Markov pour une ligne de texte contenant un numéro de téléphone

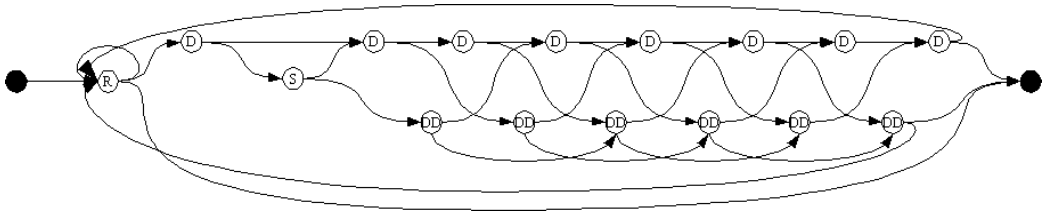

FIG. 13 – Modèle de Markov pour une ligne de texte contenant un code client

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 14 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.

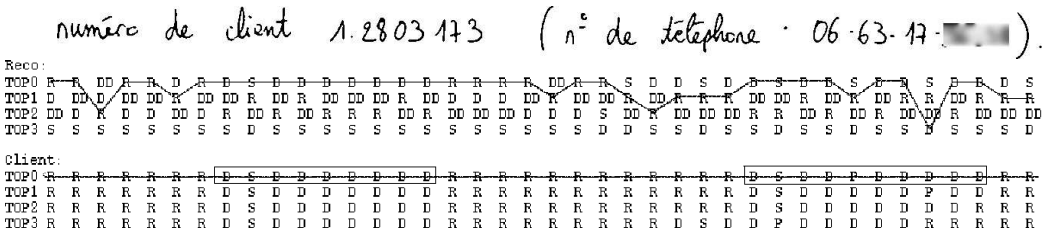

FIG. 14 – Alignement des hypothèses de classification des composantes d'une ligne de texte manuscrit sur le modèle syntaxique d'une ligne de texte contenant un code client.

Réciproquement, la figure 15 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.

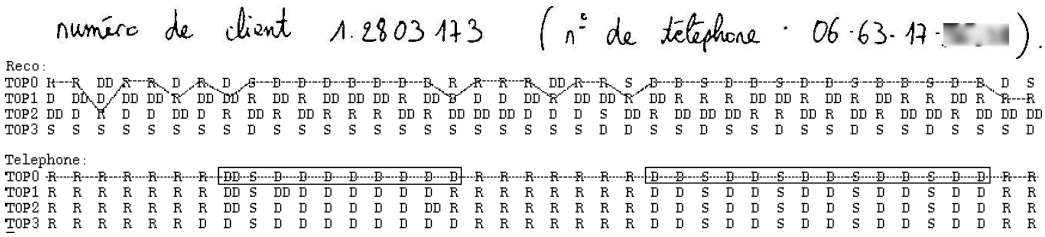

FIG. 15 – Alignement des hypothèses de classification des composantes d'une ligne de texte manuscrit sur le modèle syntaxique d'une ligne de texte contenant un numéro de téléphone.

Nous avons présenté dans cette partie la construction des modèles de Markov permettant l'extraction des champs numériques dans un document sans reconnaissance chiffre ni segmentation. L'étape suivante consiste à soumettre ces hypothèses de localisation au module de reconnaissance de champs numériques.

8 Reconnaissance des champs

Rappelons que la méthode présentée pour la reconnaissance des champs numériques est constituée de deux étapes : la première étant chargée de

les localiser, la seconde de les reconnaître. Nous avons présenté la méthode d'extraction exploitant la syntaxe des champs pour les localiser, nous nous focalisons maintenant sur la reconnaissance de ces champs. Cette étape permettra d'obtenir la valeur numérique des champs, mais aussi de rejeter une partie de la fausse alarme générée par l'étape d'extraction.

8.1 Principe

Contrairement à la majorité des 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 nous a permis de localiser les champs numériques sans les reconnaître. La reconnaissance intervient donc en fin de traitement et permet la 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 16).

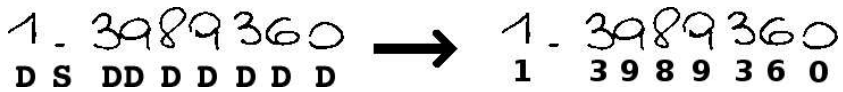

FIG. 16 – Détermination des hypothèses de classification *chiffre* à partir des hypothèses de classification *Digit, Séparateur, Double Digit.*

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". La description du classifieur chiffre est présentée dans la section 8.2. 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. Cette étape est présentée dans la section 8.3. Dans la section 8.4, nous présentons les résultats obtenus sur les champs numériques isolés.

8.2 Classifieur chiffre

La reconnaissance de chiffres isolés a bénéficié de très nombreux travaux ces dernières années, notamment dans le cadre de la reconnaissance de montants numériques de chèques, de champs numériques dans les formulaires, ou encore de reconnaissance de codes postaux dans les adresses postales [21]. Ces systèmes sont basés sur de nombreux extracteurs de caractéristiques [12] et des classifieurs performants [20]. Les systèmes donnant les meilleurs résultats sont généralement conçus autour de classifieurs de type réseau de neurones ou, plus récemment, de machines à vecteur de support (SVM). 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. C'est ce que nous avons fait en reprenant le même système de reconnaissance que dans la partie 6.1 : extraction de deux vecteurs de caractéristiques (statistique/structurel et histogramme des directions du chain-code) soumis à deux réseaux de neurones de type MLP, dont les sorties sont combinées par la suite.

Les deux réseaux possèdent la même topologie que pour la discrimination 4 classes. Ils ont été entrainés sur une base de 115000 chiffres manuscrits étiquetés provenant de formulaires. La base de test permettant de contrôler les apprentissages des réseaux et la base de validation sont constituées chacune de 39000 éléments. Les performances de notre classifieur sont donnés dans le tableau 6.

taux de reconnaissance	RANG1	RANG2	RANG3
MLP128	95,76	98,39	99,18
MLP117	96,87	98,80	99,38
combinaison	98,00	99,24	99,65

TAB. 6 – Taux de reconnaissance en première, deuxième et troisième proposition du classifieur chiffre.

8.3 Reconnaissance des chiffres liés

Nous avons présenté le classifieur chiffre permettant de reconnaître les chiffres isolés, nous nous intéressons dans cette partie à la reconnaissance des composantes dont l'hypothèse de classification lors de la première étape est "chiffres liés" (DD). Une stratégie pour cette opération pourrait être la reconnaissance globale de la composante à l'aide d'un classifieur "chiffres liés" comportant autant de classes que de combinaison possibles, soit 100 (classifieur

100 classes [00..99]). Cette stratégie nécessite toutefois une base d'apprentissage conséquente comportant un nombre suffisamment élevé d'éléments dans chaque classe, afin de couvrir la variabilité inhérente à un problème réel : différents type d'écriture, nature des liaisons entre les chiffres (liaisons hautes, basses, multiples), etc. La conception d'un tel classifieur est donc a priori difficile à envisager. Nous avons donc orienté notre approche vers une segmentation de la composante pour identifier les deux chiffres qui la constituent. Dans la mesure où il est très difficile de déterminer sans reconnaissance le meilleur chemin de coupure pour séparer deux chiffres liés, nous avons mis en oeuvre une stratégie de segmentation-reconnaissance à l'échelle de la composante. Plusieurs hypothèses de segmentation sont générées et soumises au classifieur chiffre qui se prononce sur les deux chiffres. Le choix de la meilleure hypothèse est déterminé à partir des confiances fournies par le classifieur. Cette stratégie repose donc sur la mise en oeuvre d'un module de segmentation permettant la génération de plusieurs chemins de coupures, et sur un module de décision qui se prononce sur le choix du meilleur chemin de segmentation. Nous décrivons maitenant ces deux étapes.

8.3.1 Segmentation des composantes

Il existe de très nombreuses méthodes de segmentation explicite, généralement basées sur l'analyse des contours [11]. Nous avons utilisé une méthode de segmentation inspirée de l'algorithme "drop fall" [9, 18], qui consiste à segmenter la composante selon 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 17).

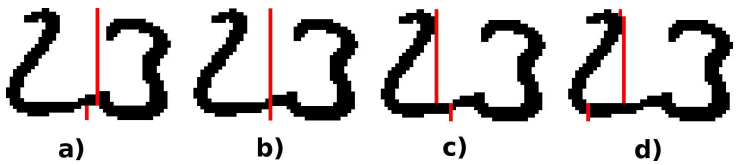

FIG. 17 – Les 4 variantes de l'algorithme "drop fall" : a) ascendant gauche b) ascendant droit c) descendant gauche d) descendant droit.

Un paramètre déterminant de cet algorithme est le point de départ de la

chute de la goutte. Dans [9], 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 18 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 18 b).

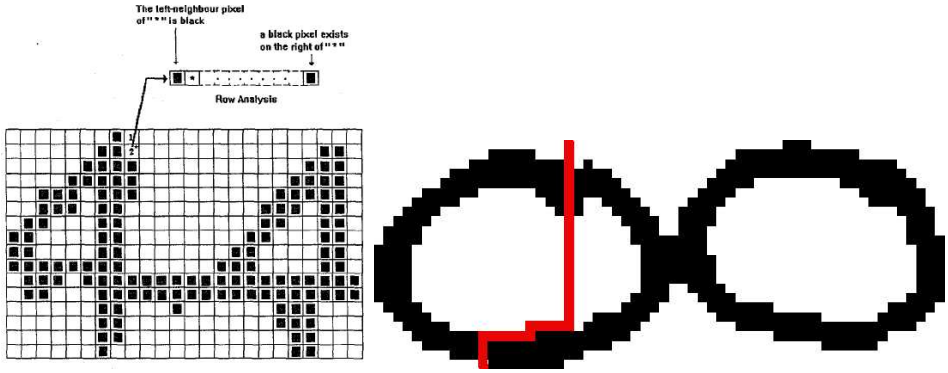

FIG. 18 – a) Point de départ pour l'algorithme drop fall par Congedo [9]. b) échec de la segmentation avec une telle initialisation pour un drop fall descendant.

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" [25] de la composante. Un "water reservoir" est une métaphore pour illustrer la région où les chiffres se touchent (voir figure 19). Les réservoirs sont obtenus en considérant les zones résultantes d'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".

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 19) : 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 20).

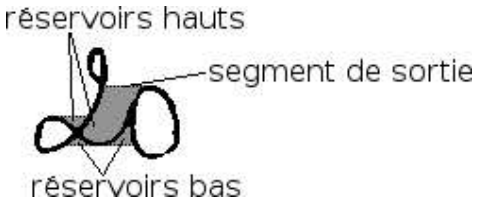

FIG. 19 – "Water reservoirs" haut et bas d'une composante. Le segment de sortie correspond à la surface extérieure du réservoir.

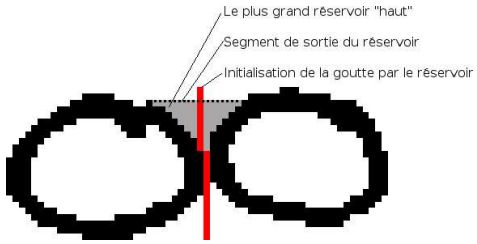

FIG. 20 – Segmentation "drop fall" descendant avec initialisation par les réservoirs.

8.3.2 Sélection du meilleur chemin de segmentation

Nous pouvons donc générer quatre chemins de coupure selon les variantes du drop fall présentées précédemment. Il s'agit dans ce module de sélectionner le "meilleur" chemin parmi les hypothèses générées, décision pour laquelle il nous faut définir un critère fiable traduisant la qualité de la segmentation. Nous proposons de soumettre chaque paire de composantes segmentées à notre classifieur chiffre, et d'utiliser un critère basé sur les scores de confiance associés aux propositions du classifieur. Nous justifions ce choix en réalisant l'expérience suivante : nous soumettons au classifieur chiffre de la partie 8.2 un ensemble comportant environ 2000 chiffres et 4000 rejets (composantes textuelles, chiffres liés, chiffres mal segmentés, bruit) et nous analysons le score de confiance associé à la première proposition. La figure 21 montre que lorsque la confiance associée à la première proposition du classifieur chiffre est forte, la probabilité d'avoir un chiffre est forte, et la probabilité d'avoir un rejet est faible ; et inversement. Il est donc raisonnable de penser que si les chiffres liés sont bien segmentés, les confiances associées aux deux premières propositions seront élevées. 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. La figure 22 présente la segmentation et la reconnaissance d'une composante "double di-

git" selon les quatre variantes du "drop fall" ; ici le drop fall ascendant gauche maximise le produit des confiances, cette hypothèse est donc conservée.

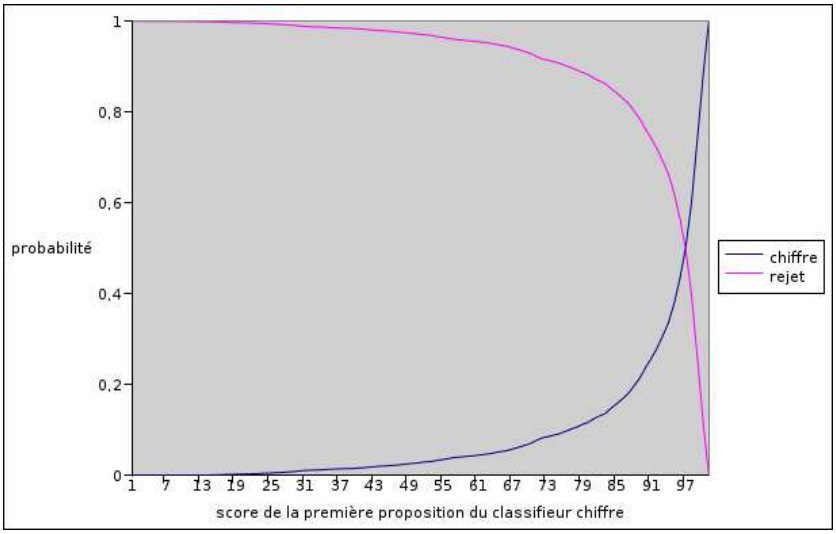

FIG. 21 – Probabilité qu'une entité soit un chiffre ou un rejet en fonction du score de la première proposition du classifieur chiffre.

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

FIG. 22 – 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.

8.3.3 Résultats

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 90%.

8.4 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 7 donne les taux de reconnaissance au niveau champs.

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

TAB. 7 – Taux de reconnaissance des champs isolés.

9 Résultats

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.

Nous présentons dans un premier temps les résultats intermédiaires obtenus à l'issue de l'étape d'extraction des champs, puis nous donnons les résultats en rappel-précision à la fin de la chaîne de traitement. Nous présentons enfin un module de vérification des champs permettant d'améliorer la précision du système.

9.1 Résultats à l'issue de l'extraction des champs.

La détection des champs numériques est réalisée en effectuant l'analyse de chaque ligne d'un document. L'analyseur syntaxique étiquette l'ensemble des

composantes de la ligne en cours d'analyse. La séquence d'étiquette 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 23).

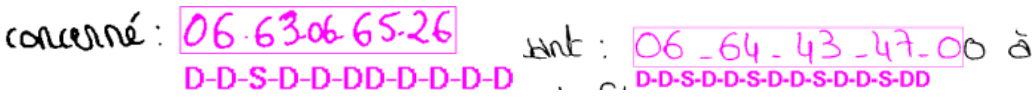

FIG. 23 – Exemples de champs considérés comme bien et mal détectés. L'alignement du deuxième champ est faux puisqu'il manque la dernière composante : nous le comptabilisons comme une non détection, mais aussi comme une fausse alarme.

L'analyseur syntaxique propose une liste de N solutions correspondant aux N meilleurs chemins dans le treillis d'observations. Par conséquent, un champ "bien détecté au rang N" signifie que la bonne hypothèse de localisation du champ figure parmi les N premières solutions de l'analyseur syntaxique.

La figure 24 présente un exemple de détection de codes postaux, numéros de téléphone et codes client. 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'analyseur sont susceptibles d'être soumis à un moteur de reconnaissance.

Le tableau 8 donne les taux de détection des champs en rang 1, 2 et 5 pour les deux types de combinaison de classifieurs.

Classifieur	codes postaux	téléphones	codes client
prod(MLP117, MLP128)	69 / 81 / 89	75 / 81 / 91	81 / 89 / 94
mean(MLP117, MLP128)	66 / 78 / 91	70 / 79 / 89	78 / 87 / 91

TAB. 8 – Taux de détection en rang 1/ rang 2/ rang 5 pour les deux types de combinaison de classifieurs

La règle de combinaison "produit" donne les meilleurs taux de détection pour les trois types de champ, quel que soit le rang considéré. On peut expliquer ce résultat en considérant le comportement des deux opérateurs : l'opérateur mean est capable de fournir une confiance assez élevée même si les classifieurs ne fournisent pas tous les deux la bonne proposition en premier

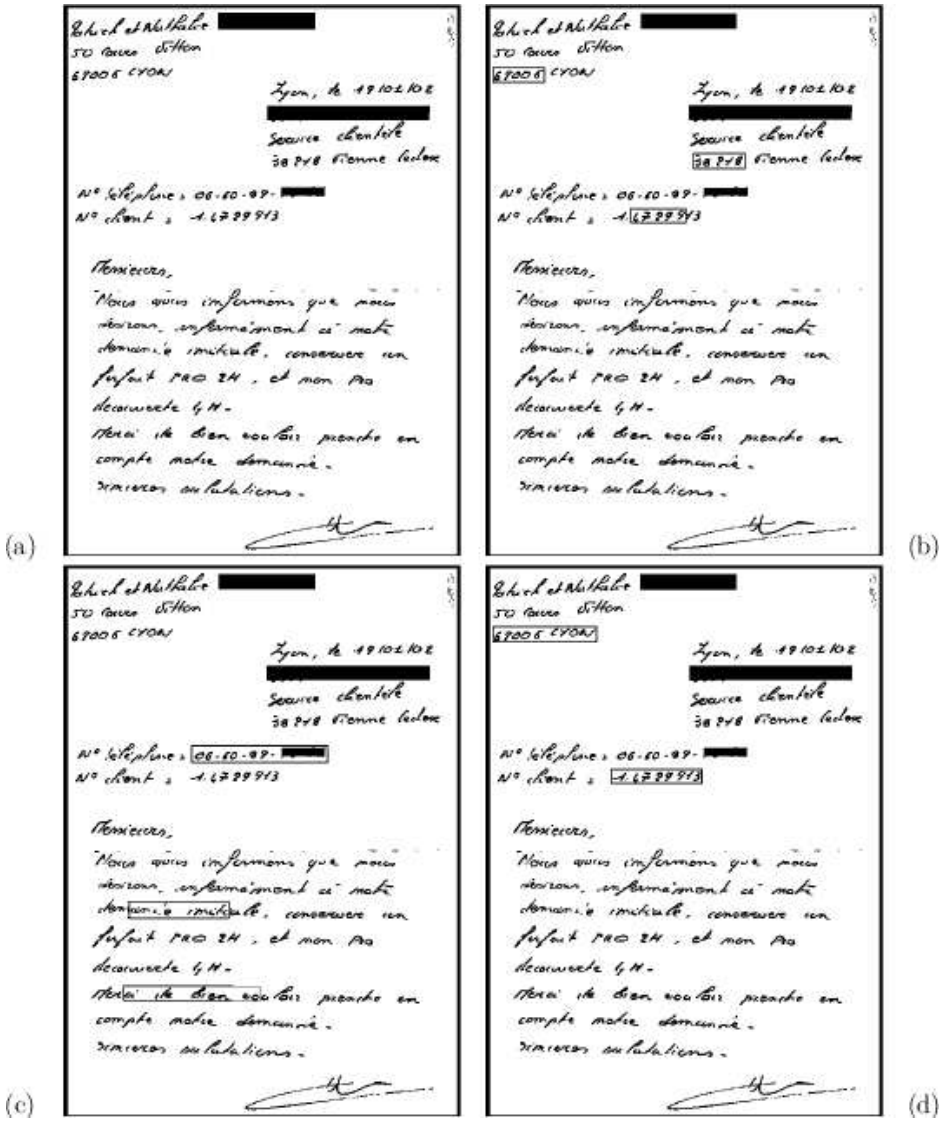

FIG. 24 – (a) Image originale ; (b) codes postaux détectés (séquences encadrées), (c) numéros de téléphone, (d) codes clients.

choix. En revanche, l'opérateur produit fournit une confiance globale élevée dans le seul cas où les deux confiances d'entrée sont élevées ; il pénalise donc fortement les hypothèses pour lesquelles au moins un des classifieurs ne se prononce pas favorablement. Ceci présente l'avantage de «verrouiller» un certain nombre d'hypothèses de classification, permettant ainsi de guider plus efficacement l'alignement des observations lors de l'analyse syntaxique.

Nous conserverons par la suite la fusion de type produit, celle-ci donnant les meilleurs résultats en détection.

On constate de plus 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 8 que le taux de détection progresse très significativement entre la première et la deuxième proposition, et qu'il atteint plus de 90% en considérant les cinq premières. Sur la base de test, les résultats montrent que 90% des composantes connexes du document peuvent être rejetées. Ces résultats prometteurs montrent le potentiel de l'approche à fournir les bonnes séquences de composantes tout en rejetant la majorité du document.

9.2 Rappel-précision du système

Nous présentons dans cette section les résultats obtenus en fin de chaîne de traitement, à l'issue de la reconnaissance des champs. Afin de ne pas multiplier les résultats, nous ne détaillons pas les résultats suivant le type de champ. 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 de tout type, détectés et reconnus.

Nous évaluons notre système par la courbe rappel-précision, couramment utilisée pour évaluer les performances d'un système d'extraction d'information [1]. Le rappel et la précision sont définis de la manière suivante :

rappel = nombre de champs bien reconnus / nombre de champs à reconnaître

précision = nombre de champs bien reconnus / nombre de champs proposés par le système

Le rappel mesure la capacité du système à localiser et reconnaître tous les champs d'un document. La précision traduit la capacité du système à ne fournir que des bonnes propositions et donc à limiter la fausse alarme.

La figure 25 donne la courbe rappel/précision du système.

En conservant les N meilleurs choix de l'analyseur, le rappel augmente et la précision diminue. Nous detectons 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.

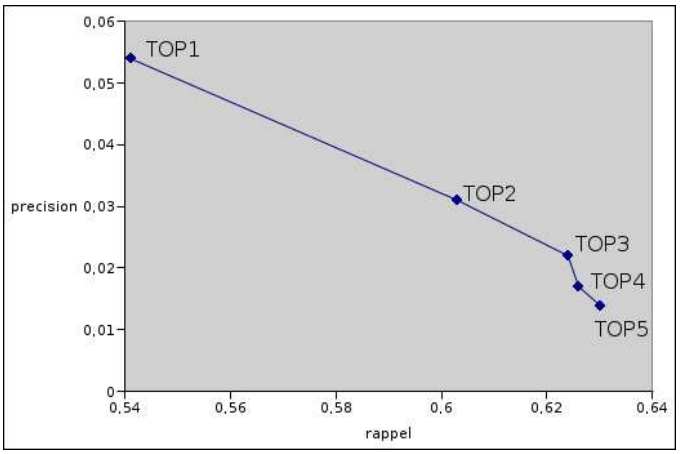

FIG. 25 – Courbe rappel/précision du système. Plus on conserve les propositions d'un rang élevé, plus on augmente le rappel, mais plus on diminue la précision.

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.

On constate que le système est capable de reconnaître de 54 à 63% des codes postaux, codes clients et numéros de téléphone des documents. La précision du système est en revanche relativement faible. Nous proposons donc un module de vérification permettant d'accepter ou de rejeter les séquences de composantes reconnues.

9.3 Limitation de la fausse alarme

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. Nous proposons toutefois une méthode de rejet des champs basée sur l'interprétation d'un certain nombre de scores provenant des différentes

étapes de traitement. Ces scores constituent un vecteur de caractéristiques soumis à un classifieur de type MLP qui se prononce sur l'acceptation ou le rejet de l'hypothèse de champs.

Le vecteur comporte 14 caractéristiques réparties en trois familles :

Caractéristiques issues de la localisation : lors de l'analyse syntaxique d'une ligne de texte, l'algorithme de Viterbi fournit un score d'alignement des composantes sur les modèles qui traduit la fiabilité de la localisation des champs. Le score de l'alignement et les écarts avec les scores des autres alignements sont retenus comme caractéristiques.

Caractéristiques provenant de la reconnaissance : partant de l'hypothèse selon laquelle une séquence de composantes non numériques produit des confiances faibles lors de l'étape de reconnaissance (voir figure 21), nous avons choisi d'intégrer dans le vecteur les trois caractéristiques suivantes : les moyennes géométrique et arithmétique des scores associés aux premières propositions du classifieur chiffre pour toutes les composantes du champ, ainsi que le minimum des scores de confiance.

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 champ numérique présentent généralement des régularités que ne possèdent pas les fausses alarmes. Nous avons donc ajouté les caractéristiques suivantes :

- L'écart type des ordonnées minimum et maximum des chiffres
- L'écart type des hauteurs et largeurs des chiffres
- L'écart type des distances entre les abscisses des centres de gravité des chiffres

Le MLP est entrainé sur une base de 17000 champs numériques (16800 fausses alarmes et 200 véritables champs). L'unique sortie du classifieur fournit une valeur entre 0 (rejet du champ) et 1 (acceptation du champ). La figure 26 montre l'évolution du rappel et de la précision avant et après l'étape de vérification, suivant le rang considéré.

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

9.4 Temps de traitement

Le temps de traitement est également un critère important pour évaluer le 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. Il faut environ 15 minutes pour extraire

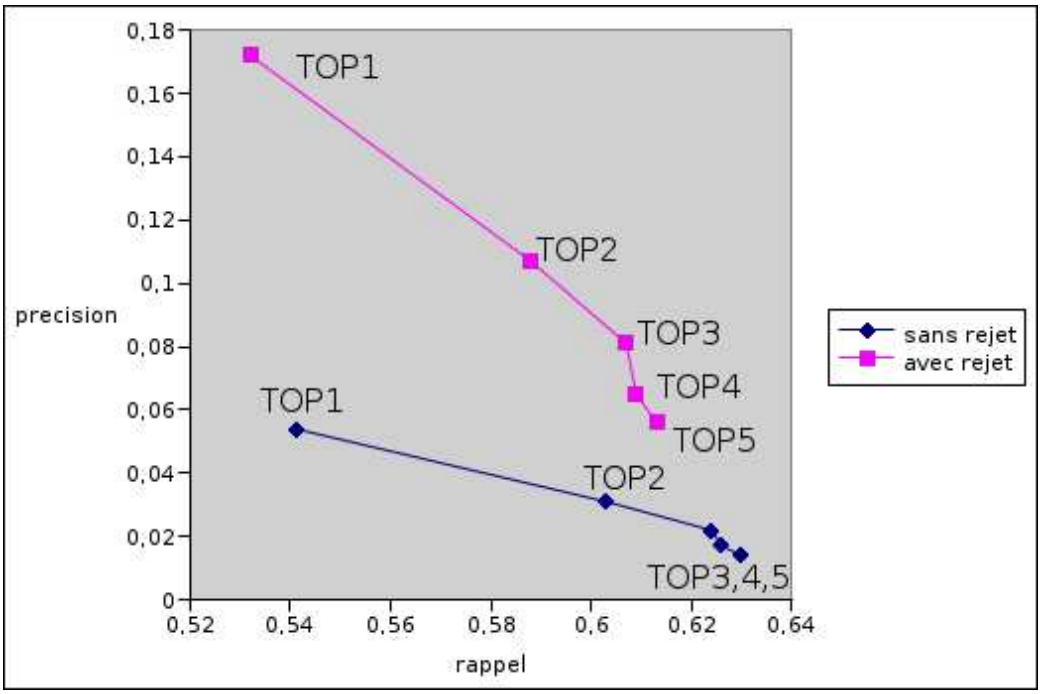

FIG. 26 – Courbe rappel/précision du système avant et après rejet des champs par le module de vérification.

et reconnaître les trois types de champ des 293 images de notre base de test sur une machine cadencée à 1,5GHz, soit environ 3 secondes par image. Ces temps comprennent l'intégralité des traitements, depuis l'extraction des composantes connexes jusqu'à la reconnaissance des champs. Notons toutefois que les traitements n'ont pas fait l'objet des différentes optimisations possibles en vue d'une industrialisation.

10 Conclusion et perspectives

Dans le cadre du traitement automatique de courriers entrants manuscrits, nous avons présenté une méthode d'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. 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 qu'une majorité de champs peut être détectée, et que 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.

La méthode d'extraction utilisée permet d'effectuer une reconnaissance des champs largement contrainte puisque les chiffres liés sont identifiés. La méthode de reconnaissance fournit la bonne hypothèse de reconnaissance dans 80% des cas sur les champs bien détectés. Au total, plus de 60% des champs présents dans un document sont correctement extraits et reconnus par le système. La précision du système est cependant assez faible puisqu'un nombre important de fausses alarmes est généré. Une méthode de vérification des hypothèses de champs a donc été mise en place et permet de multiplier par trois la précision du système.

Notons que l'intégration d'un tel système en milieu industriel pourra bénéficier d'un certain nombre de connaissances *a priori* spécifiques aux types de champs recherchés (connaissances que nous n'avons pas intégrées ici dans le processus de localisation) afin d'en améliorer les performances et en particulier la précision. Par exemple, un numéro de téléphone commence toujours par un '0' ; les codes postaux se trouvent généralement dans la partie supérieure du document ; etc. Un module de mise en concurence des champs pourra également être developpé pour éviter les fausses alarmes dues à l'inclusion d'un champ dans un autre (voir figures 14 et 15).

Afin de fiabiliser notre système, la mise en parallèle de notre approche avec des stratégies alternatives pourra être effectuée. En particulier, il serait intéressant d'appliquer des modèles de lignes intégrant les valeurs numériques des chiffres, afin de pouvoir prendre en compte les contraintes mentionnées précédemment (numéro de téléphone comencant par "06", etc.) dès la phase de localisation. Cette méthode impose la localisation de tous les chiffres dans le document : chiffres isolés et chiffres liés. Contrairement à la méthode présentée dans cet article, une phase de segmentation des composantes est donc nécessaire. La clé du problème réside dans le contournement des stratégies classiques de sur-segmentation systématique des composantes qui entrainent une combinatoire très importante et donc des temps de traitement conséquents. Nous travaillons actuellement sur ce sujet.

Références

[1] Salton, G., "Automatic information organization and retrieval", *Mac Graw Hill Book Co., NY*, 1968.

[2] Forney, G.D., "The Viterbi algorithm", *Proc. IEEE*, vol. 61, 1973, pp. 268–278.

[3] LeCun, Y., B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition", *Neural Computation*, vol. 1, no. 4, 1989, pp. 541–551.

[4] Rabiner, L. R. "A tutorial on hidden markov models and selected apllications in speech recognition". In *Readings in Speech Recognition.* Kaufmann, 1990, pp. 267–296.

[5] Richard, M.D. and R.P. Lippmann, "Neural network classifiers estimate bayesian a posteriori probabilities", *Neural Computation*, vol. 3, 1991, pp. 461–483.

[6] Xu, L., A. Kryzak, C.Y. Suen and K. Liu, "Method of combining multiple classifiers and their applications to handwritting recognition", *IEEE Trans. on SMC*, vol. 22, no. 3, 1992, pp. 418–435.

[7] Kimura, F., S. Tsuruoka, Y. Miyake and M. Shridhar, "A lexicon directed algorithm for recognition of unconstrained handwritten words", *IEICE Trans. on Information & Syst.*, vol. E77-D, no. 7, 1994, pp. 785–793.

[8] Bishop, C.M., *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[9] Congedo, G., G. Dimauro, S. Impedovo and G. Pirlo, "Segmentation of numeric strings", *ICDAR'95*, vol. 2, 1995, pp. 1038–1041.

[10] Likforman-Sulem, L. and C. Faure, "Une methode de resolution des conflits d'alignements pour la segmentation des documents manuscrits", *Traitement du Signal*, vol. 12, 1995, pp. 541–549.

[11] Casey, R. and E. Lecolinet, "A survey of methods and strategies in character segmentation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 7, 1996, pp. 690–706.

[12] Trier, O.D., A.K. Jain and T. Taxt, "Feature extraction methods for character recognition : A survey", *Pattern Recognition*, vol. 29, no. 4, 1996, pp. 641–662.

[13] Dzuba, G., A. Filatov, D. Gershuny, I. Kil and V. Nikitin. *Check amount recognition based on the cross validation of courtesy and le-*

gal amount fields. Automatic Bank Check Processing. World Scientific, 1997, pp. 177–194.

[14] Gorski, N., "Optimizing error-reject trade off in recognition systems", *ICDAR'97*, vol. 2, 1997, pp. 1092–1096.

[15] Kim, G. and V. Govindaraju, "A lexicon driven approach to handwritten word recognition for real-time applications", *IEEE Trans. on PAMI*, vol. 19, no. 4, 1997, pp. 366–378.

[16] Heutte, L., T. Paquet, J.V. Moreau, Y. Lecourtier and C. Olivier, "A structural/statistical feature based vector for handwritten character recognition", *Pattern Recognition Letters*, vol. 19, 1998, pp. 629–641.

[17] Kim, G. and V. Govindaraju, "Handwritten phrase recognition as applied to street name images", *Pattern Recognition*, vol. 31, no. 1, 1998, pp. 41–51.

[18] Dey, S. "Adding feedback to improve segmentation and recognition of handwritten numerals". Master's thesis, Massachusetts Institute of Technology, 1999.

[19] Lorette, G., "Handwritting recognition or reading ? what is the situation at the dawn of the 3rd millenium ?", *IJDAR*, vol. 2, no. 1, 1999, pp. 2–12.

[20] Jain, A.K., R.P.W. Duin and J. Mao, "Statistical pattern recognition : A review", *IEEE Trans. on PAMI*, vol. 22, no. 1, 2000, pp. 4–37.

[21] Plamondon, R. and S.N. Srihari, "On-line and off-line handwritting recognition : A comprehensive survey", *IEEE Trans. on PAMI*, vol. 22, no. 1, 2000, pp. 63–84.

[22] El-Yacoubi, A., M. Gilloux and J.-M. Bertille, "A statistical approach for phrase location and recognition within a text line : An application to street name recognition", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, 2002, pp. 172–188.

[23] Morita, M., R. Sabourin, F. Bortolozzi and C.Y. Suen, "Segmentation and recognition of handwritten dates", *IWFHR*, 2002, pp. 105–110.

[24] Zouari, H., L. Heutte, Y. Lecourtier and A. Alimi, "Un panorama des méthodes de combinaison de classifieurs en reconnaissance de formes", *RFIA'2002*, vol. 2, 2002, pp. 499–508.

[25] Pal, U., A. Belaïd and C. Choisy, "Touching numeral segmentation using water reservoir concept", *Pattern Recognition Letters*, vol. 24, 2003, pp. 261–272.

[26] Pitrelli, J.F. and M.P. Perrone, "Confidence-scoring post-processing for off-line handwritten-character recognition verification", *ICDAR'03*, vol. 1, 2003, pp. 278–282.

[27] Prevost, L., C. Michel-Sendis, A. Moises, L. Oudot and M. Milgram, "Combining model-based and discriminative classifiers : application to handwritten character recognition", *ICDAR'03*, vol. 1, 2003, pp. 31–35.

[28] Rahman, A.F.R. and M.C. Fairhurst, "Multiple classifier decision combination strategies for character recognition : A review", *IJDAR*, vol. 5, 2003, pp. 166–194.

[29] Vinciarelli, A., S. Bengio and H. Bunke, "Offline recognition of unconstrained handwritten texts using hmms and statistical language models", *IEEE Trans. on PAMI*, vol. 26, no. 6, 2004, pp. 709–720.