Combining neural-based approaches and linguistic knowledge for text recognition in multimedia documents

Khaoula Elagouni

Thèse

THESE INSA Rennes
sous le sceau de l'Université européenne de Bretagne
pour obtenir le titre de
DOCTEUR DE L'INSA DE RENNES
Spécialité : Informatique

présentée par
Khaoula Elagouni
ECOLE DOCTORALE : *MATISSE*
LABORATOIRE : *IRISA*

Combining neural-based approaches and linguistic knowledge for text recognition in multimedia documents

Thèse soutenue le 28.05.2013
devant le jury composé de :

Christian Viard-Gaudin
Prof. d'Université, université de Nantes, Irccyn / Président du jury
Olivier Lézoray
Prof. d'Université, université de Caen Basse-Normandie / rapporteur
Bernard Mérialdo
Professeur d'Université, Eurecom / rapporteur
Emmanuel Morin
Professeur d'Université, université de Nantes, Lina / examinateur
Pascale Sébillot
Prof. d'Université, INSA de Rennes, Irisa / directrice de thèse
Christophe Garcia
Professeur d'Université, INSA de Lyon, Liris / co-encadrant
Franck Mamalet
Chercheur, Orange labs / co-encadrant industriel

Combining neural-based approaches and linguistic knowledge for text recognition in multimedia documents

Khaoula Elagouni

En partenariat avec

To Moez, to Elyes,
To my parents, family, friends and all those who supported me during these years...

Acknowledgement

This thesis owes its existence to the help and the support of many people.

First, I would like to express my deep gratitude to my advisors Franck Mamalet, Christophe Garcia and Pascale Sébillot for all their valuable guidance in these three last years. I enjoyed working with them and learning from them. I am particularly thankful for their large contribution during the publication stage of this work, in order to create well-structured and readable texts.

I also want to thank the members of my jury; in particular Olivier Lézoray and Bernard Mérialdo, for the time they spent reviewing my thesis manuscript, and for the valuable feedback and suggestions they provided me with. My thanks also go to Christian Viard-Gaudin for honoring me by presiding my jury, and to Emmanuel Morin for his interest to my work and his relevant questions during my PhD defense.

I would like to thank the Orange Labs company, in particular Alexandre Nolle (head of the ACTS unit) and Sid-Ahmed Berrani (head of the MAS team), for giving me the opportunity to explore an interesting industrial topic. I also want to thank all the members and PhD students of the MAS team, in particular Moez, Alina, Ali and Haykel, for their cooperative spirit and the excellent working atmosphere. I really spent three very pleasant years.

I also would like to express my gratitude to everybody at TexMex research team of the IRISA Laboratory, and particularly Patrick Gros (head of TexMex) for their advices and help during these years.

Finally, I want to say thank you to my parents for their continuing support in every respect and to my Moez, and my sweet angel Elyes for their constant encouragement and love.

Contents

List of Figures

List of Tables

French summary

0.1 Introduction

Avec le développement de nouveaux systèmes d'acquisition d'images et l'avènement de nombreux services de partage de vidéos, l'indexation automatique de documents multimédias est devenue cruciale pour gérer ces vastes collections. L'enjeu majeur consiste à extraire l'information pertinente permettant de résumer les contenus et de retrouver les documents.

Durant ces dernières années, de nombreux travaux se sont focalisés sur la problématique de l'indexation d'images et de vidéos fondée sur l'analyse automatique des contenus multimédias. Certains proposent de décrire le contenu au moyen d'images-clés [CZKA02], en se basant sur la classification d'évènements [SW05], en optant pour la détection d'objets considérés de haut niveau sémantique [MBPLM08], voire en intégrant la transcription de la parole prononcée [CZKA02] (dans le cas de la vidéo). D'autres optent pour la prise en compte des textes présents dans les documents multimédia comme nouveau moyen d'accès à la sémantique des contenus [LS96]. C'est dans ce contexte que sŠinscrivent nos travaux de thèse qui se focalisent sur la problématique de la reconnaissance automatique de textes dans les documents multimédia. Cet intérêt est justifié par le fait que ces textes—qui peuvent correspondre à des titres de reportages, à des noms de personnes ou de villes, *etc.*—représentent des indices sémantiques forts et fournissent des éléments importants pour de nombreuses applications telles que l'indexation et la recherche des images et des vidéos, l'archivage et le chapitrage du flux TV, les bibliothèques numériques, la vision robotique, etc. Cette extraction d'indices textuels nécessite cependant des systèmes robustes à la variabilité de styles et de tailles des caractères, à la faible résolution, à la complexité du fond, aux conditions d'acquisition difficiles, etc.

Dans cette thèse, nous proposons des systèmes complets d'OCR (*Optical Character Recognition*) spécifiquement adaptés aux images et aux vidéos et qui s'appliquent aussi bien aux textes incrustés (ajoutés artificiellement dans les images ou les vidéos) qu'aux textes de scène (acquis n'importe où; sur des affiches, des murs ou des panneaux et qui peuvent avoir des fontes extrêmement variables et être pris dans des conditions assez complexes). Deux types d'approches, en utilisant et en évitant l'étape de la segmentation en caractères, sont conçus et étudiés tout en mettant en évidence les

avantages et les limites de cette étape.

Cette thèse est organisée comme suit. Après avoir présenté notre première approche fondée sur une segmentation adaptée à la morphologie locale des images de textes dans la section 0.2, deux autres méthodes qui se passent de cette étape sont décrites dans les sections 0.3 et 0.4. Alors que la première approche est fondée sur un processus de "scanning" multi-échelle et un modèle de graphe, la deuxième s'appuie sur une nouvelle représentation des images de textes et sur un modèle de classification connexionniste. Nos trois systèmes d'OCR sont ensuite testés et évalués sur deux bases de textes; en l'occurrence une base de textes incrustés et une base de textes de scène. Les résultats de ces expérimentations sont fournis et discutés dans la section 0.5. Enfin, la section 0.6 conclut cette étude et met en évidence nos travaux futurs.

0.2 L'approche fondée sur la segmentation

Afin de reconnaitre les textes présents dans les documents multimédia, nous proposons une première approche qui consiste à segmenter les images de textes pour obtenir des régions contenant des caractères individuels avant d'entamer leur reconnaissance. Contrairement aux méthodes de l'état de l'art, notre approche définit des segmentations non-linéaires fiables et précises. Outre la robustesse de la méthode de reconnaissance de caractères fondée sur une approche de classification neuronale, notre seconde contribution principale réside dans l'introduction d'un mode de supervision reposant sur un modèle de langue.

Vu que la segmentation de l'image de textes en caractères est un point crucial pour la reconnaissance (toute erreur réduit en effet directement les performances de l'OCR), nous nous sommes donc intéressés à mettre en œuvre une méthode segmentation fiable qui permet de séparer les caractères tout en s'adaptant à la morphologie locale de l'image. Pour ce faire, nous commençons par l'analyse statistique d'intensités des images de textes (qu'on combine à intégration multi-temporelle, dans le cas de la vidée) dans le but de discriminer entre la classe "texte" et la classe "fond". Une carte floue de degrés d'appartenance à la classe "texte" est ainsi générée. En utilisant cette carte, nous déterminons ensuite les séparations entre les caractères comme des chemins traversant le fond et qui coupent l'image du texte verticalement. Ces chemins sont calculés avec un algorithme du plus court chemin spécifiquement adapté à notre application. L'intérêt de cette approche de segmentation réside dans sa capacité à fournir des séparations précises et adaptées à la morphologie des caractères. Par ailleurs, nous distinguons entre deux types de segmentations : des segmentations dites "fiables" et dont on est sûr et des segmentations dites "à risque" qui sont douteuses et qui seront remises en question par la suite. Comme le montre la figure 1, ces segmentations "à risque" peuvent correspondre à des sur-segmentations (telles que le cas du "r") ou à des segmentations de caractères attachés à un fond complexe (telles que le cas du "rt").

Figure 1: Un exemple de segmentations obtenues : les chemins verts correspondent aux segmentations "fiables"alors que les rouges correspondent aux segmentations "à risque".

Une fois les caractères sont segmentés, nous pouvons donc adresser leur reconnaissance. Contrairement à la majorité des méthodes de l'état de l'art, dans le cadre de notre travail nous proposons de mettre au point un modèle de classification générique robuste à la grande variabilité de style, de taille et de couleur des caractères. Les méthodes neuronales semblent convenir parfaitement à ces attentes. Nous nous appuyons donc sur un réseau de neurones à convolutions [LB95] capable d'apprendre automatiquement en même temps à extraire les caractéristiques appropriées et à reconnaître les classes de caractères, sans aucune phase de prétraitement ou de binarisation. Pour notre tâche de reconnaissance de caractères, nous avons testé plusieurs configurations de réseau avant d'opter pour une architecture composée de cinq couches cachées et d'une couche finale de sortie. Le réseau prend en entrée une image de caractère couleur redimensionnée à la taille de $S \times S$ pixels et produit en sortie un vecteur de taille N (le nombre de classes considérées). Les valeurs de ce vecteur de sortie peuvent être interprétées comme des scores traduisant l'appartenance de l'image d'entrée à chaque classe. Le caractère reconnu correspond ainsi à la classe obtenant la valeur de sortie la plus élevée.

Malgré les bonnes performances de la reconnaissance neuronale, des erreurs peuvent être produites à cause d'une confusion de caractères visuellement similaires, de la complexité du fond et de la mauvaise qualité des images. Pour pallier les ambiguïtés relatives à cette reconnaissance locale caractère par caractère, nous proposons d'introduire des connaissances linguistiques qui vont piloter les étapes de l'OCR. Dans ce contexte, les modèles de langue de type n-grammes (qui sont déjà très utilisés dans le domaine de la transcription de la parole mais aussi de la traduction automatique,) ont montré leur capacité à améliorer les performances de reconnaissance en prenant en compte le contexte lexical des mots. Pour notre problème de reconnaissance de texets, nous proposons d'utiliser ce type de connaissance linguistique et d'intégrer un modèle n-grammes appris sur un corpus de mots afin d'estimer la probabilité qu'une séquence de lettres soit observée dans une langue donnée. Les probabilités estimées sont ensuite introduites dans notre système de reconnaissance pour gérer les hypothèses de segmentations (notamment les segmentations "à risque") et réduire les erreurs liés à la confusion de caractères.

L'approche proposée est évaluée sur des textes incrustés et des textes de scène. Les résultats obtenus sont présentés dans la section 0.5.

0.3 L'approche sans segmentation

Bien que notre premier système d'OCR repose sur une méthode segmentation précise calculant des séparations non-linéaires entre les caractères, les performances de ce système restent faibles dans le cas des images présentant d'importantes distorsions. Ce fait peut être expliqué par l'étape de la segmentation qui, dans le cas de ces images, produit de nombreuses sous- et sur-segmentations, conduisant à des erreurs de reconnaissance.

Pour remédier à cette limite, nous proposons une deuxième approche qui se passe de l'étape de la segmentation en intégrant un processus de scanning multi-échelle permettant de reconnaitre les caractères à leur propre position et échelle directement à partir de l'image du texte. L'idée est d'utiliser quatre fenêtres glissantes de tailles différentes (proportionnelles à la hauteur de l'image) qui sont déplacés à travers l'image dans le but d'avoir au moins une fenêtre qui sera bien alignée avec chaque caractère. La figure 2 illustre ce processus. Pour couvrir tous les caractères et bien les cadrer, nous proposons aussi d'adapter les bords verticaux des fenêtres à la morphologie locale des images. Ainsi, des bords non-linéaires (calculés de manière similaire à celle des segmentations non-linéaires de l'approche fondée sur la segmentation présentée en section 0.2) sont attribués à chaque fenêtre glissante dans l'image.

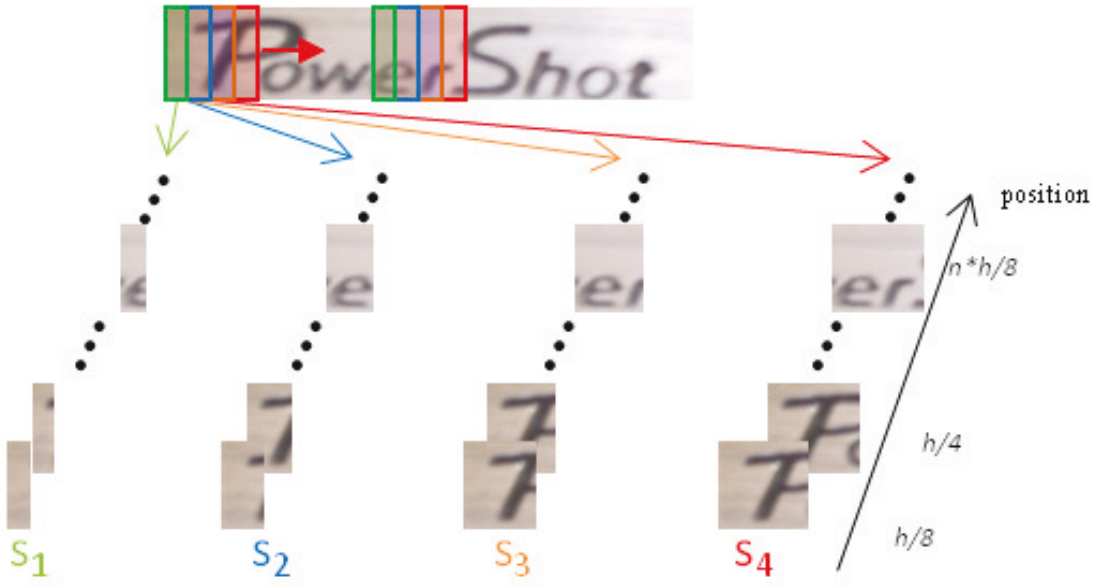

Figure 2: Le processus de "scanning" multi-échelle de l'image du texte: h désigne l'hauteur de l'image.

Comme le montre la figure 2, l'application de du processus de scanning multi-échelle aux images de textes génère de nombreuses fenêtres qui peuvent correspondre à des caractères individuels bien centrés ou à des images de caractères non-valides

(tels que des parties de caractères, des caractères mal-alignés ou des espaces inter-caractères). Une étape de classification est ainsi nécessaire pour identifier le contenu de ces fenêtres. Dans ce contexte, nous proposons une classification hiérarchique qui débute par une phase de tri pour distinguer les fenêtres contenant des caractères "valides" du reste, et qui ensuite analyse ces fenêtres sélectionnés afin de reconnaitre leurs caractères. Pour ceci, nous nous basons sur la combinaison de deux réseaux de neurones à convolutions (ayant des architectures similaires à celui utilisé pour la reconnaissance des caractères dans l'approche fondée sur la segmentation présentée en section 0.2) dont les rôles respectifs sont le tri de fenêtres et la reconnaissance de caractères.

Après avoir scanné les images de textes et classés les fenêtres glissantes, la prochaine étape est d'analyser ces résultats de classification pour reconnaitre les textes présents dans les images. Pour cette tâche, nous optons pour un modèle de graphe permettant de représenter les contraintes spatiales entre les différentes fenêtres. Pour la construction de ce graphe, nous représentons les bords de toutes les fenêtres résultantes par des nœuds connectés par des arcs chacun représentant une fenêtre. En attribuant les résultats de la classification aux arcs, toutes les combinaisons possibles de fenêtres sont testées et évalués afin de reconnaitre le texte présent dans l'image. Un algorithme de Viterbi est ainsi appliqué dans le graphe pour déterminer le chemin le plus probable (évitant les arcs correspondant à des fenêtres contenant de caractères non-valides) et donc obtenir la séquence de caractères (*i.e.*, le texte) reconnue.

Dans cette approche, nous proposons aussi d'introduire certaines connaissances linguistiques pour prendre en considération le contexte lexical des mots et lever quelques ambiguïtés de la classification (notamment les cas de certaines fenêtres mal-alignées avec un caractère qui sont confondues avec d'autres classes de caractères telle que une partie d'un "W" confondue avec un "V" ou un "N"). Pour cette approche, nous nous reposons aussi sur un modèle de langue n-grammes caractères et nous intégrons les probabilités estimés par ce dernier dans notre graphe. Ces probabilités permettent ainsi de piloter le processus de la reconnaissance tout en pondérant les transitions entre les arcs du graphe. Le texte reconnu est donc obtenu en prenant en compte deux informations complémentaires; à savoir les résultats de la classification et le contexte lexical.

Cette approche est évaluée aussi sur des textes incrustés et des textes de scène et ses performances de reconnaissance sont comparées à celles de l'approche fondée sur la segmentation. La section 0.5 rapporte les principaux résultats obtenus et souligne l'intérêt et les limites de se passer de la phase de segmentation.

0.4 L'approche récurrente connexionniste

En intégrant un processus de "scanning" multi-échelle et en utilisant un modèle de graphe, notre seconde approche prouve qu'il est possible d'éviter la phase cruciale de la segmentation qui peut réduire les performances de reconnaissance, en particulier dans

le cas des textes de scène. Néanmoins, la principale faiblesse de ce système demeure la complexité du graphe qui nécessite de tester un grand nombre de combinaisons de fenêtres avant d'obtenir le texte reconnu.

Notre troisième approche propose ainsi un nouveau moyen pour se passer de la segmentation et éviter la complexité du modèle de graphe. Cette méthode opère en deux phases : tout d'abord en générant une représentation originale des images de textes fondée sur des séquences de caractéristiques apprises, ensuite en utilisant un modèle connexionniste récurrent spécifique capable de classer ces caractéristiques prenant en compte leur dépendance temporelle.

Dans cette approche, les images de textes sont tout d'abord scannées à différentes échelles en utilisant le même processus de "scanning" défini pour l'approche précédente (voir section 0.3). Les fenêtres résultantes sont ensuite employées pour produire une représentation pertinente des images de textes. Contrairement à la majorité des méthodes de l'état de l'art qui optent pour des représentations fondées sur des caractéristiques conçues manuellement, nous proposons de représenter chacune de ces fenêtres par un ensemble de caractéristiques apprises par un modèle neuronale—en l'occurrence un réseau de neurones à convolutions. L'idée consiste à entrainer un réseau de neurones à reconnaitre des images de caractères et une fois l'apprentissage est terminé, utiliser l'avant-dernière couche du réseau comme un extracteur de caractéristiques. Typiquement, dans notre travail chaque fenêtre glissante est présentée au réseau appris pour récupérer les activations de la dernière couche générant ainsi un vecteur de caractéristiques. L'ensemble des vecteurs obtenus, pour une image de texte donnée, est ensuite rassemblé en une séquence de vecteurs constituant une représentation multi-échelle de l'image.

Dans le but de reconnaitre les textes présents dans les images, la prochaine étape est d'analyser les représentations obtenues et classer les vecteurs des caractéristiques apprises. Pour ce faire, nous choisissons d'utiliser un réseau de neurones récurrents particulier, le *bidirectional long-short term memory* (BLSTM), ayant des connexions récurrentes lui conférant une mémoire interne et le rendant ainsi capable de résoudre des problèmes de classification de séquences de données. Outre sa capacité à gérer la dépendance entre les éléments successifs d'une séquence, le BLSTM offre l'avantage de prendre en compte aussi bien le contexte passé que le contexte futur lors de la classification. Néanmoins, ce modèle nécessite une opération de segmentation pour préciser la position exacte de chaque caractère dans la séquence de caractéristiques d'entrée. Vu que notre approche vise à se passer de toute segmentation, nous intégrons, dans notre réseau BLSTM, une couche spécifique, appelée classification temporelle connexionniste (CTC), introduite par Graves *et al.* [GLF+09] et permettant d'étendre l'application d'un réseau de neurones récurrents au cas des données non segmentées. En effet, la CTC permet de créer le lien entre la séquence de sortie d'un BLSTM et la séquence cible de caractères en introduisant une classe supplémentaire, appelée BLANK, qui sera activée entre deux caractères. Une fois l'apprentissage effectué, il est nécessaire d'interpréter les sorties du BLSTM pour en déduire la séquence des

caractères reconnues. Cette étape porte le nom de décodage. La figure 3 montre un exemple de texte reconnu en décodant les séquences de sortie du réseau.

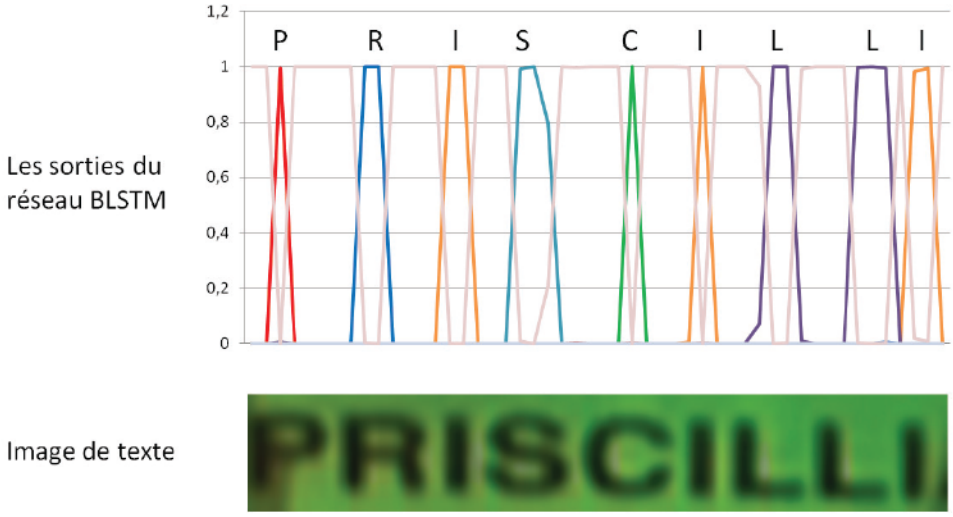

Figure 3: Illutration des sorties du BLSTM appris en utilisant la CTC : chaque courbe d'une couleur représente le niveau d'activation d'une classe donnée et la courbe rose claire correspond à la classe supplémentaire "BLANK".

0.5 Résultats expérimentaux

Cette section présente nos principales expérimentations et leurs résultats. Elle débute par une description brève des bases utilisées dans cette étude, en l'occurrence une base de textes incrustés dans des vidéos réelles de journaux télévisées français et une base publique de textes de scène (la base ICDAR 2003). En suite, nous présentons l'évaluation de nos trois approches de reconnaissance de textes réalisée sur ces deux bases et nous discutons les principaux résultats.

0.5.1 Données expérimentales

Nos expérimentations sont effectuées sur deux types de textes représentés dans deux bases :

- Une base de textes incrustés dans des vidéos de journaux télévisés français : cette base comporte des textes assez variables en tailles (de 8 à 24 pixels de hauteur), couleurs, styles et fonds (fonds uniformes ou plus complexes). Notons que, pour cette base, avant d'appliquer nos systèmes d'OCR, nous avons mis en place une chaine automatique de traitement qui permet de détecter et d'extraire les textes incrustés dans les vidéos.

- Une base publique de textes de scène, ICDAR 2003 : cette base comporte des images de textes saisies n'importe ou dans l'environnement dans des conditions assez difficiles (faibles résolutions, illuminations non uniformes, en présence d'ombre et de reflets, etc.). Les textes de cette base sont extrêmement variables et peuvent être imprimés, écrits ou même dessinés.

Les figures 4 et 5 montrent quelques exemples d'images de textes extraites de ces deux bases.

Figure 4: Exemples de textes incrustés dans les vidéos de journaux télévisés.

Figure 5: Exemples de textes de scène de la base ICDAR 2003.

0.5.2 Performance des approches proposées

Nos trois systèmes d'OCR ont été testés et évalués sur les deux bases présentées ci-dessus.

En utilisant la chaine conçue de détection et suivi de textes dans les vidéos, de nombreuses expérimentations ont été effectuées pour évaluer nos approches et comparer leurs performances à celle des méthodes existantes et de certains moteurs d'OCR commerciaux, en l'occurrence ABBYYFineReader OCR et Tesseract OCR.

Les résultats obtenus ont montrés que toutes nos approches obtiennent des bonnes performances sur cette base avec des scores de taux de reconnaissance caractères supérieurs à 90%. En particulier, les scores (95% et 88% de taux de reconnaissance caractère et taux de reconnaissance mot) accomplis par la méthode fondée sur la segmentation prouvent que lorsque l'étape de segmentation fonctionne bien, elle permet une reconnaissance précise et ainsi une amélioration des résultats. L'approche connexionniste a également réalisé une excellente performance avec un taux de reconnaissance caractère de 97.35% et un taux de reconnaissance mot de $87,20\%$ (ce taux inférieur à celui atteint par l'OCR fondé sur la segmentation peut être expliqué par le manque des certains espaces entre les mots, ce qui conduit à des erreurs de deux mots consécutifs). Ces résultats mettent en évidence la contribution des représentations basées les caractéristiques apprises et démontre la grande capacité de la méthode proposée à gérer les dépendances entre les vecteurs de caractéristiques tout en évitant l'étape de la segmentation.

En ce qui concerne la comparaison avec les systèmes d'OCR commerciaux, toutes nos méthodes ont atteint des résultats bien meilleurs que ceux du moteur Tesseract OCR (avec plus de +10% de taux de reconnaissance mot). Notre approche fondée sur la segmentation et celle connexionniste réalisent des performances similaires à celle de ABBYY FineReader OCR avec une différence de respectivement $+0,13\%$ et $-0,50\%$ de mots correctement reconnus. Cependant, l'approche sans segmentation obtient un taux de reconnaissance plus faible que ABBYY FineReader OCR (avec -6% de taux de reconnaissance mot). Ce résultat peut être expliqué par la complexité du modèle de graphe qui produit des espaces manquants conduisant à des erreurs de reconnaissance de mots.

Les trois approches proposées ont été également évaluées sur la base de texte de scène. Les tests réalisés ont montré que la méthode fondée sur la segmentation permet d'obtenir des performances similaires à celles des méthodes state-of-the-art, tandis que l'OCR sans segmentation atteint des résultats qui dépasse les autres méthodes existantes (environ 47% de taux de reconnaissance mots correspondant à 70% de taux de reconnaissance caractères). Cela prouve l'apport de se passer de l'étape de segmentation et met en évidence ses limites ; toute erreur de segmentation diminue directement les performances de reconnaissance du système. En effet, dans le cas particulier des textes de scène, les importantes distorsions présentes dans les images rendent la segmentation particulièrement difficile menant à des erreurs de sur- et sous-segmentations qui réduisent considérablement les performances de reconnaissance.

Concernant l'approche connexionniste, les tests effectués ont montré que, bien que cette méthode est capable d'atteindre de bons scores sur l'ensemble d'apprentissage (98% de taux de reconnaissance caractère), néanmoins elle obtient de moins bonnes

performances sur l'ensemble de test (seulement 56% de taux de reconnaissance caractère). Nous pouvons expliquer ce fait par un problème sur-apprentissage, probablement parce que l'ensemble de données d'apprentissage ne comprend pas assez de variabilité dans le contexte passé et futur de chaque lettre. Ce résultat peut également être justifié par l'absence d'un modèle de langage dans cette approche.

Les performances de nos méthodes sur cette base de textes de scène ont été aussi comparées à celles de deux moteurs d'OCR commerciaux, ABBYYFineReader OCR et Tesseract OCR. Les tests réalisés ont montré que toutes nos méthodes obtiennent des résultats meilleurs que ceux des systèmes commerciaux. Cela prouve que, bien que les systèmes OCR commerciaux puissent réaliser des résultats satisfaisants sur les textes incrustés, nos systèmes se distinguent par leur grande capacité à gérer à la fois les textes incrustés et les textes de scène.

0.6 Conclusion

Dans cette thèse, nous avons élaboré trois approches complètes de reconnaissance de textes dans les documents multimédia. Outre leur capacité à traiter aussi bien les textes incrustés que les textes de scène, nos méthodes conçoivent de nouveaux moyens pour entamer les différentes étapes de la reconnaissance et proposent des solutions originales pour entamer de nombreuses difficultés rencontrés par les méthodes de la littérature.

Une première contribution de ce travail réside dans la définition de segmentations non linéaires entre les caractères qui s'adaptent à la morphologie locale des images. Ces segmentations permettent ainsi de séparer les caractères de manière précise favorisant une meilleure reconnaissance. Nous avons aussi prouvé qu'il est possible de se passer de l'étape de segmentation en intégrant un processus de "scanning" multi-échelle et un modèle de graphe. Dans ce travail, une représentation originale des images de textes fondée sur des caractéristiques apprises par un modèle neuronale a été proposée. Cette représentation a été utilisé pour alimenter un modèle de classification connexionniste particulier qui en combinant un réseau de neurones récurrent spécial et une classification connexionniste permet de gérer la dépendance des caractèristiques apprises et ainsi reconnaitre le texte sans aucune phase de segmentation. Par ailleurs, à travers nos expérimentations nous avons montré l'intérêt de l'intégration des modèles de langues dans nos approches et mis en évidence leur apport en terme de performance.

Comme perspectives d'extention de ce travail, nous pouvons distinguer des trois types de perspectives :

- Des perspectives directes qui constituent une continuité de nos travaux et peuvent améliorer les performances de nos approches: parmi ces approches, nous citons l'intégration d'un modèle de langue dans l'approche connexioniste qui permetra ainsi introduire le contexte lexical des mots et réduire les erreurs.

Une autre piste qui peut être creuser réside dans l'intégration des matrices de confusion (qui fournissent une information précise sur les erreurs de confusion du reconnaisseur de caractères) pour lever les ambiguités de reconnaissance.

- Des perspectives à long terme qui consistent à explorer des nouvelles techniques qui peuvent s'appliquer à certaines problématiques liées à la reconnaissance de textes. Dans ce contexte, nous pensons à la super-résolution [BJ08] qui peut améliorer la qualité de la segmentation, les techniques d'apprentissage non-supervisé, tel que les auto-encodeurs [RPCL07], qui peuvent être utiles pour générer des nouvelles représentation pertinentes des images de textes.

- Des perspectives d'ordre applicatif : Les résultats prometteurs obtenus dans le cadre de ce travail de thèse permettent ainsi dŠenvisager lŠintégration de nos OCR dans un système dŠindexation et de recherche de vidéos. Plus particulièrement, les textes reconnus serviront à extraire des informations de haut niveau sémantique permettant dŠindexer les vidéos.

Chapter 1

Introduction

The volume of audio-visual documents continue to grow tremendously especially with the advent of photo and video sharing, delinearized television programs and video on-demand professional systems. In this context, the access to the contents of multimedia documents and their understanding become an issue of great importance. Because the manual annotation of videos and images is extremely expensive and time consuming, performing an automatic extraction of useful information present in an image or a video is desirable and enables the design of powerful indexing, searching and browsing systems able to deal with voluminous multimedia databases.

During the last years, a lot of work has been dedicated to the problem of automatically indexing images and videos. Several attempts were proposed to describe digital video contents by means of objects identified as semantically important [MBPLM08], by sets of key frames [CSL02], by classifying shots and events [SW05], or also taking into account speech transcription [CZKA02]. Other approaches [LS96] made use of texts present in images and videos as another way to access to the semantics of these multimedia documents. So does our work which aims at providing accurate methods able to extract and recognize the textual clues embedded or captured in images and videos.

Actually, textual patterns present in multimedia documents provide high-level semantic clues (names and functions of persons, places and streets names, dates, titles of TV reports, names of products, etc.) often interesting for content-based image retrieval. Several applications and services can be developed using extracted and recognized texts. For instance, texts recognized in a TV stream provide keywords that can feed an indexing and retrieval system or help highlighting events (such as sport scores) and summarizing contents. For mobile phones, services such as a tourist translation assistant can be implemented relying on the recognized foreign texts captured with the mobile camera during a trip. In the case of digital libraries, recognized texts permit an easy management (including archiving, search, translation, etc.) of stored books, historical documents, magazines and journals and enable the access to their content. Recognized texts can also be used for many other applications such as

teaching videos and robotic vision systems.

All these applications require the help of a powerful Optical Character Recognition (OCR) system able to correctly recognize different kinds of texts embedded or captured in multimedia documents.

In this context, the design of efficient OCR systems specifically adapted to multimedia documents is an important issue and continues to be an active research field. In the community of text recognition, texts in multimedia documents have been classified into two main categories [JIKKJ04]:

- "caption" texts: which are texts overlaid artificially on images and videos (*cf.* (A) and (B) in Fig. 1.1). These texts can be created with clean background or superimposed with transparency on the image or on the video sequence;

- "scene" texts: which are texts existing naturally in the scene and captured with the acquisition system—namely a camera—in an image or in a video (*cf.* (C) and (D) in Fig. 1.1). Due to the under-controlled acquisition conditions, "scene" texts are often harder to recognize than "caption" texts.

In general, for both text categories, several difficulties make the task of text recognition a challenging problem that continues to interest many researchers, among which:

- the huge diversity of texts properties: texts embedded or captured in images and videos can vary a lot. First, in term of size, the height of characters can range from some pixels to some hundreds of pixels. Texts can also be of different colors or even multicolored. A great variability can be observed in term of fonts, a character can have extremely different shapes, can have outlines or not, and texts can have variable inter-character distance (*i.e.*, the distance between two successive characters) (*cf.* (A) in Fig. 1.2);

- the complex backgrounds: text images can be considered as a sequence of characters printed, superimposed, written, drawn or captured on a given background. Nevertheless, backgrounds can be so complex that the distinction between the characters and the background is extremely hard. For instance, texts can be superimposed on moving backgrounds in some video cases. Backgrounds and texts can also share a similar color so that even a human may fail to identify the text (*cf.* (B) in Fig. 1.2);

- the difficult acquisition conditions: the devices used to capture text images or videos and the conditions of the acquisition often influence the quality and the resolution of the text image to recognize. Several difficulties, such as non uniform lighting, occlusions, blurring effects, decoding artifacts, can be observed leading to a harder recognition task (*cf.* (C) in Fig. 1.2).

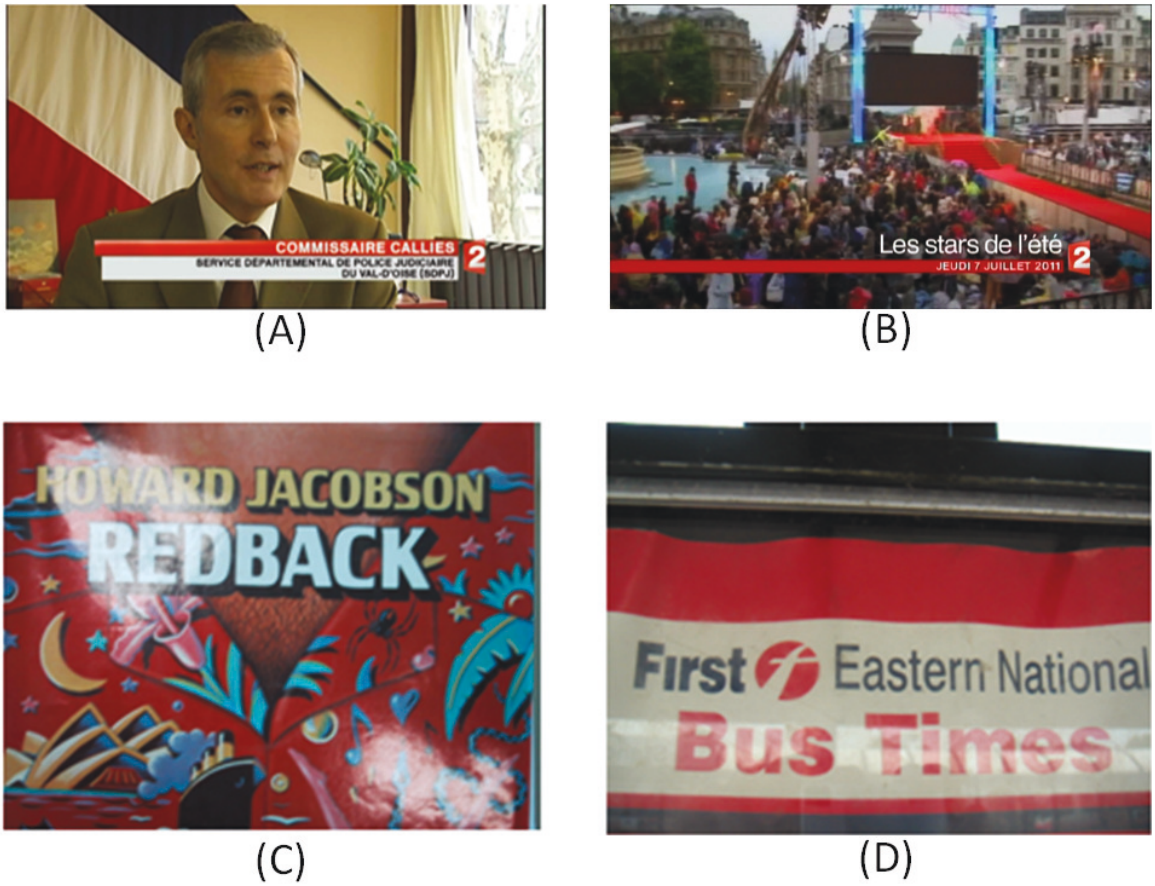

Figure 1.1: Examples of "caption" (A, B) and "scene" (C, D) texts.

In order to cope with these challenges, the design of efficient OCR systems specifically adapted to multimedia documents and able to deal with the different difficulties mentioned above is required. However, most existing systems are developed for only one kind of texts or address one single difficulty (such as the low resolution of text images or videos, complex backgrounds, etc.).

In this work, we focus on the recognition task for the two categories of texts—"caption" as well as "scene" texts. Our objectives are thus to design powerful OCR systems that are able to handle both "caption" and "scene" texts and, at the same time, to tackle the different challenges listed above. Through the proposed systems, we also aim at addressing the different steps involved in the text recognition task and at providing novel solutions to several issues encountered by the state-of-the-art methods.

The main contributions of our work are listed below.

First, an OCR system that relies on a character segmentation step is proposed. This OCR consists in segmenting the text image into individual characters, before recognizing them. This system highlights the difficulties related to the character

Figure 1.2: Examples of challenging texts: (A) texts of different fonts and sizes, (B) texts on complex backgrounds and (C) texts captured with hard conditions.

segmentation phase, and performs segmentations in a way that is different from the methods used in the literature: it proposes nonlinear segmentation well adapted to the local morphology of images, and hence enabling a better character recognition. The second contribution of this system lies in the robustness of the character recognition method based on a neural classification approach, where no preprocessing is necessary contrary to several other methods of the state-of-the-art. Due to this efficient neural-based character recognizer, our method is able to deal with extremely variable texts. Finally, we also incorporate a supervision scheme relying on a language model, in order to reduce character confusion and to improve recognition performance.

Two other segmentation-free OCR systems are proposed. They address the text recognition problem in a different way and avoid the step of character segmentation. In contrast to the main methodology in the literature, the absence of the crucial segmentation step allows to overcome several challenges related to complex backgrounds, difficult acquisition conditions and touching characters. The first main contribution of these systems lies in the incorporation of a multi-scale scanning process, using sliding windows, permitting the recognition of characters directly in the whole text image.

In the first segmentation-free OCR system, the recognition step is ensured by a robust window classifier based on a neural model. Another strength of this system consists in the construction of a graph model able to represent the sliding windows spatial constraints and to integrate some linguistic knowledge to drive the recognition scheme and increase the system's performance.

The second segmentation-free OCR system aims to learn the succession of characters directly from the sliding windows with a recurrent connectionist approach. While hand-crafted features are widely used by the community of text recognition to represent and thus classify texts, one of the main contribution of this OCR lies in the representation of text images by some sequences of learnt features. This is done by training a convolutional neural network to produce a relevant representation of sliding windows, robust to noise, geometrical transformations and deformations. Another strength of this approach is the design of a connectionist recurrent model specifically adapted to the learnt representations and whose task is to recognize texts taking into account the dependencies between successive learnt features.

In this work, many experimentations are carried out on two datasets: a "caption" text dataset extracted from real digital videos, and the public "scene" text dataset ICDAR 2003 [LPS+03]. Each proposed OCR system is tested and evaluated on both datasets. Results are detailed and discussed, highlighting the benefits and the limits of each proposed OCR systems and comparing their performance with those of the state-of-the-art methods.

This rest of this dissertation is organized as follows:

- Chapter 2 presents a literature survey on text recognition including all steps involved in this task, namely text image preprocessing, character segmentation, character recognition and text recognition. The interactions between these steps are also discussed.

- Chapter 3 describes the databases used in this work, namely a database of "caption" texts extracted from digital videos and the public database of natural "scene" texts ICDAR 2003. The main characteristics and difficulties of these databases are highlighted.

- Chapters 4, 5 and 6 propose the contributions of this dissertation and respectively detail each of the designed OCR systems, together with experiments and results.

- Chapter 7 draws conclusions of this study and proposes perspectives for future directions.

Chapter 2

Related work

2.1 Introduction

Since the 90's, research in OCR systems has been an abundant field of research in pattern recognition and computer vision. Prior works have mainly focused on systems operating on scanned documents and handwritten texts. Recently a considerable progress has been made in different other domains such as text recognition in historical documents, in images and in videos. Fig. 2.1 depicts some applications of text recognition and illustrates their related acquisition modalities.

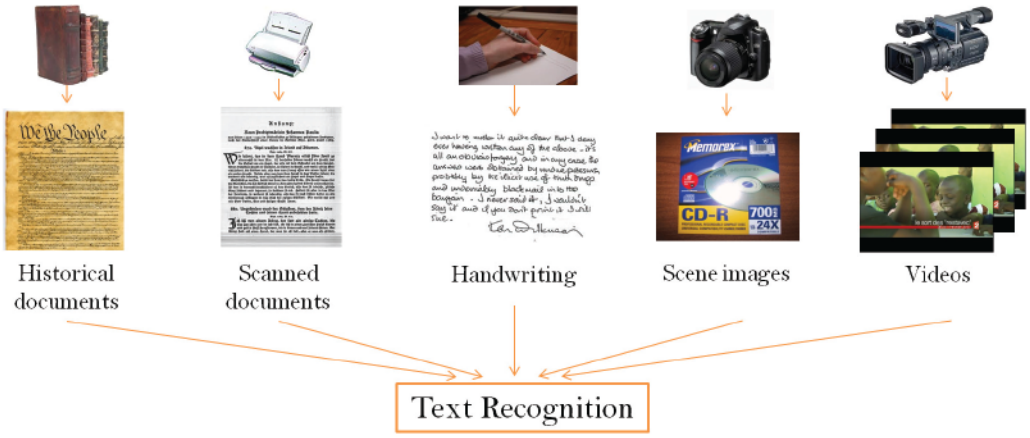

Figure 2.1: Examples of text recognition tasks.

This chapter presents a survey of main works achieved in the specific field of text recognition in images and videos. The community has distinguished different issues related to this recognition problem, including text detection [LS96, LDK00, YHGZ05, DG08, PST10], text tracking and localization [QLWS07], text image preprocessing [HYZ02, COB04, YPX09, YW05, LPM07], character segmentation [CL02, MTG06, SGD09, PSST11], character recognition [CGR05, KHE05, SG07a] and text recognition

[ZC03, YEYT11, EGS11]. A review of the new advances achieved, in the last years, in these issues has been presented in [SPB12].

In this thesis, we focus on the steps involved in the sole text recognition task and present here successively their related work ranging from prior to recent studies. Fig. 2.2 shows these different steps and their interactions. Notice that, in our work, we use an existing text detector that we adapt to our datasets (*cf.* chapter 3).

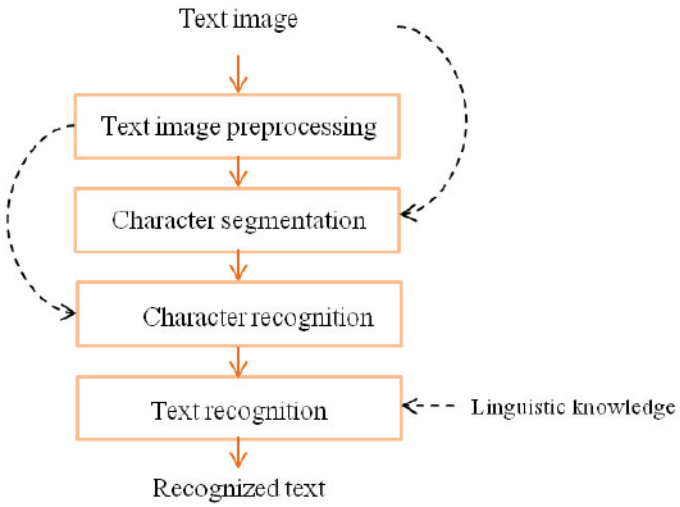

Figure 2.2: Text recognition steps.

2.2 Text image preprocessing

Texts embedded or captured in images and videos can be of different sizes, colors and can have complex backgrounds with different kinds of distortions (noise, blur, occlusion, etc.). These facts make text recognition a hard task and may lead to poor results.

To overcome these difficulties and achieve a good OCR accuracy, some researchers proposed to insert a step of text image enhancement before applying any recognizer. The idea is to improve the quality of the text image and remove noise so that a clear text image with clean background, easy to recognize, can be obtained. Most designed preprocessing methods can be classified into three main categories: text image binarization, super-resolution and multi-frame integration.

2.2.1 Text image binarization

A large number of methods, focusing on preprocessing, propose the binarization as a necessary step useful for good recognition performance. The aim of these methods is

to analyze the text image and label its pixels to text and non-text regions. A binary image where the foreground—namely the text or the characters—and the background are distinguished is then obtained.

This preprocessing is particularly challenging in the case of color text images and video texts because of the complex background and various artifacts. Different works thus present several techniques of binarization, that can be classified into two major categories: statistical thresholding-based methods and machine learning-based methods.

Statistical thresholding-based methods

Prior binarization techniques relied on a global thresholding method where only one threshold is computed for the whole text image. In [Ots75], Otsu *et al.* assumed that an image contains two classes of pixels, namely the text and the background. By analyzing the gray-level histogram, they proposed a method which selects the global threshold that maximizes the separability between the two classes. Sato *et al.* [SKHS98] also presented a global thresholding binarization technique. Their idea is to use four directional filters to extract character features, then apply a fixed threshold to the output of the filters. These approaches obtain good results on images with high contrast and clean background, which is rarely the case of natural "scene" text images and video texts.

In 1985, Niblack *et al.* [Nib85] proposed to improve binarization performance by computing adaptive thresholds which take into account both the gray-level histogram and neighborhood information. The designed algorithm determines for each pixel a threshold T calculated on the basis of the intensity statistics within a local neighborhood in a window of fixed size:

$$T = \mu + k\sigma \tag{2.1}$$

where μ and σ are the standard deviation and the mean of the gray value within the window, and k is a fixed parameter set to -0.2.

However, this binarization technique can introduce some noise when considering windows in regions of background with no text. In order to overcome this drawback, Sauvola *et al.* [SSHP97] defined a new formula to calculate the local thresholds assuming that pixels of text are usually near to 0 and those of the background are close to 255:

$$T = \mu(1 - k(1 - \frac{\sigma}{R})) \tag{2.2}$$

where R is the dynamics of the standard deviation.

Fig. 2.3 provides some results of thresholding binarization techniques applied to "scene" text images.

Chen *et al.* [CY04] also improved Niblack *et al.*'s algorithm by considering a local neighborhood in a window of adaptive size. Window sizes are thus selected to lead to

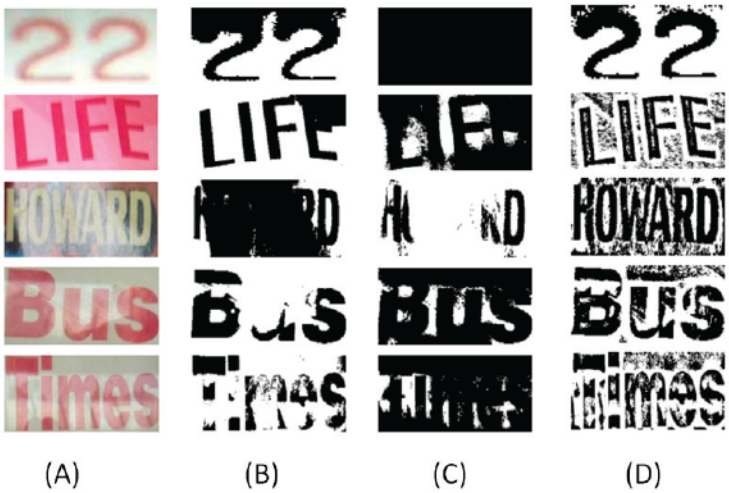

Figure 2.3: Comparison of some thresholding-based binarizations: (A) original text images, (B), (C) and (D) are respectively results of methods presented in [Ots75], [SSHP97] and [Nib85] (figure provided by [MAJ11]).

smooth areas with low standard deviation (under a fixed threshold).

$$T_r = \mu_r + k\sigma_r \tag{2.3}$$

where r is the adaptive size of windows.

Other authors (Wolf *et al.* [WJ04]) were also interested in Niblack *et al.*'s algorithm and proposed to formulate the decision of binarization in terms of local contrast instead of gray values. A new formula of threshold is thus defined:

$$T = (1-a)\mu + aM + a\frac{\sigma}{R}(\mu - M) \tag{2.4}$$

where a is a parameter that controls the incertitude related to μ and M is the minimum gray value of the whole text image.

Recently, another binarization approach was proposed to specifically deal with images with complex background and low contrast [ZLT10]. Using a Canny edge detector, text boundaries are first selected. Then, the inner side of contours is determined relying on a local threshold method. Finally, contours are filled up to form text regions. This approach enables to enhance text image quality and increase recognition performance. However, in the case of small characters, the detection of character boundaries usually fails, leading to poor binarization results.

Ntirogiannis *et al.* [NGP11] also proposed a thresholding-based binarization. After detecting the text baselines, these authors identify the main body of the text and determine the stroke width. A first thresholding step is applied with two different thresholds: one for the baselines area and one for the remaining image area. Using the

obtained result, a convex hills analysis is performed to update the text body region and apply a final thresholding step.

Machine learning-based methods

Most thresholding methods rely on the computation of one global threshold or local ones that serve to make the binarization decision. However, this kind of approach usually fails when the text and background have similar textures or colors or in the case of complex backgrounds. To solve this problem, machine learning-based techniques were proposed.

Several works have considered the text image binarization as a segmentation problem and proposed to use the clustering as an effective technique able to take into account color information as gray level ones. Particularly, the K-means clustering algorithm, which aims to separate a set of elements into K clusters so that the inter-cluster similarity is low and the intra-cluster one is high, was adopted by many researches.

In 2004, Gllavata *et al.* [GESF04] presented a method that separates the text from the background using the K-means clustering. Their algorithm consists of three main steps. First, the dominating text color is estimated using a color quantization method [Wu96]. A wavelet transform is then applied to the image in order to select character features. Finally, pixels are classified into two clusters (namely the text and the background) relying on a K-means algorithm applied to feature vectors that include the dominating color and the normalized wavelet coefficients.

The K-means clustering technique was also chosen by Song *et al.* [SLP+08]. These authors apply the K-means algorithm to the RGB values of pixels in order to segment the text image into K clusters (K can be 2, 3 or 4). Obtained clusters are then analyzed depending on their centers; clusters with the highest centers are considered as text while the others are considered as background. Experiments showed that best performances are obtained with K equal to 3.

Recently, Wakahara *et al.* [WK11] also used the K-means clustering to binarize color "scene" text images with complex background. The first step of their approach is to segment the text images into K clusters using the K-means algorithm. A set of $2^K - 1$ tentatively binarized images are then generated by the dichotomization of the K obtained clusters. Every binarized image is segmented into a sequence of single character-like images relying on an aspect character ratio. Using Support Vector Machines (SVM), for each binarized text image, the "character-likeliness" of each single character-like image is evaluated. Finally, the text image with the maximum average of "character-likeliness" is selected.

However, the main drawback of the K-means algorithm is its inability to model local information when segmenting the image. The decision of binarization of each pixel is made only based on its intensity value without any consideration of its neighborhood, which makes binarization particularly sensitive to noise and complex background. To overcome this problem, some authors have proposed to rely on the Markov

Random Field (MRF) modeling. Indeed, MRF models, which are stochastic models, permit to estimate the conditional probability of each pixel depending on its intensity and on a defined surrounding neighborhood.

In 2002, Wolf *et al.* [WD02] proposed to use MRF to model the prior in a maximum a posteriori (MAP) estimator. The obtained MAP is optimized with a Simulated Annealing (SA) algorithm [KGJV83]. Even though the neighborhood considered in the MRF was learned from a set of training data, obtained results did not outperform thresholding-based methods. This result can be explained by the weakness of the optimization algorithm.

The binarization method of Chen *et al.* [COB02] relies also on the MRF technique. First, a Gaussian Mixture Model (GMM) is learned with an Expectation-Maximization (EM) algorithm to estimate distributions of text and background. Then, a MAP optimization is performed introducing some prior modeled with a MRF and using a fast Iterated Conditional Modes (ICM) algorithm [GG84].

Mishra *et al.* [MAJ11] focused on the case of natural scene images and presented a method robust to different kinds of degradations. After combining a MRF and a GMM to model text and background colors, these authors selected seeds for each class (*cf.* Fig. 2.4) and applied a standard graph cut algorithm to obtain an initial binarization. An iterative process, alternating the re-estimation of the Gaussian mixture and the application of the graph cut algorithm, is applied until obtaining a clean binary image (*i.e.*, convergence conditions).

Figure 2.4: An example of selected seeds: (A) original "scene" text image, and (B) text and background seeds: text ones are in blue and background ones in red [MAJ11].

The GMM technique was also used by Ye *et al.* [YGH04] but applied to a selected set of pixels instead of the whole image pixels. In this work, images are first analyzed with a Canny detector in order to select some parts of text pixels (*cf.* Fig. 2.5) that serve to train a GMM and estimate color models of the text and the background. The binarization step is performed introducing some spatial connectivity information and using the connected components analysis in order to remove noise.

Li *et al.* [LBWX10] integrated local visual information and contextual label information in a Conditional Random Field (CRF) based approach. Their aim is to remove

Figure 2.5: Illustration of text pixel selection: (A) original text image, (B) Canny detector result, (C) parallel edge lines and (D) selected text pixels [YGH04].

local ambiguities and improve binarization performance. This method was dedicated to natural "scene" text images with complex background and obtained good results.

Saïdane *et al.* [SG07b] introduced an automatic binarization step based on a learning model particularly robust to complex background and low resolution. In this work, a convolutional neural network (ConvNet) [LB95] is trained to take as input a color text image and to return a binary image where text pixels are set to 0 and background ones are set to 255. The architecture of the network consists of 4 layers: a convolution layer, a sub-sampling layer, an up-sampling layer and an inverse convolution layer.

Recently, Shi *et al.* [SXWZ12] addressed the binarization in a different way. The main idea of their method is to apply an adaptive graph cut algorithm where some pixels, selected considering the sole properties of the text, are identified as initial seeds. The proposed binarization involves three steps. Aiming to provide local adaptive binarization more adapted to each character or part of character, the text image is first split into several sub-images. This is done relying on a set of processings including edge detection, connected components analysis, and filtering. Each sub-image is then analyzed in order to select some pixels with a high confidence to belong to text and to background. Obtained pixels are finally considered as hard constraints seeds for the graph cut algorithm applied to segment each sub-image.

2.2.2 Super-resolution

In addition to the classical step of binarization, many researches proposed to increase the resolution of text images in order to obtain images of better quality where texts are easier to recognize. Actually, the issue of super-resolution was a subject of several works for general image processing [FREM04, Fat07]. However, the special case of text images remains particularly challenging because of the specifics of the patterns (such as the size and the thickness of characters) and the high number of edges. In this context, several attempts were made to adapt classical super-resolution methods and to design new ones specifically adapted to text images and text videos.

Interpolation methods are the most common approach used for image resolution enhancement. In [YGH04], before the binarization step, authors applied a sub-pixel

interpolation in order to get text images with more important regions containing clear characters. Wolf *et al.* [WJC02] also applied a bi-linear interpolation to texts embedded in videos aiming at increasing their resolution enough so that commercial OCR softwares could process them (actually, commercial OCR softwares are mainly developed to deal with scanned documents with high resolution).

Nevertheless, due to their sensitivity to noise, interpolation methods usually deteriorate the quality of text images introducing some irregularities around edges. To tackle these difficulties, Li *et al.* [LD00] presented a super-resolution method that relies on a projection onto convex sets (POCS) [PSMT97]. First, a bi-linear interpolation is applied to the text image in order to obtain an initial result. Then, considering some sets of convex constraints, for each pixel, a residual function is evaluated and back-projected. This is repeated iteratively until convergence. Fig. 2.6 illustrates this method applied to a text detected in a video frame.

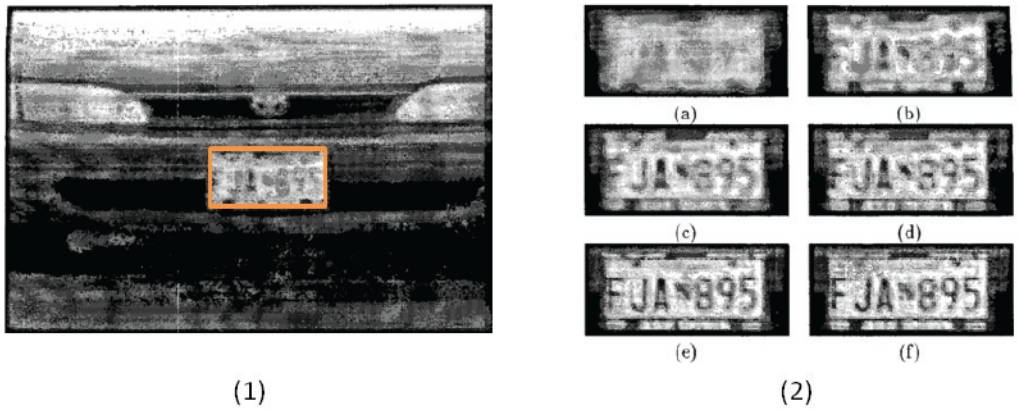

Figure 2.6: Illustration of Li *et al.*'s method [LD00]: (1) a video frame with blurred license plate, (2) the results of the POCS-based super-resolution ((a)-(f) correspond to the images obtained at iterations 1, 5, 10, 20, 50 and 100).

To increase the robustness against impulsive noise, others authors proposed approaches based on the quadratic 2D Teager filter [MS01], which can be considered as a useful contrast enhancer for images. The idea is to apply the Teager filter to text images in order to get a high pass of input images (*i.e.*, images with highlighted edges and eliminated noise) and thus enable a specific processing for edges regions.

In [MTM05], authors focused on the case of video text. After the motion estimation with Taylor series and the application of the 2D Teager filter to all frames containing text, original frames and their obtained high pass images are warped and bi-linearly interpolated to get two super-resolution images. Images are then fused and denoised for a final result. Fig. 2.7 illustrates the results of the different steps of this method.

To introduce a further robustness to camera motion, Malczeweski *et al.* [MS06] integrated "projective fit" and "projective flow" techniques to estimate motion param-

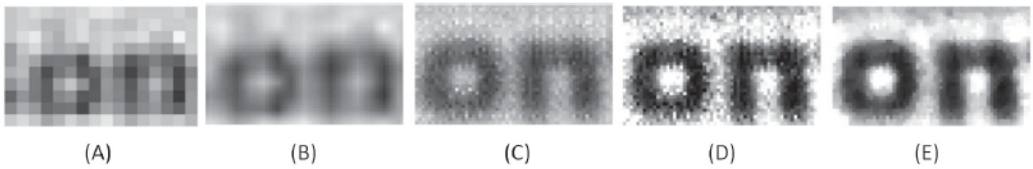

Figure 2.7: Illustration of Mancas-Thillou *et al.*'s method [MTM05]: (A) original low resolution text image, (B) bi-linear interpolation of (A), (C) super resolution output without the Teager-filtered image, (D) and (E) the super resolution results respectively before and after applying the denoising step.

eters and relocate texts in a regular grid using the iterative POCS combined with a biharmonic spline interpolation. Obtained text images and original ones are then filtered with the 2D Teager filter and fused to obtain the final super-resolution image.

Banerjee *et al.* [BJ08] were interested in performing a super-resolution algorithm able to create an image with smooth regions (inside and outside the text region) and sharp discontinuities across edges. To that end, they first evaluated the edge directed tangent field of text images, and incorporated this edge information into the formulation of the energy function in a MRF model. The super-resolution text image is finally obtained by reducing the energy function.

2.2.3 Multi-frame integration

Multi-frame integration is a preprocessing step applied only in the case of video text data. The key idea of this task is to take advantage of the temporal redundancy of a text appearing for some time in many frames in order to get a clear text image with clean background and high contrast. Among the proposed methods, two major approaches can be distinguished: those based on the averaging technique and those based on the minimum/maximum integration.

In the first category, the enhancement of text image and cleanness of the background is achieved by averaging values of pixels over multiple frames where the text appears. In [LD99], after detecting texts embedded in videos, the mean of the whole sequence of text images is computed, to obtain an image containing enhanced text region on a clean background where noise and artifacts are reduced. Hua *et al.* [HYZ02] were also interested in solving the problem related to complex backgrounds by utilizing the multi-frame integration. They proposed to improve Li *et al.*'s method by removing frames where text is not clear. Assuming that texts embedded in videos can be clear in some frames and less clear in other ones (because of a low contrast or a complex background), a selection of clean text frames (at worse, clean blocks of texts) is performed. This selection is based on the analysis of the contrast in the area around the bounding box of detected texts. An averaging of obtained frames and blocks is finally applied to get a clear text image.

The second category of approaches relies on a time-based minimum pixel value (or

variation) search. Assuming that a text keeps the same color during its appearance and is brighter than the background, the enhanced text image is obtained by assigning to each pixel a value equal to the minimum of all its intensities over the different frames containing the text [SKH+99]. However, this method cannot be applied in the case of dark text on bright background. To tackle this problem, Lienhart *et al.* [LW02] have proposed another method that takes the temporal redundancy into account to identify the background by its temporal variation. They considered that background often changes significantly through time while text usually keeps its main color and shape. Relying on these hypotheses, a minimum/maximum operation is applied in order to obtain, for each pixel, its two extremum values (the minimum and the maximum ones). The analysis of these values enables to distinguish text from background. Wang *et al.* [WJW04] also used a time-based minimum (or maximum) pixel value search to obtain two images: one containing for each pixel the maximum value through time and another containing for each pixel the minimum value through time. Both images are then analyzed with a Sobel filter in order to detect edges. The image with the fewest edges, considered as the one with the uniform color, is selected to be used for text blocks classification (into background or text).

Recently, Yi *et al.* [YPX09] proposed to combine both techniques—namely averaging and time-based minimum—aiming at getting clean and clear texts to improve recognition performance. Moreover, a text-intensity detector is integrated to filter blurred texts and remove bad effects. Experiments carried out by these authors showed that their method obtains good results, outperforming Hua *et al.*'s method [HYZ02]. Fig. 2.8 shows a comparison between results obtained with approaches presented in [YPX09], [HYZ02] and [LW02].

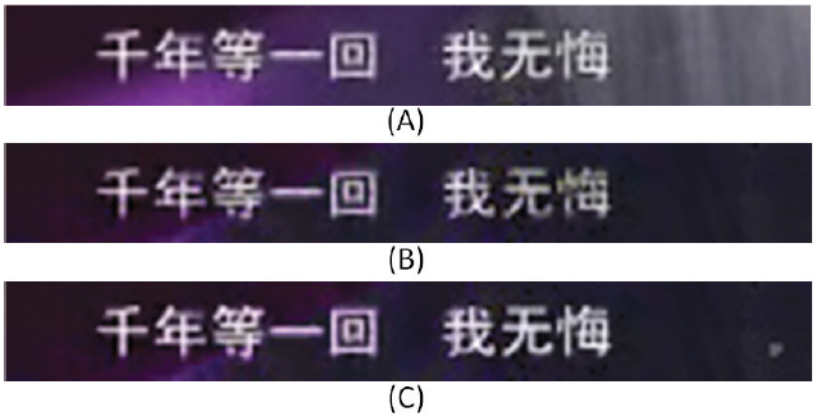

Figure 2.8: Comparison of some multi-frame integration techniques: (A), (B) and (C) are results obtained with approaches presented in [HYZ02], [LW02] and [YPX09] (figure provided by [YPX09]).

2.3 Character segmentation

Before recognizing characters, a step of segmentation is often necessary to separate the text into single characters easier to analyze and classify. In this context, the automatic segmentation of characters is considered as a crucial step in most OCR systems; each segmentation error (*i.e.*, either an over-segmentation or a sub-segmentation) involves recognition errors and directly reduces OCR performance.

In the literature, several approaches were dedicated to this issue and have proposed different strategies to address this problem. Casey *et al.* provided a complete survey of character segmentation methods in [CL02] and identified three main categories of approaches, that we adopt in our work:

- *dissection segmentation* approaches, which segment characters using only image analysis processing techniques,
- *recognition-based segmentation* approaches, which segment the text into single characters taking into account recognition results,
- *segmentation-free* approaches, which consist in recognizing a succession of characters directly from the whole text image without any segmentation.

2.3.1 Dissection segmentation

The dissection segmentation category includes all methods that tend to separate characters mainly relying on techniques of image or/and signal processing.

Projection profile-based segmentation

The projection profile analysis technique is one of the most prior common dissection segmentations. It consists in a kind of compact representation of the spatial distribution of the text image pixels content (*i.e.*, intensity values). This technique allows to estimate columns that do not contain text pixels and thus to deduce separations between characters.

In [LSA94], the authors addressed the problem of touching character segmentation in printed documents and presented a method based on profile projection analysis. Two kinds of profile projection are computed for binary text images: the first one counts for each vertical column the total number of black pixels (*i.e.*, pixels assumed to belong to the text), and the second one represents each column by the position of top pixel belonging to the external text contour. Two thresholds are then applied to build projection profiles in order to select segmentation positions. Nevertheless, this approach is only effective on text images of high quality and whose binarization was successful (clean text and background). For this reason, the projection profile technique was mainly used for machine printed characters segmentation but rarely in the case of captured or embedded texts.

[MZJ+07] is one of the few works dedicated to video text recognition that employed the projection profile method. In this method, gray scale text images are used directly (without any binarization) to build an intensity profile projection representing the sums of pixel values over each column. Obtained profiles are then thresholded to get segmentation positions.

Recently, Shivakumara *et al.* [SBS+11] proposed a character segmentation method based on a modified profile projection technique (*cf.* Fig. 2.9). This method first relies on a binarization process, and evaluates, for each column, a text height difference (THd) defined as the distance between the first and the last pixels within the column belonging to the text. As in previous methods, profiles are built and segmentation boundaries are obtained by thresholding computed profiles.

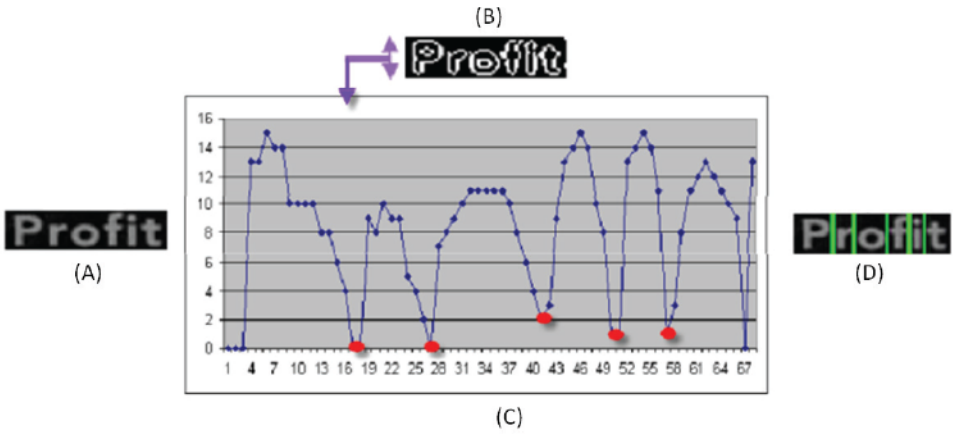

Figure 2.9: Illustration of Shivakumara *et al.*'s segmentation technique [SBS+11]: (A) input text image, (B) boundaries of the text image used to evaluate the THd, (C) profiles of the computed THd, red dots (that correspond to THd less than two pixels) identify segmentation positions, and (D) obtained character segmentation result (figure provided by [SBS+11]).

Other authors [SKH+99] proposed to use the projection profile technique as a primary segmentation. Text images are first filtered and binarized (using a thresholding-based method) to compute vertical and horizontal projection profile. Obtained profiles are employed to determine candidates for character segmentation that will be discussed depending on recognition results (see next subsection).

Shortest path algorithm-based segmentation

In [KHE05], Kopf *et al.* tackled the problem of character segmentation in low resolution images and videos and showed that the projection profile technique is not well adapted to this case (since results of projection profile are sensitive to thresholds and can thus lead to missed separations and split characters). Instead, the authors

proposed a method based on Dijkstra's shortest path algorithm in order to segment characters taking into account the local morphology of images. The idea is to search paths, from the top to the bottom of the text image, that can correspond to separations between characters. Selected paths are computed as successions of pixels having the lowest difference between their values.

Tse *et al.* [TCJY07] also used the shortest path algorithm to segment characters in gray document images. The designed method applies the algorithm only to separate touching characters. After an initial step that identifies touching characters, the algorithm is performed to find a path from the top to the bottom that corresponds to the separation between touching characters. In this approach, selected paths are computed to be the ones with the lowest cost (*i.e.*, the path whose pixels intensities sum is the lowest).

Recently, Phan *et al.* [PSST11] proposed a robust shortest path-based segmentation that includes two additional steps: an initial one to select candidate cuts and a final one to remove false ones (*i.e.*, over-segmentations). The first step relies on a gradient vector flow to identify pixels of potential paths. Using obtained pixels, several paths crossing the text image from the top to the bottom are determined, then checked to see if they correspond either to segmentations between characters or not. To do so, the authors assume that for a true segmentation, the path computed from the top to the bottom is close to the one computed from the bottom to the top. Thus for each obtained path, a backward path computation is performed to compare both paths and decide either to keep or discard this path. Fig. 2.10 shows a comparison between texts images segmented with Kopf *et al.*'s method [KHE05] and Phan *et al.*'s one [PSST11].

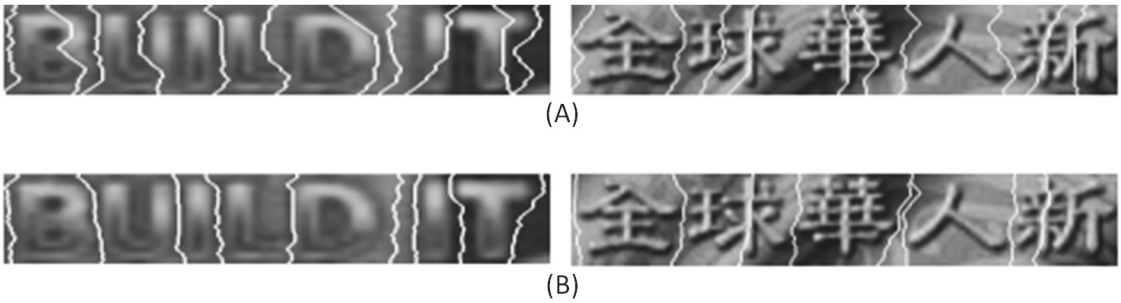

Figure 2.10: Comparison of shortest path algorithm-based segmentation techniques: (A) and (B) are results obtained with methods respectively proposed in [KHE05] and in [PSST11] (figure provided by [PSST11]).

Split and merge algorithm-based segmentation

Boundary-based segmentations—namely the projection profile analysis and the shortest path algorithm—usually seek to segment text images by detecting edges and sepa-

rations between characters. In addition to these segmentations, region-based segmentations were also used to solve the problem of character segmentation. In contrast to boundary-based approaches, region-based ones tend to identify segmented regions by exploiting the homogeneity of the spatial information.

In [YBYK11], a split and merge algorithm was used to separate characters in license plate images. After a first step of text image binarization, a connected component analysis is applied to get an initial set of isolated blobs. Depending on some geometrical properties (such as the width, the high, the area, etc.), non-character blobs (blobs not containing single characters) are selected to be removed (if they contain noise), merged (if they contain fragments of one character) or split (if they contain more than one character). Fig. 2.11 illustrates the different steps of this method.

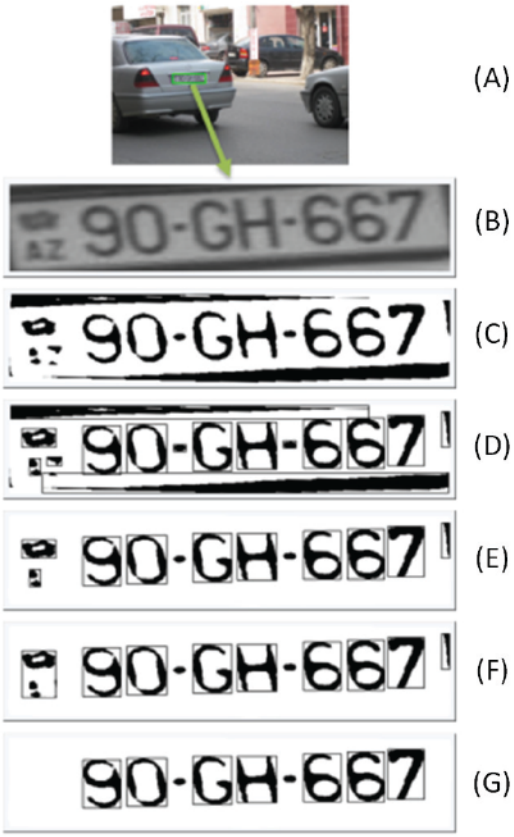

Figure 2.11: Steps of Yoon *et al.*'s character segmentation method [YBYK11]: (A) the scene image, (B) the extracted license plate image, (C) the binary image of (B), (D) the connected component analysis result, (E) removing noisy blobs, (F) merging blobs, and (G) the final segmentation result after blob selection (figure provided by [YBYK11]).

Huang *et al.* [HMZ09] also employed a split and merge algorithm to segment

characters in texts extracted from videos. Their method involves two main steps: an initial segmentation based on the projection profile analysis, and a merge and split step based on assumptions about the width and the height of characters.

Other dissection segmentation

Besides the approaches presented above, other original methods for character segmentation were designed.

Lue *et al.* [LWC+10] proposed to combine a connected component analysis and some periphery features to segment characters. Their method first analyzes binarized text images in order to provide an initial segmentation and select touched characters. A specific segmentation based on periphery features is then performed to deal with identified touched characters. Used features are composed of 32 character contour values extracted to characterize the character periphery (*cf.* Fig. 2.12). Finally, a verification step is applied to check obtained segmentation results.

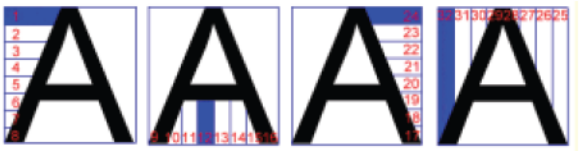

Figure 2.12: Illustration of periphery features [LWC+10].

In [SG08], Saïdane and Garcia proposed an original method for character segmentation in color "scene" text images. The designed automatic segmentation is based on a convolutional neural network model that was trained to determine segmentation positions. This network takes as input a color text image, is composed of three convolution layers, and returns a vector of outputs indicating positions of segmentation borders.

2.3.2 Recognition-based segmentation

Assuming that character segmentation is a complex problem, some attempts to explore the interaction between the segmentation and the recognition have been done. The idea is to take advantage of recognition outputs in the segmentation process to validate segmented regions. Two methodologies can be distinguished: the first, most common, one consists in building concurrent segmentations and relies on a character recognizer to accept correct ones and reject false candidates, while the second works by adjusting segmentation parameters based on recognition results.

In the first methodology, Sato *et al.*'s technique [SKHS98, SKH+99] is one of the first recognition-based approaches. In this work, a simple segmentation step, based on profile projection technique, is first applied to select candidates. Considering obtained segmentations, several potential character images are identified and analyzed. Each

one is characterized by a value encoding its similarity with its most probable class of character. This similarity (or matching) value is computed with a correlation metric between some reference patterns and the normalized character image. Once recognition results are obtained, a final selection of segmentations is made keeping only those corresponding to segmented regions with high similarity values.

Saïdane *et al.* [SGD09] also proposed a method that aims to get all possible segmentations before removing false ones using recognition results. An over-segmentation step is first performed using the method described in [SG08]. A graph model is built by creating connections between neighboring segmentation candidates and integrating character recognition results to weight connections. The character recognition step is performed with the help of the neural model presented in [SG07a] (*cf.* subsection 2.4.2). Using a best path search algorithm, the most probable sequence of characters is finally obtained keeping correct segmentations and removing false ones.

Another recognition-based segmentation dedicated to degraded text recognition was presented by Jun *et al.* [JHF+05]. In this method, an initial basic set of segments is first computed using a Principal Component Analysis (PCA) [Jol05]. Then, for each segment, some character features are extracted in order to identify segments containing single characters and recognize them. The remaining segments are merged depending on their features and spatial localizations to recognize characters that they contain.

In contrast to these methods, Mancas-Thillou and Bernard [MTG06] presented a segmentation method that uses character recognition to obtain the optimal parameters of the segmentation algorithm. The designed approach relies on Log-Gabor filters able to locate and detect separations between characters. The choice of the Log-Gabor filters parameter—namely the bandwidth—was performed depending on recognition results after applying an OCR. The character recognizer is based on a multilayer perceptron (MLP) network that was trained to classify some extracted features.

2.3.3 Segmentation-free

In contrast to previous segmentation methods, few other authors have proposed OCR systems that do not rely on any segmentation process. This family of approaches addresses the problem of text recognition in complex images where separations between characters are particularly hard to identify (extremely complex background, broken characters, attached characters, etc.). These approaches do not aim at separating characters; instead they consider entire word text images as single units and recognize them as such.

In [KSIA04], a method for text recognition dedicated to natural scene image retrieval is presented. No character segmentation is performed; instead a scanning process is applied to a set of generated multi-resolution text images. Resulting clipped regions are then classified to confirm if they contain characters or not and to recog-

nize characters. The character recognition step is based on a features extraction and a matching process that consists in projecting the features vector to built subspace then detecting the class that corresponds the best to the input character image.

Negishi et al. [NIOA05] also performed a method able to recognize texts captured in scene images without any character segmentation. The main idea of this method consists in extracting some features (namely corners and curves, each one characterized by a histogram of edge directions) from the entire text image. Then, a voting algorithm able to combine obtained features and analyze their spatial localization is used, in order to recognize characters present in the text image.

In [FF09], Fan and Fan addressed the automatic license plate recognition and proposed a graphical model able to recognize a sequence of characters without any explicit segmentation. To that end, a Markov network is designed to formulate the problem of the joint segmentation and recognition as a 1-D case problem. The network consists of two layers (one to represent the segmentation, the other one to represent the recognition) and takes as input vectors of low-level features extracted from the text image.

Recently, Wang *et al.* [WB10] proposed a word recognition approach that relies on a generic object recognition method, in which words are considered as objects to be classified. Without any character segmentation, a character detection step is performed to localize characters then recognize them with the help of some extracted features. The words are finally recognized using a pictorial structure of recognized characters, and classified using a dictionary.

2.4 Character recognition

The goal of text image recognition is to identify the sequence of characters contained in the image. In this issue, the step of character recognition is an essential task since it permits to label individual character images and thus recognize the sequence of characters.

During last years, a great number of methods were proposed. Among these character recognizers, two main categories of approaches can be distinguished: pattern matching-based approaches and machine learning-based approaches.

2.4.1 Pattern matching-based approaches

Pattern matching-based approaches are methods that tend to classify character images relying on matching algorithms able to measure the similarity between some extracted patterns and a database of patterns. In this family of approaches, characters are usually characterized by a set of hand-crafted features (such as edges, profiles and texture descriptors). Typically, a database of models of features is first generated. Then, for each character image, appropriate features are extracted and matched against the database in order to recognize the class of the input image.

In 2003, Zhang *et al.* proposed a character recognition method based on a likelihood measurement [ZC03]. Character images are first binarized and analyzed to extract a set of features including Zernike magnitude, direction proportion, first-order peripheral features, second-order peripheral features, vertical and horizontal projection profiles. This extraction step leads to an input feature vector of 207 values. This vector is compared to a dataset of vectors extracted from training examples to finally find the class with the highest likelihood.

In [KHE05], Kopf *et al.* proposed to identify characters by means of their contours. Using an extended Curvature Scale Space (CSS) technique, contours of characters are analyzed and represented by some features selected by the authors as useful for the classification. A matching algorithm able to consider geometrical transformations (*i.e.*, shifts and rotations) finally classifies extracted features and recognize characters.

Shivakumara *et al.* also defined a recognizer based on characters edges [SPLT11]. First, character images are resized to 64×64 pixels. Then a Canny detector, able to preserve the shape of the characters, is applied to get image edges. Computed edges serve to feed a recognizer defined by means of a hierarchical classification. Fig. 2.13 illustrates character edges features used for the classification. Character images are hence classified progressively using a voting algorithm.

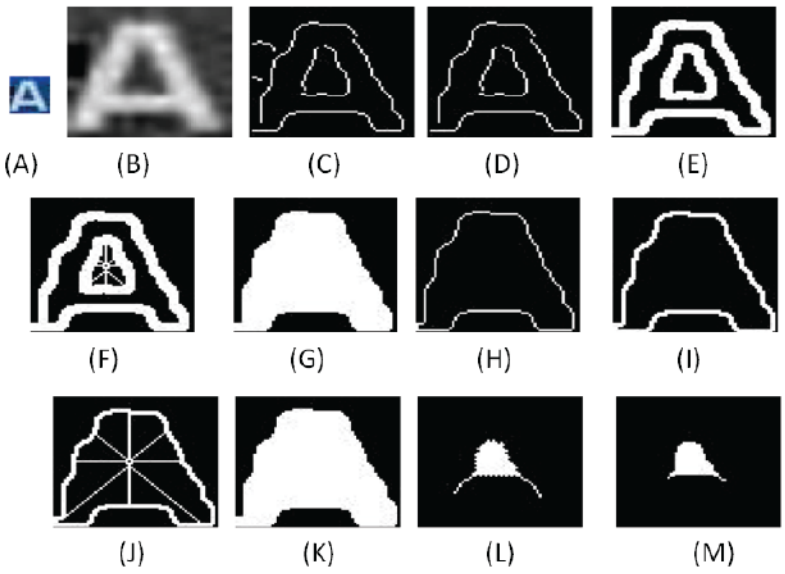

Figure 2.13: Edges features used for the hierarchical classification in Shivakumara *et al.*'s method [SPLT11]: (A) original color character image, (B) resized gray level image, (C) result of the Canny detector, (D) filtered edge map, (E) dilated edges, (F) outlets found in 8 directions from the centroid of the character, (G) filled edges, (H) perimeter of filled edges, (I) dilated perimeter, (J) outlets found in 8 directions from the centroid of the character, (K) filled edges, (L) shrunk, (M) after removing end points from (L) (figure provided by [SPLT11]).

A method specifically designed to recognize characters extracted from videos was proposed in [CGR05] to identify characters by their side-profiles. Admitting that texts embedded in videos have similar fonts, the four side-profiles (namely top, left, right and bottom as shown in Fig. 2.14) were considered as sufficient to represent and classify characters. In the presented method, character images are first binarized then analyzed to generate their corresponding side-profiles using the technique described in [JSS99]. Obtained side-profiles are finally matched against a database constructed with training samples of characters.

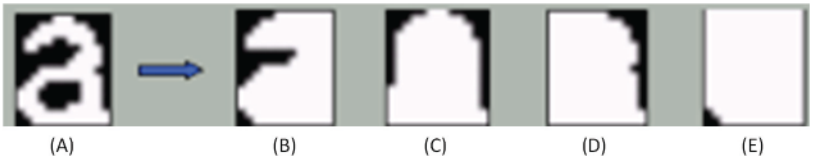

Figure 2.14: An example of profile sides of character "a": (A) binarized character image, (B), (C), (D) and (E) are respectively left, top, right and bottom profile sides (figure provided by [CGR05]).

Halima *et al.* also used a projection profile technique to recognize Arabic character images extracted from digital videos [HKA10]. Besides vertical, horizontal, diagonal and slanting diagonal profiles, some occlusion and transition features were equally used in order to define a recognition method robust to distortions and translations. Obtained features are classified with a matching step ensured by a K-nearest neighbor (KNN) algorithm. In this work, the authors demonstrated that best performance are obtained with $K = 10$.

The nearest neighborhood algorithm was also used by Iwamura *et al.* to recognize camera-captured characters [ITK10]. Their method consists in matching binarized images of characters against a dataset of stored samples in order to find the most similar one and thus identify the class of the input image. To do so, the connected components of each character image are analyzed and described by means of vectors of features which have the particularity to be affine invariant. The vectors serve to match the character image using a KNN algorithm.

Other authors proposed to rely on corners and curves to recognize characters in natural scene images. In [NIOA05], four points (namely the upper left, the upper right, the lower left and the lower right) and four curves (namely up, down, left and right) are extracted and considered as features. The step of pattern matching, that relies on a voting algorithm, is performed to allow a recognition particularly robust to distortions.

A character recognition approach based on the technique of the Gaussian Affine Transform (GAT) was designed by Yokobayashi *et al.* and applied to natural "scene" character images [YW05, YW06]. The method operates on binarized images and applies a modified GAT correlation, specifically robust to geometric transformations,

between the input image and a set of templates (one template per class). Matching scores are thus computed and analyzed to recognize the class of the input character image.

In [ODT+09], a technique of character recognition is presented, which is dedicated to low resolution images. It consists in two stages: a learning stage and a recognition stage. The first stage computes one subspace per class of character. This is done by constructing eigenvectors obtained after normalizing training data and applying an autocorrelation between examples of the same class. Fig. 2.15 shows an example of obtained eigenvectors. The recognition stage, whose purpose is to classify character images, evaluates the similarities between the input image and the eigenvectors of computed subspaces. Finally, the appropriate class is recognized as the one with which the character image matches the best.

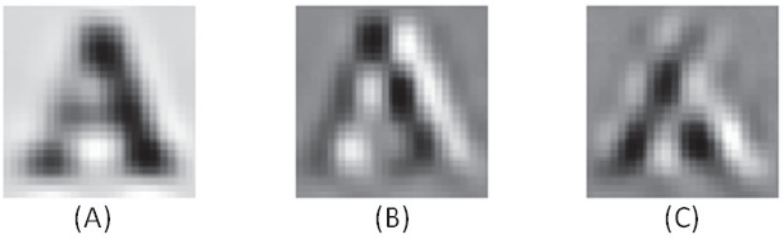

Figure 2.15: Illustration of eigenvectors obtained from 100 examples of character "A": (A), (B), and (C) are respectively the first, second and third eigenvectors (figure provided by [ODT+09]).

In order to recognize Korean characters, Park *et al.* [PLK+10] designed a method that relies on shape-based statistical features (namely the horizontal, vertical, left-diagonal and right-diagonal segments of each character). These features are extracted from each input image, then matched against a dataset of features using a minimum distance classifier.

In [UMS08], the authors focused on the special issue of recognizing texts captured by a hand-held camera moving along the text. A mosaicing-by-recognition method able to jointly create a clean text image and recognize characters is presented. The first step consists in simply concatenating *one-pixel slits* (which are central regions of the image of size $1 \times H$, with H the height of the image) of all frames containing the text. The problem of recognition is then considered as a task of deformed text recognition: the authors propose to formulate it as an optimal path problem that evaluates the similarity between transformed input frames and reference patterns. The similarity is calculated based on matching costs between one-pixel slits of the input frames and the columns of the reference patterns.

These methods generally achieve good results. However, as in any pattern recognition problem, the major issue is to define the robust features that represent the characters independently of the image resolution, the background complexity, and

potential distortions. Therefore, performance of the techniques may vary widely depending on the chosen features and the image quality.

2.4.2 Machine learning-based approaches

The key idea of this category of approaches is to train a model to classify character images so that the decision of the recognized class is taken automatically relying on previous examples observed in the training phase. Several models were proposed in the literature; some learn to classify images directly from the images themselves while others propose to use hand-crafted features or learnt features.

The SVM classifier is one of the learning-based models used to recognize characters extracted from images and videos. In [DAS01], the authors proposed to train a SVM model with some hand-crafted features to classify characters. A large range of features (namely regional features, run features, balance and symmetry features, occupancy features, skeleton features, corner features, etc.) were extracted from binarized character images and used to generate a vector of 172 values presented to the SVM model. Since the number of classes in this classification task is greater than two, several one-vs-one SVMs were trained and combined to recognize the different classes of characters.

Another model, namely the convolutional neural network, was also employed to recognize character images. The convolutional neural network is a special form of multilayer perceptron able to classify extremely variable character images without any preprocessing step. In [JSVR05], a character recognition approach based on a convolutional neural classification is proposed to identify characters in images of low resolution. The designed method takes as input an image of 29×29 pixels and returns an output vector of 72 values encoding probabilities to belong to the classes of characters. Saïdane and Garcia also presented an automatic neural-based approach for natural "scene" character recognition [SG07a]. The proposed recognizer takes as input an image of 48×48 pixels and returns an output vector of 38 values (one per class of character). Note that artificial neural networks were also used in document analysis problems including preprocessing, segmentation, and recognition. Marinai *et al.* [MGS05] present a survey of main works achieved in this domain.

Inspired by speech recognition, Som *et al.* [SCS09] defined an OCR system that uses a Hidden Markov Model (HMM) able to learn to identify characters as a sequence of states. After building bootstrap glyph models with a synthetic set of examples (one example per class), the training stage was performed by means of three iterations of a maximum likelihood process to obtain baseline glyph models. Characters are then recognized using the computed models.

Recently, a method based on unsupervised feature learning was proposed aiming to detect and recognize characters in natural scene images [CCC+11]. A set of 8 gray-scale patches is first extracted from a training data and then used to run an unsupervised learning algorithm in order to obtain a mapping from the input patches

to vectors of features. The learning stage is ensured by a variant of K-means clustering. A further step of features reduction is performed relying on a spatial pooling and yielding a final vector of 9 features for each character image of 36×36 pixels. A linear SVM model is finally trained to classify generated features and recognize characters.

2.5 Text recognition

Once text images are segmented and single characters are recognized, the final output of OCR systems (*i.e.*, recognized texts) can be calculated. In this context, several techniques were proposed to identify embedded and captured texts.

OCR systems using dissection segmentation methods usually identify recognized texts as sequences of recognized characters [KHE05, MZJ+07]. Nevertheless, some original work proposed to use more efficient methods. In [ZC03], Zhang *et al.* presented a Bayesian framework for video text recognition. The method consists in formulating the problem of words recognition with the maximum a posteriori (MAP) where character recognition results and a word knowledge model are included. The word knowledge model is computed on the basis of trained language models.

In contrast to these approaches, OCR systems relying on a character recognition-based segmentation method require well-adapted techniques able to handle different concurrent segmentations and to take into consideration character recognition results. Sato *et al.* proposed a dynamic programming method that allows to obtain the recognized text taking into account potential segmentations and character recognition results [SKH+99]. The idea is to find the sequence of segmentations (and thus the sequence of characters) that maximizes an evaluation value expressed on the basis of character recognition results. In [SGD09], the authors presented a text recognition graph model whose nodes represent concurrent segmentation borders, and edges are weighted with character recognition results. The final recognized text is obtained using a best path search algorithm applied within the built graph.

Regarding OCR systems based on segmentation-free approaches, the task is harder and efficient methods are necessary to produce the recognized texts. In [WB10], after detecting and recognizing character, a pictorial structure [FE73] is built to represent each word. This structure is then used to evaluate several character configurations of the word within the input image. Each configuration is characterized by a sort of score based on character recognition results and distortion costs (the distortion cost is evaluated depending on the spatial relationship between two successive characters). The recognized text is finally determined as the sequence of characters corresponding to the optimal character configuration. To recognize Kanji texts captured in natural scene images, a voting method was chosen by Kusashi *et al.* to identify the sequence of characters corresponding to the input image [KSIA04]. It enables to use the regularity of character placement to reduce recognition errors in complex backgrounds.

High level information, such as language properties and lexicons, can also be integrated to improve the performance of OCR systems. A recognition error correction

method was introduced by Thillou *et al.* [TFG05] and used as a post-processing step in an OCR system. Two levels of error correction were considered and applied successively:

- a low-level error correction which relies on the confidence in character recognition results;
- a high-level error correction that uses a language model—namely a character n-gram model [MS99]—able to remove some character confusions.

Som *et al.* also proposed to integrate a character n-gram language model as a post-processing step [SCS09]. Their OCR system was applied to texts embedded in Turkish broadcast news and experiments showed that the language model permits to reduce word errors significantly.

In contrast to these works, Weinman *et al.* presented an OCR system that incorporates a lexicon into the recognition scheme [WLMH09]. The idea consists in recognizing texts by means of a unified processing taking into account the character appearance, the language, the similarity with other characters and the presence in the lexicon.

2.6 Conclusion

This chapter reviewed the main approaches and presented the recent advances achieved in text recognition in images and videos. The different steps that can be involved in the text recognition task (namely text image preprocessing, character segmentation, character recognition, and text recognition) were presented and their related work discussed.

In this review, we notice that most approaches rely on preprocessing steps in order to clean text images and to improve recognition performance. Furthermore, several approaches are designed to deal only with binary images and consider that text recognition in multimedia documents—*i.e.*, in images and videos—is a simple extension of classical OCR systems, initially developed for printed documents. In addition, the main methodology in the literature consists in involving a step of character segmentation (few methods avoid that step) and computes linear separations not necessarily adapted to the morphology of the image. Concerning the character recognition issue, main existing methods use hand-crafted features adapted to a particular task or dataset.

In our work, in contrast to most state-of-the-art methods, we propose approaches adapted to the special task of text recognition in images and videos. Our approaches require no binarization step and provide novel solutions. Two categories of methods are proposed: segmentation-based and segmentation-free methods, in order to study the benefits and limits of the segmentation step. The designed approaches rely on different novel segmentation techniques and use a neural classification model trained

to recognize characters and able to learn to extract relevant features without any preprocessing step. Moreover, some linguistic knowledge is integrated in the OCR systems to tackle drawbacks of local character by character recognition and improve text recognition performance. Our different OCRs are applied and compared on two databases—namely texts embedded in videos and texts captured in natural scene images—and their results are described and discussed highlighting the benefits and the limits of each system.

Chapter 3

Datasets and experimental settings

The experiments performed in this work are carried out on two main types of multimedia documents: "caption" texts which are overlaid artificially on videos, and "scene" texts which exist naturally in scenes and are captured in images. The choice of these two categories of texts is motivated on the one hand by our desire to cover several use cases of text recognition and on the other hand by our interest to both analyze recognition problems and propose solutions able to deal with the two kinds of texts. Therefore, two different datasets are used in our work to test the OCR schemes that will be proposed and evaluate their performance in both text recognition tasks.

This chapter describes the chosen datasets—Dataset I and Dataset II—and presents their acquisition environment. It also defines the evaluation metrics used.

3.1 The "caption" text video dataset: Dataset I

Dataset I consists of 32 videos of French news broadcast programs of the France 2 channel. Each video, encoded in MPEG-4 (H. 264) format at 720x576 resolution, is about 30 minutes long. Numerous "caption" texts are embedded in each video and often provide useful clues to understand its content (texts can correspond to persons and places names, dates, titles of the news or TV reporters, etc.).

Even if, as mentioned before, our thesis work focuses only on the text recognition step, we present in the first subsection a scheme that we have designed to detect, track and extract embedded texts in videos. The datasets that have been obtained with this scheme from broadcasted TV videos are then detailed.

3.1.1 Text detection and tracking

The first step of video text recognition consists in locating and extracting texts embedded in videos. Therefore, we first perform a detection and tracking step to extract texts from videos. The aim of this processing is to extract all images corresponding

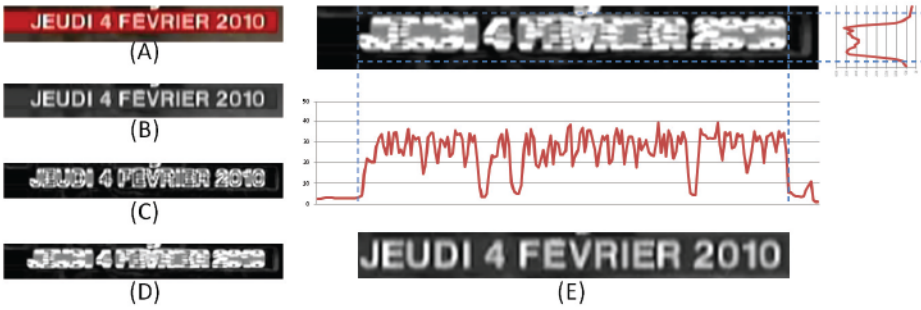

Figure 3.1: Detection post-processing: (A) and (B) are the color and the gray level text images detected with the Delakis and Garcia's detector, (C) is the Sobel operator result, (D) is the morphological dilatation of (C) and (E) is the final text image obtained after applying the profile projection technique.

to texts (using the detection step) and identify those containing the same text (by the tracking processing).

Delakis and Garcia [DG08] have provided a solution for horizontal text detection in images. This robust method relies on a convolutional neural network and obtains good results. We choose to make use of this detector and to adapt it to the case of videos. For each text detection obtained with Delakis and Garcia's detector, on a video frame, we first adjust the bounding box, in order to crop the precise area. Fig. 3.1 depicts this process: a Sobel operator, able to detect edges, is applied on the gray level image, followed by the horizontal and vertical projection. Two thresholds are chosen to discard left or right columns, and top or bottom lines.

In order to track texts embedded in videos, the detection step (namely the application of Delakis and Garcia's detector followed by the detection post-processing) is applied every two seconds. This detection has two objectives: first to locate new texts that are going to appear, and second to determine texts previously displayed that are going to disappear. Since static texts embedded in videos are considered, a tracking process that determines the starting and ending times of each localized text is introduced (*cf.* Fig. 3.2).

The tracking task is ensured by a similarity measure computed between the image of the detected text and images appearing at the same location during the 2 second sequence. In our work, the digital image correlation is used as an accurate indicator to evaluate the visual similarity and is computed, on each pixel A of the text image, as follows:

$$Correlation(A) = \frac{\sum_{B \in W_A} (I_B^1 - \bar{I}_A^1)(I_B^2 - \bar{I}_A^2)}{\sqrt{\sum_{B \in W_A} (I_B^1 - \bar{I}_A^1)^2} \sqrt{\sum_{B \in W_A} (I_B^2 - \bar{I}_A^2)^2}} \tag{3.1}$$

where A and B are pixels of the text image, W_A is a window of size 3×3 centered on A, I^1 and I^2 are the intensities of the images to compare, and $\bar{I}^1$ and $\bar{I}^2$ are the

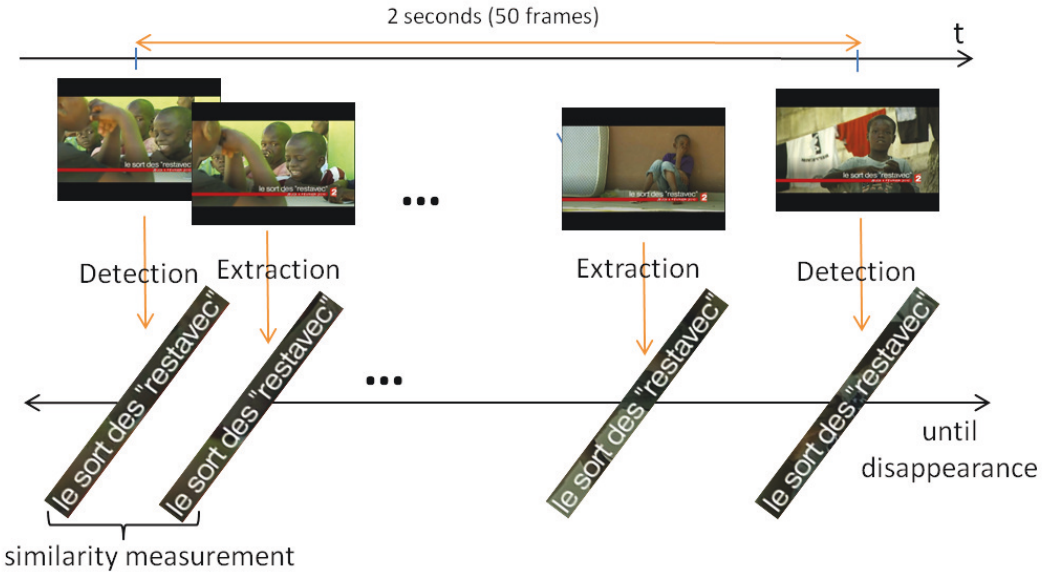

Figure 3.2: Text detection and tracking scheme in videos.

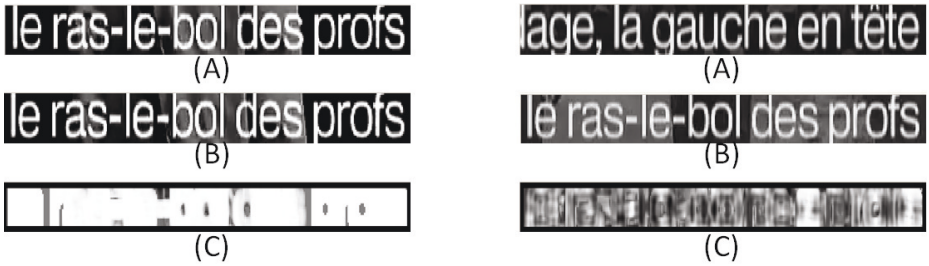

Figure 3.3: Two examples of image correlation: (A) and (B) are images to compare and (C) the output of the correlation.

mean values of I^1 and I^2 within the window. The outputs of the correlation processing (images where each pixel is represented by its correlation value computed using Eq. 3.1) are then analyzed in order to decide if correlated images correspond to the same text or not (if the correlation value is over a fixed threshold, images are considered as corresponding to the same texts, and conversely). Fig. 3.3 shows examples of correlation outputs: the example on the left illustrates the case of correlated images corresponding to the same text while the example on the right illustrates the case of images with different texts.

Once texts are extracted, they can be used to generate the required datasets.

Figure 3.4: Examples of character images of the CharDatasetI.

Figure 3.5: Examples of non valid character images of the GarbDatasetI.

3.1.2 Character dataset: CharDatasetI and GarbDatasetI

Using the detection and tracking scheme described above, "caption" texts embedded in news videos can be extracted. First, four videos of Dataset I are treated to extract their whole texts. Obtained text images are then used to generate a dataset of characters. 15,168 color images of single characters manually segmented are extracted. In this dataset, 41 character classes are considered: 26 Latin letters (small and capital letters are not distinguished), 10 Arabic numbers, 4 special characters ('.', '-', '(', and ')') and a class for spaces between words.

Obtained images are of different sizes, fonts and colors and have different backgrounds. Fig. 3.4 shows some examples of the generated characters dataset. This database, called CharDatasetI, will be used to train our character recognizer.

1,001 images of non valid characters (*i.e.*, "garbage") are also extracted. These images correspond to poorly segmented characters, to misaligned spaces between characters, or to images of multiple characters. The obtained dataset, called GarbDatasetI, will be used to train an image classifier to distinguish images corresponding to single characters from garbage. A set of these non valid character images is illustrated in Fig. 3.5.

3.1.3 Text datasets: TextDatasetI and TextTrainDatasetI

The other twenty eight videos are annotated and divided into two dataset: TextDatasetI consisting in eight videos and used to evaluate our OCR systems performance and TextTrainDatasetI consisting in the remaining twenty ones used to train our connectionist model. Each video contains about 400 words, roughly corresponding to 2,200 characters.

The text images extracted from these videos can vary a lot in terms of size (a height of 8 to 24 pixels), color, style and background (uniform and complex moving

Figure 3.6: Examples of text images of TextDatasetI.

backgrounds). Fig. 3.6 provides some examples of embedded text images of TextDatasetI.

3.2 The natural "scene" text dataset: Dataset II

Dataset II is the public database ICDAR 2003[1] which was created for a competition on "scene" character and word recognition [LPS+03]. After describing the data capture methodology of this dataset, both the character and the text datasets are presented and their specifics detailed.

3.2.1 Data capture methodology

The images of this dataset were captured using several digital cameras with different and unknown resolutions. The scene images were taken everywhere in the environment (indoor and outdoor) where texts appear on papers, boards, walls, storefronts, door, etc. All "scene" text images are provided with some information including the position and the size of the text in the captured image and the sequence of characters corresponding to the localized text. From these images two datasets were generated: a character and a text datasets containing respectively images of individual characters and images of texts extracted from scene images.

Fig. 3.7 shows an example of a scene image extracted from the ICDAR 2003 dataset.

[1] The database ICDAR 2003 is available for download at http://algoval.essex.ac.uk/icdar/Datasets.html#Robust.

Figure 3.7: An example of a "scene" image in the ICDAR 2003 dataset.

3.2.2 Character dataset: CharDatasetII and GarbDatasetII

The character database, called CharDatasetII, consists of 5,689 color images of isolated characters. The database contains 36 classes of characters: 26 Latin letters (including small and capital letters) and 10 Arabic numbers. Provided characters are of different colors and fonts and were captured with a variety of sizes ranging from some pixels to hundreds pixels of height. Fig. 3.8 illustrates a sample of character images of CharDatasetII.

In addition to the character dataset, we used another provided set of text images, called TextTrainDatasetII, (originally created for training purpose) to extract 4,056 images of non valid characters. As GarbDatasetI, the obtained dataset, called GarbDatasetII, contains mainly poorly segmented characters, spaces between characters and images of multiple characters (*cf.* Fig. 3.9).

3.2.3 Text datasets: TextDatasetII and TextTrainDatasetII

The text databases, called TextDatasetII and TextTrainDatasetII, consist respectively of 1,110 and 1146 English "scene" texts extracted from the captured images. These databases contain images of isolated words (*i.e.*, the term word is used loosely here to mean any string of characters) with characters printed, written and painted in various fonts and colors (*cf.* Fig. 3.10). The images are captured under uncontrolled acquisition conditions and present several kinds of distortions (non uniform illumination, occlusions, shadows, blur, etc.). Texts are also of different sizes (a height of 12 to 504 pixels) and appear on complex and textured backgrounds. These issues make the problem of recognizing text in this dataset a challenging problem. One can notice

Figure 3.8: Examples of character images of the CharDatasetII.

Figure 3.9: Examples of non valid character images of GarbDatasetII.

Figure 3.10: Examples of "scene" text images of TextDatasetII.

that in some cases, even humans can fail reading them.

In our work, TextDatasetII is used to evaluate the proposed OCR approaches and to compare obtained performances to those of state-of-the-art methods, while TextTrainDatasetII is used to train a proposed connectionist model.

3.3 Evaluation metrics

The recognition performance of our character classifier on CharDatasetI and CharDatasetII is evaluated with the help of the classical character recognition rate measurement, calculated as follows:

$$CharRecoRate = \frac{\#characters\ correctly\ recognized}{\#characters\ to\ recognize} \tag{3.2}$$

To calculate the recognition performance of our complete text recognition system on TextDatasetI and TextDatasetII, the two following metrics were chosen:

- The character recognition rate which evaluates the percentage of characters correctly recognized. It is calculated using the Levenshtein distance[2] as follows:

$$CharRecoRate = \frac{\#characters\ to\ recognize - \sum d_L(recognized\ text, ground\ truth)}{\#characters\ to\ recognize} \tag{3.3}$$

[2]The Levenshtein distance [Nov66] is a metric that permits to evaluate the similarity between two strings; it calculates the minimum number of operations (including the insertion, the deletion, and the substitution of a single character) to transform a string into another one.

where d_L is the Levenshtein distance calculated for each text image in the dataset and *ground truth* is the correct text to recognize and is obtained with a manual annotation step and *#characters to recognize* is the total number of characters in all ground truth texts of the dataset.

- The word recognition rate which evaluates the percentage of words correctly recognized. Since texts embedded in videos (namely TextDatasetI) are mainly sentences and sequences of words, before evaluating the word recognition rate, an operation of alignment is applied to match recognized words with their corresponding words in the ground truth. Regarding natural "scene" texts of TextDatasetII, they consist only of single words, therefore no alignment is applied. Words correctly recognized are then determined and their recognition rate is evaluated (*cf.* Eq. 3.4).

$$WordRecoRate = \frac{\#words\ correctly\ recognized}{\#words\ to\ recognize} \tag{3.4}$$

3.4 Conclusion

Two main datasets were presented in this chapter: a "caption" text video dataset and a natural "scene" text dataset. These datasets include two types of multimedia documents (*i.e.*, videos and images) and concern two different text recognition problems, which are particularly interesting for the experimentation of our OCR systems.

In the next chapters, the approaches that we propose for text recognition in multimedia documents are described and their performance is evaluated on the presented datasets using the defined metrics: the character and the word recognition rates. Experiments are depicted and results are discussed.

Chapter 4

The segmentation-based approach

4.1 Introduction

This chapter presents our first contribution, *i.e.*, an approach designed to recognize texts in images and videos. The main idea of this approach consists in segmenting the text image into individual characters before recognizing them. In contrast with existing methods, this OCR technique first performs a nonlinear character segmentation taking into account the local morphology of text images. Moreover, the proposed OCR relies on a robust character recognizer based on a neural classification model and integrates some linguistic knowledge in order to improve its performance.

Fig. 4.1 depicts the outline of this approach that involves three processing steps. The next four sections (namely sections 4.2, 4.3, 4.4, and 4.5) describe these different steps and their interactions within the recognition scheme. The experiments achieved to evaluate our method and their results are presented and discussed in section 4.6. Finally, section 4.7 provides some conclusions and highlights the advantages and the limits of this OCR approach.

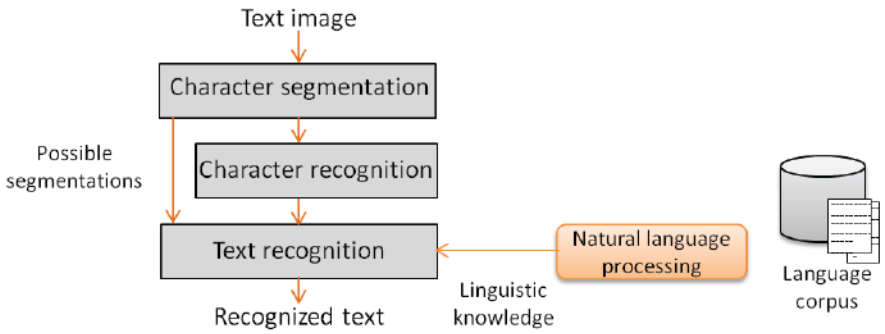

Figure 4.1: The proposed segmentation-based OCR scheme.

4.2 Character segmentation

The first step of the proposed OCR consists in splitting the text image into sub-images. The aim of this character segmentation step is to obtain one sub-image per individual character, in order to enable a recognition. In this context, the segmentation is crucial for the character recognition process: any error will directly reduce the recognition accuracy of the OCR system. On the one hand, an over-segmentation leads to individual characters segmented into two or more parts, hence the recognition of the characters becomes impossible. On the other hand, an under-segmentation induces two or more characters fused in an individual sub-image, which don't permit recognition.

Considering these issues, we propose a robust segmentation method that permits to separate characters depending on their local morphologies. To do so, our method first distinguishes the text from the background taking into account the intensities of pixels and, in the case of video texts, the temporal redundancy. This processing produces a fuzzy map that is then used to segment the characters with a shortest path algorithm. Fig. 4.2 depicts the outline of the proposed character segmentation.

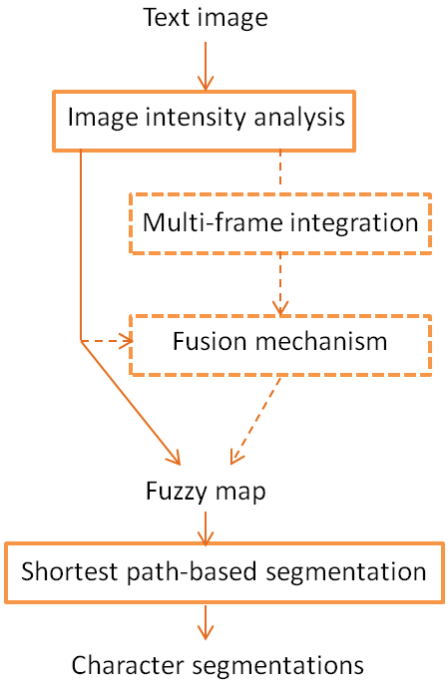

Figure 4.2: Steps of the proposed character segmentation: parts with dotted lines concern only video texts.

4.2.1 Statistical intensity analysis

In order to find reliable segmentations between characters, a preliminary treatment that distinguishes the background from the text is required. An image intensity analysis is hence performed. Assuming that pixels of a text image are of two classes—"text" and "background"—and that intensities of both classes are governed by Gaussian distributions, a Gaussian mixture model can be fit to the image histogram. Thus the distribution function related to the image intensity can be expressed as follows:

$$P_{Image}(I) = \alpha_{Text} \times P_{Text}(I, \mu_{Text}, \sigma_{Text}) + \alpha_{Back} \times P_{Back}(I, \mu_{Back}, \sigma_{Back}) \quad (4.1)$$

where P_{image} is the distribution of the text image, I is the intensity, P_{Text} and P_{Back} are the Gaussian distributions of class "text" and class "background", μ_{Text}, σ_{Text}, μ_{Back} and σ_{Back} are the distributions parameters (mean and standard deviation respectively), and α_{Text} and α_{Back} are the distributions weights.

Using the Expectation-Maximization algorithm (EM) [DLR+77], the estimation of the distributions parameters can be obtained by formalizing the problem as maximizing the likelihood between a set of observations—namely the image histogram—and a Gaussian mixture model. Once the EM algorithm converges, estimated parameters are then used to generate a fuzzy map indicating, for each pixel, its membership degree to the class "text". To do so, a model is applied such as the membership value is equal to 0 if $I < \mu_1$, 1 if $I > \mu_2$ and varies linearly between these bounds, with I the pixel intensity, μ_1 the mean of the dark distribution and μ_2 the mean of light distribution.

Note that the polarization of the text image (dark text on a light background or light text on a dark background) is determined by a simple processing that consists in comparing the intensities of pixels of the two top and the two bottom rows of the image to the intensities of the four middle rows. In the case of dark text on light background, the fuzzy map is inverted in order to get a map where pixels are characterized by their membership degree to the class "text". Fig. 4.3 shows an example of such a fuzzy map.

In addition to the image intensity analysis, in the case of video data, we decide to benefit from the temporal redundancy of texts by introducing a multi-frame integration process. Indeed, texts generally keep the same visual attributes (color, size, font, etc.) during their appearance while backgrounds can change. Our idea consists in analyzing the temporal variability and, when it is possible, in identifying the background regions by their observed variability. In subsection 2.2.3, we have distinguished two main multi-frame integration approaches proposed in the literature: those based on the averaging technique and those based on the minimum/maximum integration. Since the first category of approaches is sensitive to noise and complex background, we define a method based on the minimum/maximum integration that characterizes the image pixels by their temporal standard deviation. Typically, each pixel of the detected text is treated separately (without considering its neighborhood) to analyze its different values while the text is present and hence evaluate its temporal

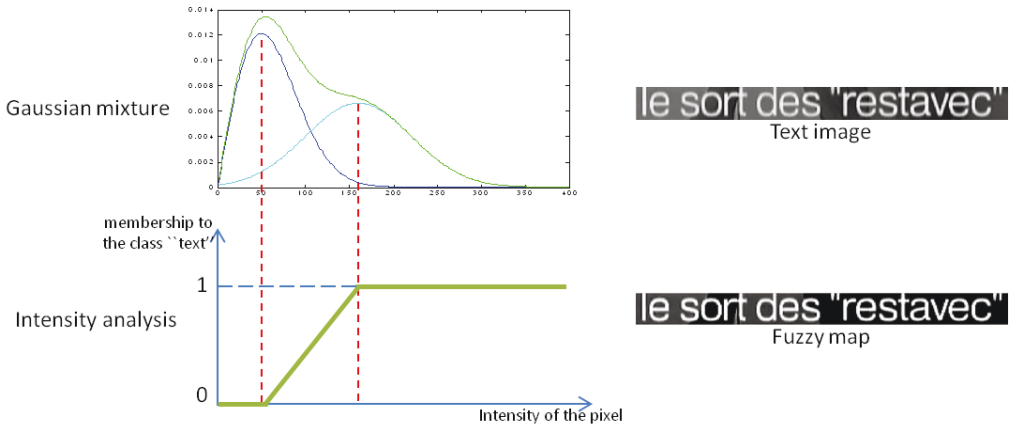

Figure 4.3: Image intensity analysis.

standard deviation. Using these results, another fuzzy map indicating for each pixel its membership degree to the class "background" is generated. Fig. 4.4 illustrates an example of such a fuzzy map.

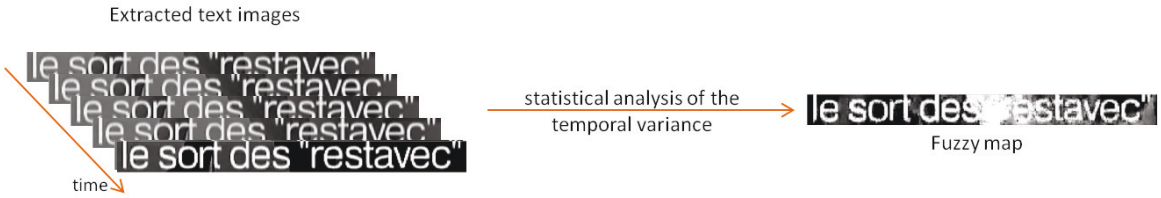

Figure 4.4: Multi-frame integration: obtained fuzzy map; dark pixels are those presenting high variations while bright ones are those with low variations.

Since intensity distributions and temporal variations are two independent sources of information, we propose to combine the two fuzzy maps described above, in order to obtain a more accurate membership one. A fusion system, with an adaptive behavior depending on the values to combine, is therefore required. According to the definitions in [Yag91], the chosen operator should be conjunctive (with a severe behavior) if both values are low, disjunctive (with an indulgent behavior) if both values are high and it should depend only on intensity distribution analysis if the temporal variation is low. The operator expressed by Eq. 4.2 satisfies these conditions:

$$f(x,y) = \begin{cases} x & if\ y \leq th \\ \sigma(x,y) = \frac{g(x,y)}{g(x,y)+g(1-x,1-y)} & otherwise \end{cases} \tag{4.2}$$

where x refers to the intensity analysis result, y refers to the temporal variation result and th is a threshold determined empirically. $\sigma(x,y)$ is the associative symmetric sum,

and $g(x, y)$ is a positive increasing function (in our application, $g(x, y)$ is defined as $g(x, y) = x \times y$).

Three examples of generated fuzzy maps are presented in Fig. 4.5. In example (A), the temporal variation allows to identify only some regions of the background. Example (C) illustrates another case in which the intensity analysis is shown to be insufficient to distinguish both classes (this can be explained by a complex background with some pixel intensities too close to the "text" mean). In both cases (A) and (C), the combination of the intensity analysis result and the temporal variation result permits to obtain a fuzzy map where the "background" and the "text" are well separated. Example (B) shows how the combination operation also permits to remove encoding artifacts observed in the temporal variation result.

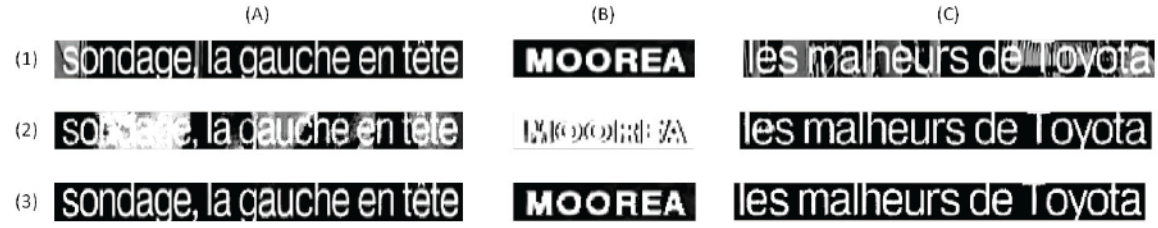

Figure 4.5: Fuzzy maps combination: (A), (B) and (C) are three examples of texts detected in videos; (1) the intensity analysis result, (2) the temporal variation result (white pixels are those with low temporal variation), (3) the fuzzy map obtained after combining (1) and (2).

As mentioned above, the fuzzy map that has been produced is then used for the character segmentation step. Note that the color images, which contain further important information for the character recognition task, are however considered for the rest of the OCR steps.

4.2.2 Shortest path-based segmentation

Our approach aims at finding nonlinear borders that separate characters while being well-suited to their different morphologies. The segmentations, which take into account the size and the shape of the text, permit to enhance the character recognition rates in the next step and thus improve the OCR performance, as we will show in section 4.6.

Inspired by [LLP02], where Lee et *al.* propose to segment texts in printed documents by using a projection profiles technique, a topographic feature analysis and a shortest path algorithm, we define the segmentation as a problem of shortest vertical path computation in the text image. Considering the fuzzy map generated above (*cf.* subsection 4.2.1) as a grid of vertices (pixels), each segmentation border is computed as the shortest vertical path containing pixels of low probabilities (*i.e.*, membership degrees) to belong to the class "text". Typically, nonlinear segmentation borders are

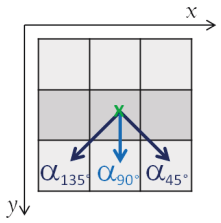

Figure 4.6: The shortest path computation: the three allowed directions and their respective weights.

computed as paths connecting pixels from the top (the first row) to the bottom (the last row) of the image without crossing any pixel belonging to the class "text" (*i.e.*, pixels having a membership degree over a fixed threshold). Paths containing pixels with a probability value above the threshold (*i.e.*, pixels with a strong probability to belong to the "text") are discarded even if they have the lowest accumulated probabilities.

In our shortest path algorithm, from each pixel, only three directions are allowed: 45 °, 90 °, and 135 ° with respect to the horizontal axis. For each direction i °, a weight $\alpha_{i\circ}$ is assigned empirically. Note that $\alpha_{45\circ} = \alpha_{135\circ}$, while $\alpha_{90\circ} \neq \alpha_{45\circ}$ (*cf.* Fig. 4.6). Typically, assuming that each pixel in the fuzzy map is identified by its coordinates (x, y) and characterized by its membership degree I, the shortest path $Path(S)$ which starts from the pixel S of the top of the image is computed with the following formula:

$$Path(S) = \left\{ A_i \mid A_{i+1} = \underset{B}{\operatorname{argmax}} \left(\alpha_{(A_i,B)} \cdot I_B\right), \forall\, 0 < i < n \right\}_{i \in \{0,\ldots,n\}} \tag{4.3}$$

where A_0 is S the first pixel in the path $Path(S)$, B is a pixel of the image, neighbor of A_i, that satisfies the two conditions $|x_B - x_{A_i}| < 2$ and $y_B = y_{A_i} + 1$, $\alpha_{(A_i,B)}$ is equal to $\alpha_{90\circ}$ if $x_B = x_{A_i}$ and to $\alpha_{45\circ}$ otherwise, and n is the number of pixels in $Path(S)$ and is equal to the height of the text image.

In order not to compute shortest paths from every pixel in the top line of the image, and to avoid over-segmentation, a criterion of minimal distance (proportional to the height of the text image) between the first vertices of two successive paths is used.

The resulting segmentation borders are characterized by the value of their highest pixel probability (*i.e.*, the pixel with the highest probability of belonging to the class "text"). This value is called the score of the path. Two categories of segmentation borders are distinguished depending on their scores:

- "Accurate" borders which correspond to paths with low scores (*i.e.*, under a threshold set empirically). These paths are considered as corresponding to correct separations between two characters.

Figure 4.7: An example of nonlinear segmentations: "accurate" ones are shown in green and "risky" ones are shown in red.

- "Risky" borders which correspond to paths with higher scores. These paths will be questioned later relying on further information in order to remove ambiguities: character recognition results (*cf.* subsection 4.3) and linguistic knowledge (*cf.* subsection 4.5).

Fig. 4.7 illustrates an example of the obtained nonlinear segmentations. As shown in this figure, "risky" segmentations (drawn in red) can correspond either to an over-segmentation (*e.g.*, the 'r' or the 'd') or to a correct separation between two touching characters on a complex background (*e.g.*, "rt").

According to the survey of character segmentation techniques presented by Casey *et al.* [CL02] (see section 2.3), our method can be considered as a hybrid method which takes advantages of both "dissection" and "recognition-based" techniques. Indeed, "accurate" segmentations are obtained by an intelligent process including an analysis of the image but without any symbol classification. Thus, they can be considered as deriving from "dissection" techniques. In contrast, "risky" segmentations that will be discussed in accordance with recognition results can be considered as derived from "recognition-based" techniques.

4.3 Character recognition

Once segmentation borders are computed, character images are generated by extracting the segmented area and filling out the remaining pixels with the mean background value. Obtained characters have then to be recognized. Among the state-of-the-art approaches dedicated to the problem of character recognition in color and in gray scale images, the dominant methodology consists first in binarizing the images and then extracting visual features to recognize characters. The main drawback of this kind of methods is that binarization may fail when the background is complex, leading to poor recognition rates. Unlike these techniques, we propose to rely on a neural classification approach, based on Convolutional Neural Networks, able to learn to extract appropriate descriptors and to recognize the character at the same time, without any binarization step.

Before describing our character recognizer, a brief introduction to Convolutional neural networks is presented.

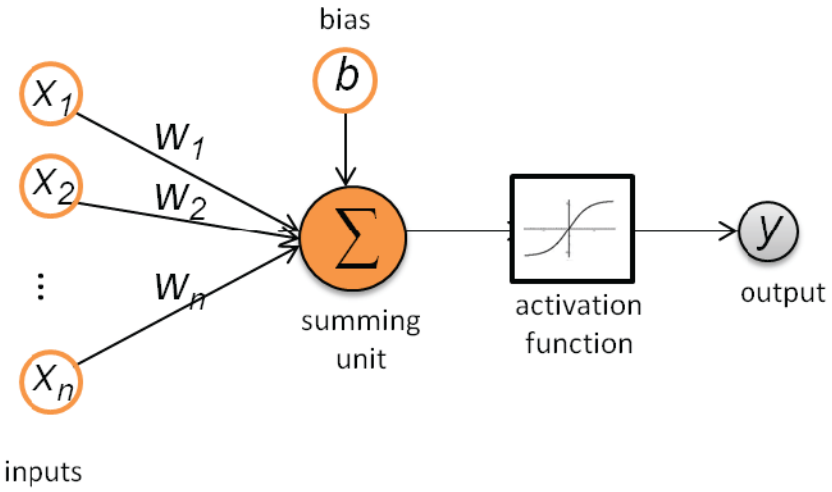

Figure 4.8: The perceptron's structure.

4.3.1 Convolutional neural networks

Convolutional neural network is a special form of Artificial Neural Networks. This section starts by introducing Artificial Neural Networks, then presents convolutional networks and details their use.

Artificial Neural Networks (ANNs) correspond to a machine learning technique that has been inspired by the human brain and the functioning of its biological neurons. Similar to a human brain, an ANN is a trainable model that consists in a set of units, called artificial neurons, connected together to perform a specific task.

The basic neural unit, illustrated in Fig. 4.8, was proposed by Rosenblatt [Ros58] and named perceptron. It has n inputs $\{x_1, x_2, ..., x_n\}$ and one output y calculated as follows:

$$y = f(\sum_{i=1}^{n} x_i.w_i + b) \tag{4.4}$$

where w_i is the weight of input x_i, b is a bias and f is a differentiable function called the activation function. Among the most common activation functions, the linear function, the sigmoid and the hyperbolic tangent are distinguished.

The Multi-Layer Perceptron (MLP) [RHW86], formed by a group of interconnected perceptrons, is the most common ANN. As shown in Fig. 4.9, a MLP consists in an input layer (containing the vector of inputs), one (or more) hidden layer(s) containing perceptrons connected to the input or the previous hidden layer and an output layer containing perceptrons connected to the last hidden layer.

To perform a given task, an ANN requires a learning phase in order to adjust its parameters (namely the weights and the biases). Using a training set containing pairs of inputs and their corresponding outputs, called targets, a training algorithm is applied to obtain the optimal parameters that enable the network to model the

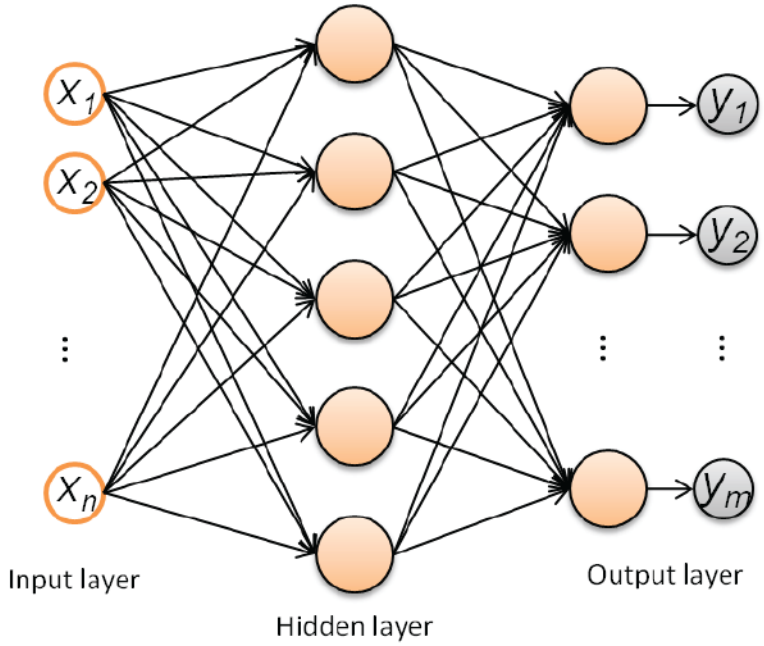

Figure 4.9: An example of a MLP's architecture.

dependency between the inputs and their targets. Among the training algorithms, the most popular is the back-propagation algorithm [Wer90] that defines an error function equal to the Mean Square Error (MSE), and uses a gradient descent technique to minimize this error and hence estimate the weights and biases. The training phase is done by alternating two steps: the forward pass which consists in evaluating the outputs of the given inputs and the backward pass which updates the parameters propagating the error relative to targets back through the network.

Though MLPs have shown a great ability to perform complex tasks, their application remains limited to vectors of inputs. In particular, they are not adapted to the case of image data since they are not able to take in consideration any spatial information. To overcome this difficulty, some works have proposed to use hand-crafted feature extractors that generate vectors of features. These vectors can then feed a MLP network. However, the major issue of this solution is to design the appropriate feature extractor able to produce the relevant information from the input image.

Convolutional Neural Networks, hereafter ConvNets, are a special form of ANN introduced by LeCun *et al.* [LB95], in order to tackle this problem and extend the use of MLPs to the case of images. ConvNets are biologically-inspired architectures by the mammalian visual cortex. Their basic idea is to define a model able to recognize visual patterns directly from the image pixels without any preprocessing step. To do so, ConvNets rely on three key ideas:

- the concept of local receptive fields that enables the network to extract elemen-

tary visual features (such as edges, corners, etc.).

- the weights sharing technique that reduces the number of parameters and hence improves the generalization ability of the network, also reducing the computational cost (since the number of trainable weights is reduced).

- the sub-sampling in the spatial domain which permits to reduce the sensitivity to some geometrical transformations such as shifts. It also reduces the computational cost.

Typically, a ConvNet is designed to take as input an image, whose values are normalized between -1 and $+1$, connected to a cascade of convolutional and sub-sampling layers, followed by a classical MLP. Each layer consists of a set of maps resulting of either a convolutional or a sub-sampling or a neuron operation. An example of such an architecture is given in Fig. 4.10. During the training phase, the ConvNet jointly learns to extract the appropriate features descriptors (convolution and sub-sampling layers) and to perform the desired classification (MLP layers).

Due to its robustness to noise, deformations, translations and scale variations, this model has proved a great ability to deal with a large number of extremely variable patterns. In the literature, ConvNets were tested on several pattern recognition tasks [LKF10] including face detection [GD04], document analysis [SSJ03] and character recognition in handwritten documents [LBBH98] and in color images [SG07a].

4.3.2 Network architecture and training

Because of its ability to deal with large image datasets and its great robustness to deformation and noise, we choose a ConvNet as a classification model for our character recognition problem (a comparison with other classification methods will be done in subsection 4.6.1).

For each character dataset (CharDatasetI and CharDatasetII described in chapter 3 respectively in subsections 3.1.2 and 3.2.2), several network architectures were tested in order to obtain the optimal ConvNet that avoids overfitting problems and increases the generalization ability of the system. At the end of these tests, a heterogeneous ConvNet architecture, called CRConvNet for Character Recognizer ConvNet, was chosen for each dataset.

Each of the CRConvNets takes as input a color character image mapped into three maps of size $T \times T$ pixels (T is fixed to 36 and 48 respectively in the case of CharDatasetI and CharDatasetII, these values are determined depending on the average size of image examples of each dataset). Each map, whose values are normalized between -1 and 1, represents a channel (R, G or B). The hidden layers of the CRConvNets consist of five layers. The two first ones, a convolutional (with kernel masks of size 5×5) and a sub-sampling layer, can be interpreted as feature extractors; *i.e.*, the first one analyzes the character image and extracts its low visual features

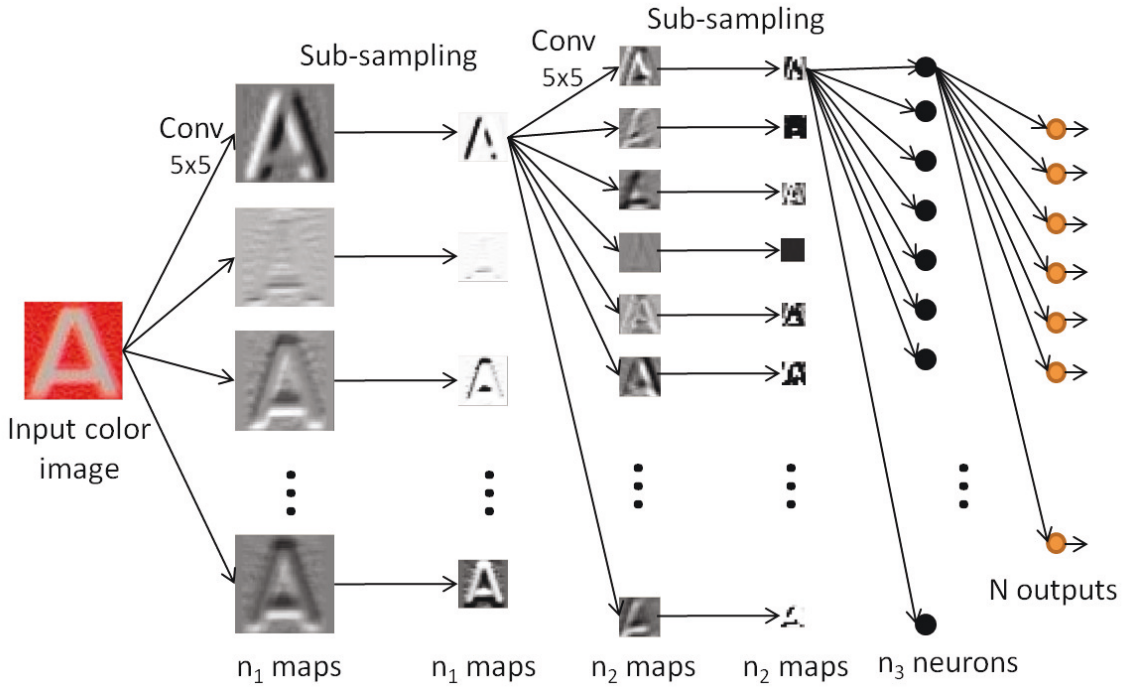

Figure 4.10: The CRConvNet architecture; Conv means convolution.

(like corners or edges), while the sub-sampling step reduces the sensitivity to affine transformations. Each of these layers generates n_1 feature maps connected to the following layers. Once extracted, visual features can be combined taking into account their spatial relationships. This step is ensured by the two next convolutional (with kernel masks of size 5×5) and sub-sampling layers. n_2 new feature maps are then generated and connected to a classical 2-layer MLP that enables the classification decision. CRConvNets return vectors of outputs of size N, corresponding to the number of considered character classes (namely 41 and 36 respectively for CharDatasetI and CharDatasetII). The neurons of the output layer use the hyperbolic tangent, defined by Lecun *et al.*, as an activation function and return values, between -1.7 and 1.7, that encode scores of the input image to belong to the given character classes. The recognized character is finally determined as the class obtaining the highest output value. Fig. 4.10 shows a CRConvNet and illustrates its architecture.

The designed networks were trained to learn to extract appropriate visual features and classify images of single characters. The parameters of the ConvNets (convolution coefficients, biases and connection weights) are adjusted using the error back-propagation algorithm. Classically, each image dataset is divided into three sets: a training and a validation sets used for the training and a test set used to evaluate the performance of the trained network. At each iteration, a color image of the training set is presented to the network and a mean square error term is computed as:

$$MSE = \frac{1}{N} \cdot \sum_{i=1}^{N} (o_i - d_i)^2 \tag{4.5}$$

where o_i is the output value of the neuron i, and d_i is its desired value (target), which is set to 1 if i corresponds to the character's class and to -1 for other neurons. The error is then back-propagated in the network layers and the parameters are updated. At each epoch[1], a classification error rate is evaluated on the training and the validation sets in order to control the generalization and the overfitting. Once the training phase is finished, the classification performance of the obtained ConvNet is evaluated on the test set (not used in the training phase).

Fig. 4.11 shows examples of character images recognized with the CRConvNet trained on CharDatasetI. In general, the recognized class is often obtained with a positive score close to 1, while the score of the remaining classes are negative and close to -1. However, for the character "t" the recognized class is "t" with a score of 0.08 and the second best response of the CRConvNet is "f" with a score of -0.057. The too close scores illustrate a strong ambiguity between these two classes, and the CRConvNet encounters a difficulty to distinguish between them.

In our text recognition scheme, the trained CRConvNet is applied to all segmented characters of the text image. When between two successive "accurate" segmentations (see section 4.2.2), "risky" ones (red separations in Fig. 4.12 - (A)) are observed, the CRConvNet is applied to all possible characters when keeping and removing each risky segmentation. Fig. 4.12 - (A) and (B) illustrate the segmentation step results and all the candidate characters on which the CRConvNet is applied.

4.4 Text recognition

Text images are thus segmented into individual characters recognized using the CRConvNet. Focus can now be put on the recognition of the whole texts (single words and sentences) detected in videos or extracted from scene images.

Since text images are segmented into separated characters, we intuitively combine individual character recognition results and recognize texts as the sequences of recognized characters. In subsection 4.2.2, two categories of segmentation borders were distinguished: "accurate" and "risky" ones. In our text recognition process, characters located between two "accurate" segmentations (green separations in Fig. 4.12 - (A)) are recognized and considered as letters of the text. However, characters segmented with one or two "risky" segmentations are analyzed to decide which ones correspond to correct characters and have to be added to the text. To do so, all the responses of the CRConvNet applied on each character candidate are first considered (all arcs in Fig. 4.12 - (B)). Different propositions of words are then tested and evaluated by

[1] An epoch is a set of iterations corresponding to the presentation of all the training images to the network.

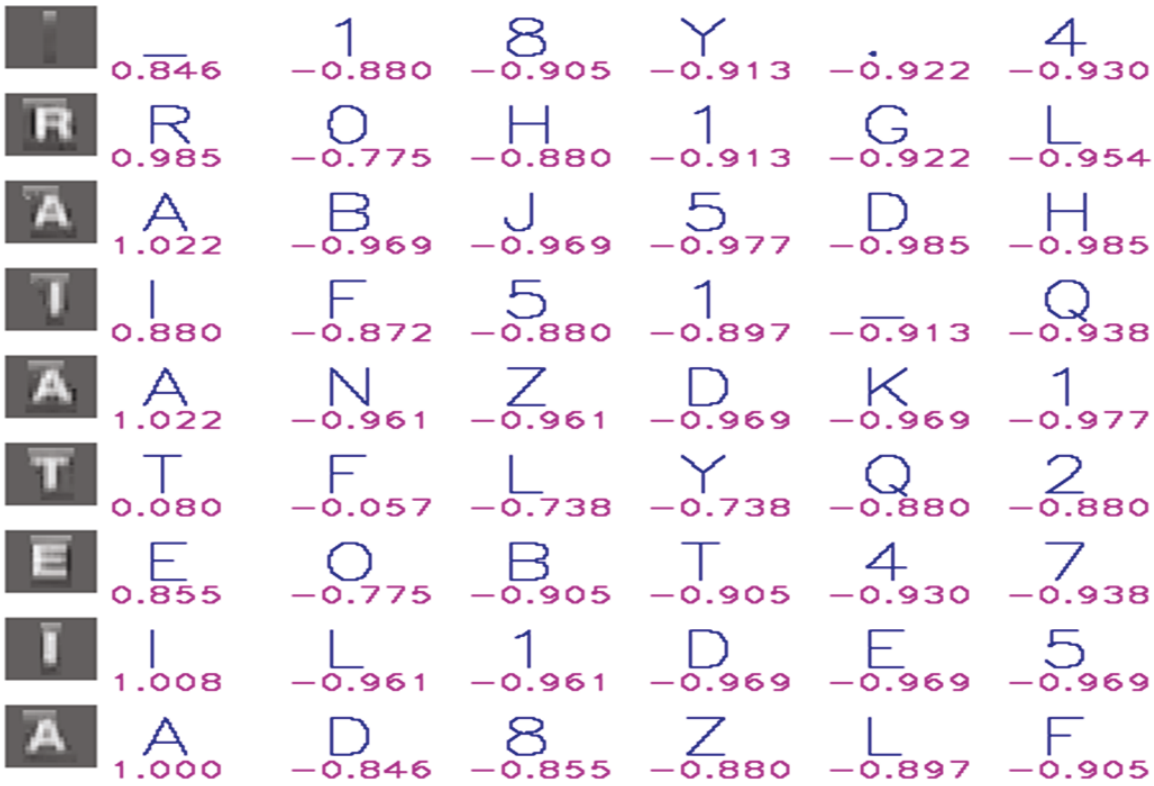

Figure 4.11: Examples of recognized characters with the CRConvNet: each line illustrates an example of a character image and its corresponding 6 best results (character class in blue and score in pink) obtained with the CRConvNet, the class "_" represents the class "space".

scores calculated depending only on the CRConvNet's outputs. The word configuration that obtains the highest score is selected. At this stage of the processing, for each possible segmentation, only the best response (*i.e.*, the class obtaining the highest probability) of the CRConvNet is considered while the rest of the responses is ignored.

As shown in Fig. 4.12, even though errors related to "risky segmentations" are reduced, confusions between similar characters are still present (such as the "v" recognized as a "y"). In section 4.5, we show how to introduce linguistic knowledge able to drive the recognition scheme and to tackle these character confusions.

4.5 Integration of linguistic knowledge

To reduce the remaining errors still produced by our recognizer, we propose to benefit from the lexical context and incorporate some linguistic knowledge to drive the whole

Figure 4.12: An example of recognized text: (A) the segmented text image: green and red separations represent respectively "accurate" and "risky" segmentations, (B) characters candidates: green arcs illustrate characters located between successive "accurate" segmentations and red dotted arcs represent different possible configurations of characters related to "risky" segmentations, and (C) the recognized text.

recognition scheme. Using this further information, we aim at tackling remaining difficulties related to character recognition (due to confusion between similar characters, and to the local character by character recognition) and at removing ambiguities related to risky segmentations and to low quality images.

In this context, statistical language models seem to be well adapted to our recognition task enabling the incorporation of linguistic properties in recognition schemes. This section first introduces the fundamentals of the language model (namely a n-gram one) and then details its integration in our text recognition scheme.

4.5.1 The n-gram language model

Widely used for speech transcription and natural language processing applications like machine translation, n-gram models have demonstrated a strong ability to improve recognition performance by considering the lexical context [BBdSM02]. They are the most common technique of statistical language modeling [MS99] and aim at detecting the regularities of a given language. Relying on a statistical analysis of a large corpus, n-gram models permit to evaluate the a priori probability of a given succession of N items (which can be characters, phonemes or words). Hence, they allow to predict the following item to be recognized given the items that have just been analyzed.

Assuming that the goal is to find the sequence of items $\hat{X}$ which maximizes the probability $p(X|signal)$, where X is a sequence of items and $signal$ is the given input signal (speech, text, image, etc.), the Maximum A Posteriori (MAP) approach can be applied to express the problem as follows:

$$\hat{X} = \underset{X}{\operatorname{argmax}}\, p(X|signal) = \underset{X}{\operatorname{argmax}}\, p(signal|X) \cdot p(X) \tag{4.6}$$

where $p(signal|X)$ is the a posteriori probability of the $signal$ given the recognized item sequence X, and $p(X)$ is the a priori probability of X. The first term $p(signal|X)$

is derived from the signal analysis, while the second one $p(X)$ is obtained from the language model and can be formulated as follows:

$$p(X) = \prod_i p(x_i|\phi(h(i))) = \prod_i p(x_i|x_1x_2...x_{i-1}) \tag{4.7}$$

where $\phi(h(i))$ is the context of the item x_i and corresponds to the sequence of items $x_1x_2...x_{i-1}$ that precedes x_i. Since $\phi(h(i))$ can contain an important number of items, an infinite number of different contexts $\phi(h(i))$ is possible and hence the estimation of the probabilities is extremely hard. The n-gram model provides a solution to this problem by assuming that an item only depends on its $n-1$ predecessors. Thus, the context history $\phi(h(i))$ can be reduced to the $n-1$ items preceding x_i and Eq. 4.7 can be rewritten into:

$$p(X) = \prod_i p(x_i|\phi(h_n(i))) = \prod_i p(x_i|x_{i-n}x_{i-n+1}...x_{i-1}) \tag{4.8}$$

Generally, the probabilities $p(x_i|x_{i-n}x_{i-n+1}...x_{i-1})$ can be estimated via the statistical analysis of a corpus. Nevertheless, when some possible n-grams does not occur in the corpus, the statistical analysis is not able to evaluate their probabilities to be observed and affects zero, while this is inaccurate. To handle this problem, several techniques, called smoothing [CG96], were proposed to adjust the estimation of the language model, and hence to produce more accurate probabilities for missing n-grams.

In practice, the order n of a n-gram model is usually equal to 2 (a bi-gram model), 3 (a tri-gram model) or 4 (a quadri-gram model), rarely beyond because such model requires a huge volume of data, necessary to reliably estimate probabilities of large context histories.

4.5.2 Integration of the language model

For our recognition problem, since single words or short sentences are considered, a character n-gram model is chosen to estimate the probabilities of sequences of letters in a given language. These probabilities are then integrated into our recognition framework to manage relationships between successive characters. Hence, when evaluating words scores (defined in section 4.4) these joint probabilities of character sequences are introduced to adjust transitions between characters and weight the different word propositions.

Typically, Eq. 4.6 becomes:

$$\hat{C} = \underset{C}{\operatorname{argmax}}\, p(\textit{signal}|C) \cdot p(C) \tag{4.9}$$

where $\hat{C}$ is the sequence of characters present in the image, $p(\textit{signal}|C)$ is the a posteriori probability of *signal* (namely the image) given the character sequence C.

This probability is computed from the character recognition results (*i.e.*, the outputs of the CRConvNet) using a classical *softmax* function, as follows:

$$p(signal|C) = \prod_i p(s_i|c_i) = \prod_i \frac{\exp(output(s_i|c_i))}{\sum_j \exp(output(s_j|c_j))} \tag{4.10}$$

where c_i, s_i and $output(s_i|c_i)$ are respectively the i^{th} character of sequence C, its image and its corresponding CRConvNet output (see section 4.3.2).

$P(C)$ is obtained from the character n-gram model (as in Eq. 4.8) and is expressed as follows:

$$p(C) = \prod_i p(c_i|\phi(h_n(i))) = \prod_i p(c_i|c_{i-n}c_{i-n+1}...c_{i-1}) \tag{4.11}$$

where $p(c_i|c_{i-n}c_{i-n+1}...c_{i-1})$ is the probability to observe c_i given its context history $c_{i-n}c_{i-n+1}...c_{i-1}$. In our experiments, using the *SRILM* toolkit [Sto02], two n-gram language models—one for the French language used for DatasetI and one for the English language for DatasetII—were trained to learn these joint probabilities of character sequences on two corpora of about respectively $10,000$ French words and $11,000$ English words. For our language training, we have tested several smoothing techniques (namely absolute discounting backing-off, Kartz's method [Kat87], Kneser-Ney's method [KN95], and Chen and Goodman's method [CG96]); best performance was obtained with Chen and Goodman's technique.

Because probabilities are low (between 0 and 1), their logarithm (between $-\infty$ and 0) is preferred, and two coefficients γ and δ are introduced as follows:

$$\hat{C} = \arg max \sum_i (log(p(s_i|c_i)) + \gamma \cdot log(p(c_i|\phi(h_n(i)))) + \delta) \tag{4.12}$$

where $\phi(h_n(i))$ corresponds to the sequence $c_{i-n}c_{i-n+1}...c_{i-1}$. γ, called the Grammar Scale Factor, encodes the weight of the language model and serves to balance the influence of the linguistic knowledge in our OCR system. The parameter δ was incorporated to compensate the over- and sub-segmentations by controlling the lengths of word candidates.

As shown in Fig. 4.13, for each word, a recognition graph is built in order to determine the most probable word hypotheses that correspond to the image. For each segmented region, the five best recognized characters are considered with their CRConvNet outputs. Each segmentation hypothesis is represented by a node where a set of optimal character sequences is generated and characterized by a score that combines recognition results, language model probabilities and segmentation hypotheses. Note that for each "risky" segmentation two paths in the graph are investigated, keeping and removing this segmentation. The best word candidates are determined using the Viterbi algorithm.

For TextDatasetI, since text images can contain sentences or sequences of words, texts are first divided into single words using the class space (presented in chapter 3),

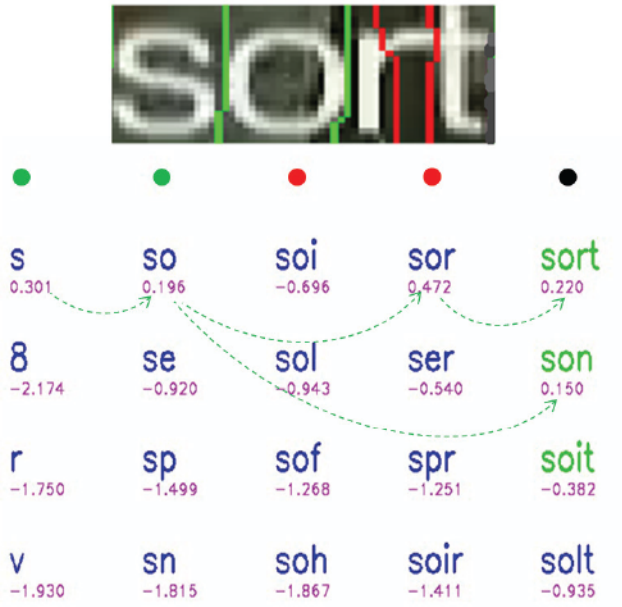

Figure 4.13: An example of recognition graph: dots correspond to segmentation borders, arcs illustrate some transitions between sequences of characters, values under each sequence of characters are their scores computed with Eq. 4.12, and words in green are words in the dictionary.

then each word is treated separately. For this dataset, word propositions are finally checked against a French dictionary, containing about $400,000$ words including named entities, and the recognized word is identified as the word belonging to the dictionary and having the best score. If no proposition is found in the dictionary, the word with the highest score is chosen. Notice that the use of a dictionary is not integrated for TextDatasetII because this scene image dataset contains an important number of texts that do not belong to the dictionary (such as acronyms, etc.).

Fig. 4.13 illustrates an example of recognized words and their corresponding transitions.

4.6 Experimental results

This section reports several experimentations carried out on DatasetI and DatasetII. First, an evaluation of our neural classification approach for the character recognition task is presented, and a comparison with a SVM-based method is provided. The complete recognition scheme for texts (words and sequences of words) is then tested and evaluated on the "caption" and "scene" text datasets. The particular impact of the language model is finally investigated.

Table 4.1: Performance of different ConvNet architectures tested on CharDatasetI (evaluated on the test set): n_1, n_2 and n_3 corresponds to the numbers of the maps and the neurons of the network (see Fig. 4.10) and RR means Recognition Rate.

n_1	n_2	n_3	**Character RR**
10	15	60	87.17%
12	25	80	98.04%
15	30	120	89.96%

Table 4.2: Classification performance of the CRConvNet (evaluated on the test sets): RR means Recognition Rate.

	Character RR	
	CharDatasetI	CharDatasetII
CRConvNet	98.04%	85.13%

4.6.1 Performance of the proposed character recognizers

The training and evaluation of the CRConvNets are performed on CharDatasetI and CharDatasetII. In both experiments, the datasets are divided randomly into three sets: a training set containing 80% of the images, a validation set and a test set, each containing 10% of the images. The first two sets are used to train the CRConvNets (as explained in subsection 4.3.2, two sets are necessary to control the training stage and to avoid overfitting) while the last one serves to evaluate the recognition performance.

In our experiments, several architectures of CRConvNets were tested on the same sets in order to obtain the optimal network that avoids both underfitting (when the network parameters are not sufficient to learn to classify all character classes) and overfitting (when the network starts to memorize images of the training set instead of learning to generalize). We report in Table 4.1 some results obtained on CharacterDatasetI with different architectures, varying the number of maps and neurons in the network.

Table 4.2 shows the classification results of the optimal configuration for each character dataset. These results are obtained after the training phase when evaluating the CRConvNets on the test set (not used in the training phase). Performance of both networks exceeds 85% of character recognition, a rate which allows to efficiently integrate our recognizers in the complete recognition scheme. A 13% difference between the results obtained on CharDatasetI and CharDatasetII can be noticed. This difference can be explained by the high variability of "scene" texts characters compared to "caption" texts characters (see Fig. 4.14).

We have also compared ConvNets and SVM classification performance for the character recognition task. Dorai *et al.* in [DAS01] have tested the ability of SVM models to recognize characters embedded in videos, but evaluated their performance on a dataset different from ours. Hence, to provide a meaningful comparison, both

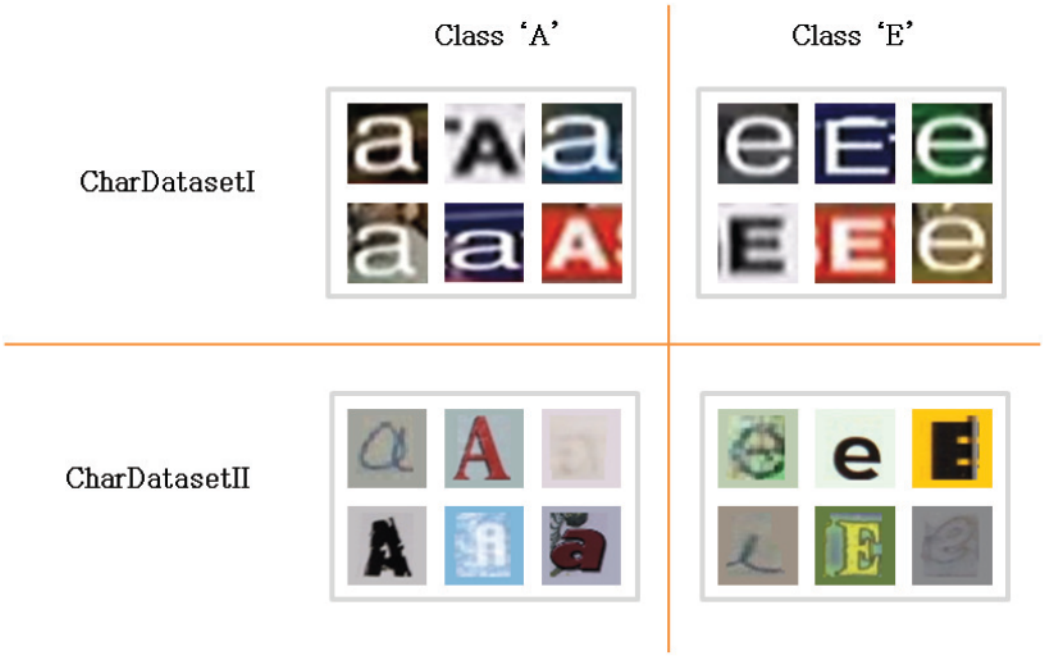

Figure 4.14: Examples of characters of CharDatasetI and CharDatasetII.

classification approaches were tested on the same databases of images, CharDatasetI and CharDatasetII, and under the same conditions (without extracting any features, images are considered as input vectors of each model).

Our SVM implementation was based on the software package *LIBSVM* [CL11], which was adapted to our datasets, to take the images as input. Using the RBF (Radial Basis Function)[2] kernel and a *one-vs-all* protocol, several configurations of the SVM were experimented, with different values for the parameters[3] C and γ. Table 4.3 depicts experimented SVM models and provides their results.

Table 4.3: Recognition rates of SVMs on a dataset of single characters: RR means Recognition Rate, C the penalty term, and SV Support Vectors.

SVM Id	**C**	**Number of SV**	**Character RR**	
			CharDatasetI	CharDatasetII
SVM_1	1	9,215	75.46%	67.60%
SVM_2	2	8,544	81.18%	72.30%
SVM_3	3	8,091	80.65%	74.94%

We can see that the recognition rates of the CRConvNets are higher than those

[2]The radial basis function is a particular function whose values are calculated depending on the distance to a center (namely the center of the kernel).

[3]The penalty parameter C of the SVM permits to control the trade off between allowing training errors and forcing rigid margins.

of SVMs. For CharDatasetI, the CRConvNet achieves a recognition rate of 98.04%, while the highest rate obtained with SVMs is 81.18% (SVM_2 model). Furthermore, in terms of trainable parameters, the CRConvNet requires 16,732 parameters, while SVM_2 needs 8,544 support vectors with 11,073,024 stored parameters. The same remarks hold when comparing the performance of the CRConvNet (about 85% of character recognition rate) and SVM models (less than 75%) on CharDataset II. Hence, we can conclude that, for this application (character recognition), ConvNets yield better classification performance with lower complexity than SVMs.

4.6.2 Performance of the segmentation-based OCR

Using the trained CRConvNets, we focus on the evaluation of the whole segmentation-based OCR tested both on TextDatasetI and TextDatasetII. Results are reported in table 4.4.

First, notice that character recognition rates are lower than those obtained in table 4.2 (from 98.04% to 95.33% in the case of DatasetI and from 85.13% to 65.33% in the case of DatasetII). This is simply due to the fact that CharDatasetI and CharDatasetII only contain perfectly segmented characters, whereas in the complete scheme some segmentation errors can appear.

Table 4.4: Recognition performance of the segmentation-based OCR: RR means Recognition Rate.

	TextDatasetI		**TextDatasetII**	
	Character RR	Word RR	Character RR	Word RR
segmentation-based OCR	95.33%	87.83%	65.33%	41.19%

Experimentations carried out on TextDatasetI (*cf.* table 4.4) show that the proposed OCR performs well on embedded texts with more than 95% of good character recognition rate. This demonstrates that when the character segmentation step works well (on text with small distortions like "caption" texts), it enhances the following phases and leads to better recognition performance. On the contrary, results obtained on natural "scene" texts (*i.e.*, TextDatasetII) show a large difference between both datasets. While, the proposed OCR achieves a word recognition rate of about 88% on TextDatasetI, it only obtains 41% on TextDatasetII. This result can be explained by three facts:

- The complexity of TextDatasetII which makes the recognition task more challenging.
- The fact that the character recognizer performs better for CharDatasetI (98% versus 85%).

- The drawbacks of the character segmentation step where any error directly induces a drop in the recognition accuracy. Particularly, in the case of natural "scene" text, images are usually affected by various distortions which make the segmentation very hard and thus lead to errors such as false segmentations considered as "accurate" ones that over-segment characters, or to confusing "risky" segmentations. Fig. 4.15 illustrates some examples of these errors.

Figure 4.15: Examples of segmentation errors produced on TextDatasetII: (A) the "scene" text image, (B) the obtained segmentations drawn on the fuzzy map (green and red separations represent "accurate" and "risky" segmentations).

Generally, the few remaining errors on TextDatasetI can be explained by some character confusions between visually similar characters and some character segmentation errors. Regarding the errors produced on TextDatasetII, the strong distortions of an important number of images and the small sizes of some of them reduce considerably the recognition performance while the crucial character segmentation step remains the major cause of errors.

We also compared the performance of our approach with state-of-the-art methods [SGD09, WB10] and commercial OCR engines (ABBYY FineReader OCR and Tesseract OCR). Results of these comparisons are reported in table 4.5. Since Saïdane *et al.* [SGD09] and Wang *et al.* [WB10] have designed their methods to recognize single words in natural scene images, comparisons with these previously published state-of-the-art methods are done only on the public database ICDAR 2003 (the "caption" texts video dataset contains mainly images with sentences). In these comparisons, two more experiments were performed on TextDatasetII, evaluating the word recognition rate as in [SGD09] and [WB10] (as a reminder experiments reported in table 4.4 have been done on the full TextDatasetII). The segmentation-based OCR is hence also evaluated on the 901 images selected in [SGD09] (Exp1) and on the 1,065 images selected in [WB10] using the same lexicon, created from all the words that appear in the test set as in [WB10] (Exp2). These different tests show that our segmentation-based OCR yields word recognition rates close to the ones obtained by other state-of-the-art methods. Indeed, our method and Saïdane *et al.*'s one obtain similar word recognition rates (with less than 1% of difference) and fail to recognize almost the same texts, where both segmentation methods produce false separations between characters. Re-

Table 4.5: Comparison of the proposed OCR systems to state-of-the-art methods and commercial OCR engines: CRR and WRR mean respectively character and word recognition rates (only WRRs are reported for experiments on TextDatasetII, since Character RRs are not provided for other methods).

OCR system	**TextDataset I**		**TextDataset II**	
			Exp1	**Exp2**
	CRR	**WRR**	**WRR**	**WRR**
Segmentation-based OCR	**95.33%**	**87.83%**	**53.28%**	**59.63%**
Saïdane *et al.* [SGD09]	-	-	54.13%	-
Wang *et al.* [WB10]	-	-	-	59.20%
ABBYY FineReader OCR	95.03%	87.70%	-	42.80%
Tesseract OCR	88.57%	70.01%	-	35.00%

garding Wang *et al.*'s method, even though it avoids any segmentation step, it obtains nearly the same performance as the one achieved with our OCR scheme.

Concerning commercial OCR systems, namely ABBYY FineReader and Tesseract, notice that, due to practical issues, these OCRs were not trained on the same datasets as our. As shown in table 4.5, results achieved on "caption" texts in TextDataset I show that the OCR that we propose outperforms the Tesseract OCR with more than +17% of words correctly recognized. Regarding ABBYY FineReader, this system obtains a word recognition rate of 87.70% (corresponding to 95.03% of character recognition rate) which is slightly lower than the one obtained by our segmentation-based OCR system (*i.e.*, 87.83% and 95.33% of word and character recognition rates). In the case of "scene" texts in TextDataset II, ABBYY FineReader and Tesseract evaluated by Wang *et al.* [WB10] obtain poor results with less than 45% of word recognition rate. Compared to this performance of commercial OCRs, our OCR achieves far better performance with more than +17% of word recognition rate. Hence, despite ABBYY FineReader achieves good results on "caption" texts, our OCR system proves its great ability to handle both "caption" and "scene" texts.

4.6.3 Contribution of the linguistic knowledge

In the previous subsection, we have given the performance of the full proposed OCR scheme and compare its performance to those of other existing methods and some commercial OCR engines. In this subsection, in order to better understand the capacities of our OCR scheme we focus on the evaluation of the contribution of the linguistic knowledge that has been incorporated in, *i.e.*, the language model and the use of the dictionary. We first present the determination of the optimal n-gram model (with its best parameters), then evaluate its influence on the recognition performance. The use of a dictionary is finally investigated highlighting its contribution.

In Eq. 4.12, two parameters γ and δ were introduced to control the relative impor-

tance of the language model and the segmentation hypothesis. Different experiments were performed to determine their optimal values. The best performance was obtained with $\gamma = 0.2$ and $\delta = 2$. Notice that when δ is low (under 2), recognition rates tend to decrease due to sub-segmentation, and when it is high (over 2), recognition rates also decrease because of over-segmentation.

In order to evaluate the influence of the parameter n (the order of the n-gram model, which corresponds to the length of the considered character context history), bi-gram (estimating about $1,350$ probabilities), tri-gram (estimating about $12,800$ probabilities) and quadri-gram (estimating about $200,000$ probabilities) models were experimented. Notice that these experiments where performed integrating the trained language models but no dictionary in order not to influence their contribution by another source of linguistic knowledge.

As shown in table 4.6, the best word recognition rate for both datasets is obtained with the tri-gram model. Actually, the poor context (only one predecessor) in the bi-gram model, even if it improves the recognition results, seems not to be sufficient to remove ambiguities related to the local character by character recognition. The quadri-gram model also permits to increase the performance of the baseline system (without any language model). However, it does not improve results compared to the tri-gram model which is less complex. For these reasons, the tri-gram model is chosen for the rest of the experiments.

Table 4.6: Evaluation of the influence of the order of the character n-gram model on the recognition performance: baseline system corresponds to our OCR without any linguistic knowledge.

n-gram	**TextDatasetI**		**TextDatasetII**	
	Character RR	Word RR	Character RR	Word RR
Baseline system	88.14%	63.04%	61.12%	34.75%
Bi-gram	91.83%	67.53%	62.58%	36.28%
Tri-gram	95.56%	85.80%	65.33%	41.19%
Quadri-gram	94.01%	80.64%	63.91%	38.46%

When incorporated in the OCR scheme (*cf.* table 4.6), the tri-gram model permits to improve the recognition performance on both TextDatasetI and TextDatasetII by respectively about 22% and 7% of word recognition rate. Fig. 4.16 presents some corrections obtained with this integration; some confusions between similar characters (*e.g.*, "i" and "j") are removed, and over-segmentations (*e.g.*, for "r") are eliminated. However, errors associated with "accurate segmentations" (*e.g.*, the space between "T" and "o" in the word "Toyota") remain difficult to correct and generally the system fails to find the right word.

Finally, we evaluated the impact of the use of a dictionary—another kind of linguistic knowledge—for TextDatasetI. Obviously, experiments demonstrated that our system can be improved when a dictionary is used (*cf.* table 4.7). They also show

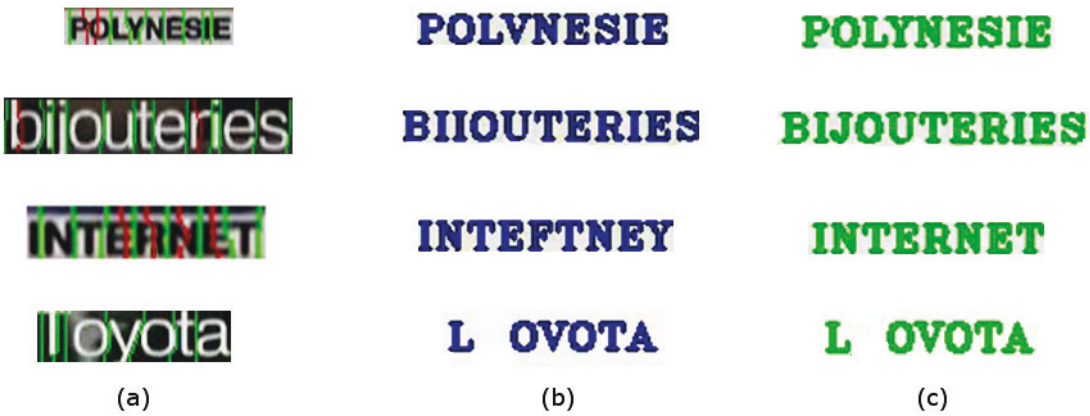

Figure 4.16: Examples of texts recognized by the segmentation-based OCR: (a) segmented images, (b) results before integrating the language model and (c) results after the integration of the language model (tri-gram).

that the combination of the two sources of linguistic knowledge—the language model and the dictionary—results in the best performance, yielding a character recognition rate of 95.33% and a word recognition rate of 87.83%. The slight drop in the character recognition rate (by about 0.23%) can be explained by the fact that some words, initially recognized correctly, but absent from the dictionary (like named entities), are finally replaced by other wrong but lexicalized propositions. Note that, for the intended applications, such as news indexing, we can expect to update a dictionary by automatically adding new named entities (such as names of persons, events, titles of films, etc.).

Table 4.7: Improving recognition performance by using a dictionary (LM means Language Model and Dic Dictionary).

Method	**TextDatasetI**	
	Character RR	Word RR
OCR (baseline system)	88.14%	63.04%
OCR+LM	95.56%	85.80%
OCR+LM+Dic	95.33%	87.83%

4.7 Conclusion

In this chapter, we have presented our first OCR scheme dedicated to texts embedded in videos or captured in scene images. This scheme, called segmentation-based OCR, relies on a character segmentation step that permits to separate characters

before their recognition. The first contribution of this system lies in the computation of nonlinear segmentation borders well adapted to the local morphology of the text image. These segmentations permit to separate characters reliably and hence enable good recognition performance. In contrast to the dominant methodology that recognize characters by means of hand-crafted features, our approach proposes a neural classification model able to learn to deal with extremely various character images without any preprocessing step. This recognizer provides outstanding results and outperforms methods relying on SVM models (98.04% and 85.13% compared to 81.18% and 72.30% of character recognition rate on "caption" and "scene" character images). From character to text, another strength of our OCR system lies the integration of linguistic knowledge (namely a language model and a dictionary) that supervises the complete recognition scheme taking into account the lexical context. This knowledge allows to reduce segmentation errors and recognition ambiguities, and hence to improve the system performance.

The proposed OCR system was tested on two datasets of "caption" texts extracted from digital videos and "scene" text images. Experiments showed that our approach performs well on "caption" texts achieving good performance that exceeds 87% of word recognition rate. However, results obtained on "scene" texts remain poor (with a word recognition rate of about 41%). These results highlight limits of the character segmentation step. Indeed, when the segmentation fails to find the correct separation between characters, it automatically induces a recognition error. In the special case of "scene" text images, that often present various kinds of distortions, several segmentation errors are produced, directly decreasing the recognition performance of the complete OCR scheme. In the following chapters, we explore segmentation-free approaches to tackle this issue.

Chapter 5

The segmentation-free approach

5.1 Introduction

Our first OCR system relied on the segmentation of text images into individual characters. Though characters are segmented depending on the local morphology of the image, enabling more accurate recognition, in the case of text images with strong distortions (such as natural "scene" texts, see Fig. 4.15), recognition performance remains low. This fact can be justified by the character segmentation step that can lead to potential under- or over-segmentations producing several recognition errors (the analysis of remaining recognition errors demonstrated that over 62% of them are due to wrong segmentations).

In order to tackle this difficulty, we propose a second OCR system that avoids any explicitly character segmentation and addresses the problem in a way different from main prior approaches (that are segmentation-based). Using a multi-scale scanning process, our second method recognizes characters directly at their appropriate scale and position within the whole text image.

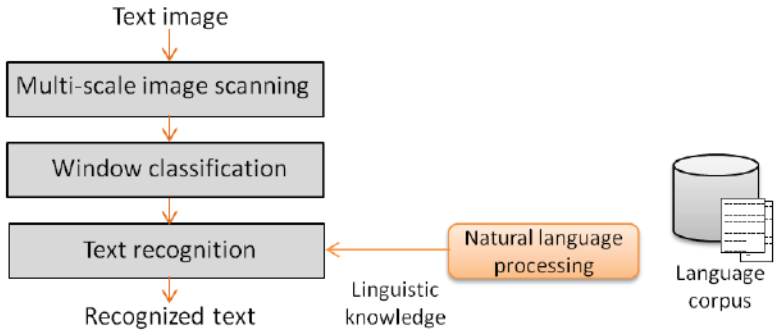

Figure 5.1: The segmentation-free OCR scheme.

This chapter presents this text recognition approach and details its different steps. As shown in Fig. 5.1, text images are first scanned using sliding windows at various

scales (*cf.* section 5.2). Obtained windows are then classified (*cf.* section 5.3) and represented by means of a graph model in order to recognize texts (*cf.* subsection 5.4.1). Some linguistic knowledge is also incorporated within the recognition process to improve the system's performance (*cf.* subsection 5.4.2). The proposed OCR is evaluated on both DatasetI and DatasetII and performance is compared to those of the segmentation-based approach (presented in chapter 4) and of other state-of-the-art methods (*cf.* section 5.5).

5.2 Multi-scale scanning scheme

In contrast to the dominant methodology that aims at segmenting the text image into separated characters, our proposal consists in scanning the full text image with sliding windows. Since the main goal of this step is to cover all characters at their appropriate position and scale a multi-scale scanning process with windows of different sizes is performed. By that way, we hope that at least one window will be aligned with each character within the text image. This section describes the proposed method that consists in scanning text images at different scales (see subsection 5.2.1) using windows with nonlinear borders (see subsection 5.2.2) adapted to the local morphology of the image.

5.2.1 Text image scanning

The first step of our segmentation-free OCR consists in scanning a text image with sliding windows aiming at covering all characters at their proper position in the image. To do so, we perform a scanning method in which successive windows are moved from the left to the right and centered at regular and close positions (not to miss any character). In our experiments, best results were obtained with a moving step of one eighth of the image height h.

Furthermore, since characters belonging to a same word can be of different sizes depending on their labels and their fonts, we consider windows at various scales (*i.e.*, windows with different widths) in order to cover different character sizes. Four scales (namely S_1 , S_2 , S_3 and S_4) are used in our experiments, corresponding to window widths equal to $h/4$, $h/2$, $3h/4$, and h. Fig. 5.2 illustrates an example of a text image scanned at these different scales and shows characters framed at their corresponding scales (*e.g.*, "S" and "e" are framed with windows equal to $3h/4$ and $h/2$ respectively) and an example of a misaligned window (centered between the "h" and "o" letters).

Using this multi-scale scanning process, several windows are considered per text image. The total number of windows is determined according to the size of each image—namely the width w and the height h (in our case $\frac{32\times w}{h} - 16$ windows). Considering $step = h/8$ the moving step of the scanning scheme, the numbers of windows at each scale are:

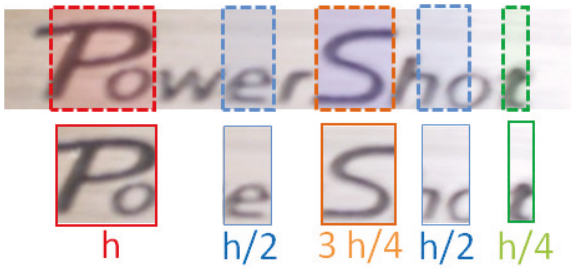

Figure 5.2: Examples of characters well framed at scales S_4 (red), S_2 (blue), S_3 (orange), and S_1 (green) and a misaligned window at scale S_2 (blue).

- $(w/step) - 1$: windows at the scale S_1 centered at positions $h/8$, $h/4$, $3h/8$, ...
- $(w/step) - 3$: windows at the scale S_2 centered at positions $h/4$, $3h/8$, $h/2$, ...
- $(w/step) - 5$: windows at the scale S_3 centered at positions $3h/8$, $h/2$, $5h/8$, ...
- $(w/step) - 7$: windows at the scale S_4 centered at positions $h/2$, $5h/8$, $3h/4$, ...

Fig. 5.3 illustrates the multi-scale scanning process and shows examples of resulting sliding windows at scales S_1, S_2, S_3, and S_4. The observed overlapping windows can be explained by the small moving step ($step = h/8$).

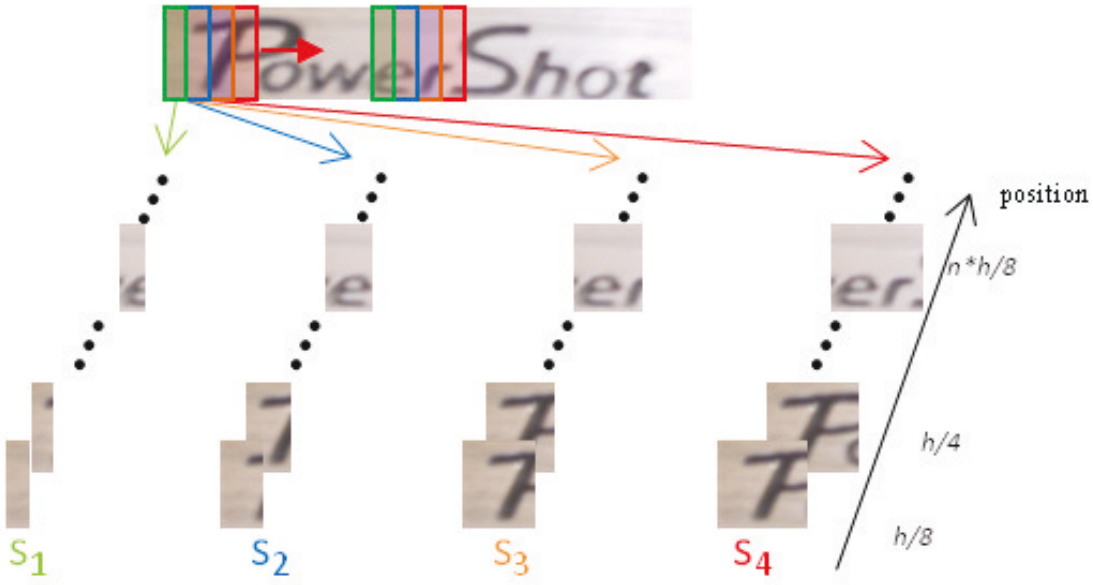

Figure 5.3: Multi-scale text image scanning scheme.

This multi-scale scanning scheme provides for each text image a set of windows. A classification step (see section 5.3) is then required to analyze every window, recogniz-

ing well-framed characters and identifying those containing non-valid ones. Nevertheless, before applying this classification, a nonlinear window borders computation (*cf.* next subsection) is integrated in order to clean well-framed character neighborhoods and hence facilitate the classification.

5.2.2 Nonlinear window borders computation

Although the proposed multi-scale scanning process helps to cover different scales and positions of characters in the text image, vertical borders of windows can extract also parts of neighbors of centered characters (*e.g.*, the "o" is present in the first window in Fig. 5.2). Hence, such linear borders can decrease the performance of window image classification.

In order to tackle this limit and increase the recognition accuracy of our OCR system, we propose to adapt the sliding window borders to the local morphology of the image. The purpose is, when possible, to clean the neighborhood of characters by removing parts of characters that could be extracted with the centered one. Typically, at each window position and scale, two nonlinear borders—namely the right and the left ones—are defined as shortest paths from the top to the bottom of the image.

The computation of these borders is performed using the same processing as for the determination of the nonlinear segmentations described in the previous chapter in section 4.2. First, a fuzzy map which encodes, for each pixel, its membership degree to the class "text" is generated. Using the resulting map, a shortest path algorithm is therefore applied to compute nonlinear borders following pixels with a low membership degree to belong to the class "text".

In contrast to the computation of nonlinear segmentations in chapter 4, since we need a path for each window border, these paths are allowed to cross pixels with a high probability to belong to the class "text". Indeed, in the case of important image distortions, non-separated characters or misaligned windows, the shortest path algorithm induces straight vertical borders since pixels in the local area have the same probability. Resulting borders are finally characterized by a score, corresponding to the value of their highest pixel probability (the pixel with the highest probability to belong to the class "background"). These border scores encode their probabilities to frame correctly the centered character.

Nonlinear borders are then used to remove parts of neighborhood characters, which are replaced by a uniform value corresponding to the mean of the class "background" (see section 4.2). By that way, sliding windows with clean neighborhood are generated enabling an accurate character recognition. Fig. 5.4 shows some obtained sliding windows with their nonlinear borders and also illustrates an example of a misaligned window with straight vertical borders. The comparison between the first window of Fig. 5.4 and the first one of Fig. 5.2 highlights the contribution of the nonlinear borders of windows that permit, in this example, to remove the "o" and hence make the recognition of the character "P" easier. In section 5.5, we demonstrate the contribution

of these nonlinear borders, evaluating their influence on the whole OCR system's performance.

Figure 5.4: Examples of sliding windows with nonlinear borders.

5.3 Window classification

The multi-scale scanning process, applied on each text image, produces several windows of different sizes. The recognition of well-framed characters is thus required to determine the text present in the image. However, before this recognition, a step of pre-filtering is necessary to identify the sliding windows containing "valid" characters and those containing "non-valid" ones.

In this context and for the same reasons as those explained in section 4.3, we chose to use a ConvNet as a model to classify sliding windows into "valid character" or "garbage" (*i.e.*, window misaligned with a character, part of a character or interstice between characters). To do so, for each character dataset—namely (CharDatasetI, GarbDatasetI) and (ChardatasetII, GarbDatasetII) described respectively in subsections 3.1.2 and 3.2.2—, several network architectures were tested for our classification task. The best configuration, hereafter WCConvNet for Window Classifier ConvNet, takes as input a color window image mapped into three maps of size $T \times T$ pixels (T is fixed to 36 and 48 respectively in the case of DatasetI and DatasetII), containing values normalized between -1 et 1. The WCConvNet output is a single neuron, using a hyperbolic tangent activation function. The response of this single neuron encodes the probability of the input window to correspond to a "valid" character. Note that the architecture of the WCConvNet is similar to that of the CRConvNet presented in section 4.3.2.

After this first classification, windows identified as not containing a character are labeled as "garbage", while remaining windows can now be analyzed to recognize the characters they contain. To that end, we leverage the CRConvNet introduced in section 4.3.2, and present these windows (with "valid" characters) to the CRConvNet's architecture whose task is to determine the class of the character well framed in the window.

At the end, each sliding window is labeled as "garbage", or by the "valid" character recognized by the CRConvNet. The whole classification scheme is illustrated in Fig. 5.5 which shows examples of sliding windows of a text image classified through the proposed processing: each well-framed character (such as "o") is identified as "valid"

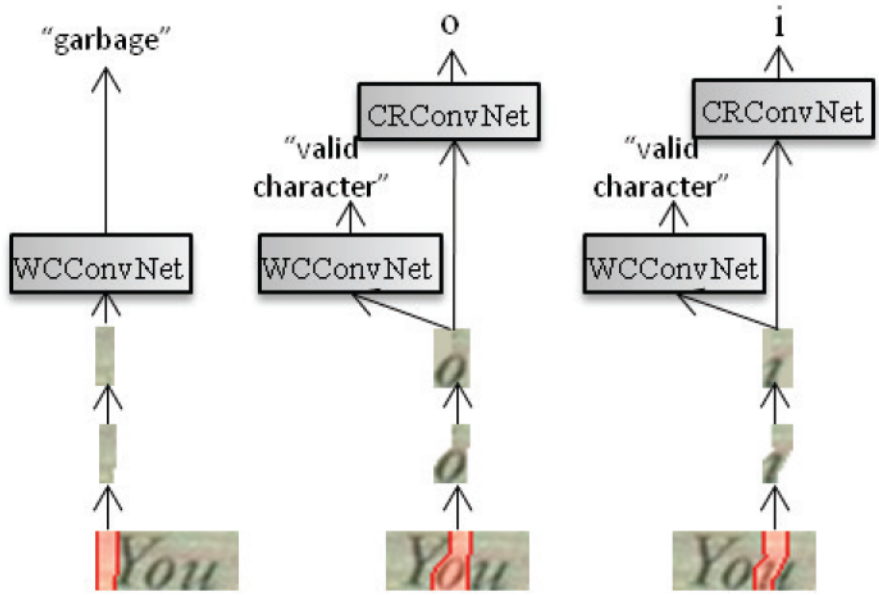

Figure 5.5: The sliding windows classification scheme: examples of windows at scale S_2 (from the left to the right, windows are placed at positions $\frac{h}{4}$, $\frac{10h}{8}$, and $\frac{17h}{8}$).

and then recognized, while interstices between characters are identified as "garbage". Nevertheless, some parts of characters can still introduce recognition confusions (*e.g.*, the part of "u" recognized as "i"). In the next section, we present the graph model proposed to deal with window classification results and to handle recognition errors.

5.4 Text recognition using a graph model

Now that text images are scanned at different scales and that "garbage" and valid characters are classified, the next step is to combine multi-scale window classification results in order to recognize the full text present in the image.

Since our multi-scale scanning process involves many overlapping windows, a graph model, able to represent spatial constraints between sliding windows, is built and used to recognize the whole text present in the image—namely the sequence of well-framed characters recognized in the window classification step—(*cf.* subsection 5.4.1). Some linguistic knowledge is then incorporated into the graph to supervise the complete OCR scheme and remove recognition ambiguities (*cf.* subsection 5.4.2).

5.4.1 Graph model construction

In order to recognize the text or the sequence of characters embedded or captured in an image, we chose to use a directed acyclic graph model to represent the different

multi-scale sliding windows and their classification results.

Indeed, a directed acyclic graph is a special structure consisting of a set of vertices connected with directed edges (ordered pairs of vertices). It permits to represent multiple routes (*i.e.*, sequences of edges) from a given source (one vertex) to a given target (one vertex). Furthermore, when weights (numeric values) are associated with edges, the graph allows to evaluate each route and thus to deduce the one with, for example, the minimal cost (or the most probable).

For our text recognition problem, a graph model where vertices correspond to windows borders (*i.e.*, positions between two successive characters) is considered. In order to encode the spatial constraints between the different sliding windows through the text image, directed edges are built to represent windows (*i.e.*, characters) by joining each two vertices corresponding to the left and the right borders of each window. Considering the fact that four window scales are used to scan the text image, each vertex v is thus connected to 4 successor and 4 predecessor vertices (*i.e.*, the right borders of the four different windows starting from v and the left borders of the four windows ending at v). Fig. 5.6 shows one part of a graph built on a simple image and illustrates windows' positions and scales.

In our case, the source (resp. target) vertex corresponds to the left (resp. right) border of the window centered at the first (resp. last) position in the text image. Our text recognition problem can thus be formulated as searching the most probable path (*i.e.*, sequence of arcs or windows corresponding to characters) that links the source vertex to the target one.

To weight our graph model, results of window classification—namely the class "garbage" and the output of the WCConvNet for non-valid characters, and the label of the recognized class and its corresponding score in the CRConvNet for valid characters—are assigned to each edge. Furthermore, since nonlinear borders of windows are characterized by scores encoding their probabilities to correspond to separations between successive characters (see section 5.2), a weight corresponding to this score is assigned to each vertex.

Using the built graph, all possible paths within the graph are tested and their scores calculated. These text scores are computed as the sums of the logarithms of the values assigned to each edge—namely the WCConvNet or the CRConvNet outputs of the input window—and the logarithms of the mean of the values assigned to the edge's vertices—namely the probabilities that the window borders correspond to separations between characters. Regarding edges representing "garbage", since they correspond to windows misaligned with any character, part of a character or interstice between characters, paths containing these edges are penalized and removed even if they yield the best score. The recognized text is determined as the sequence of characters corresponding to the most probable route (*i.e.*, the path or sequence of successive arcs that obtains the highest score). In practice, to determine the most probable sequence of characters the classical Viterbi algorithm is applied on the graph searching the best path that avoids edges of non "valid" characters.

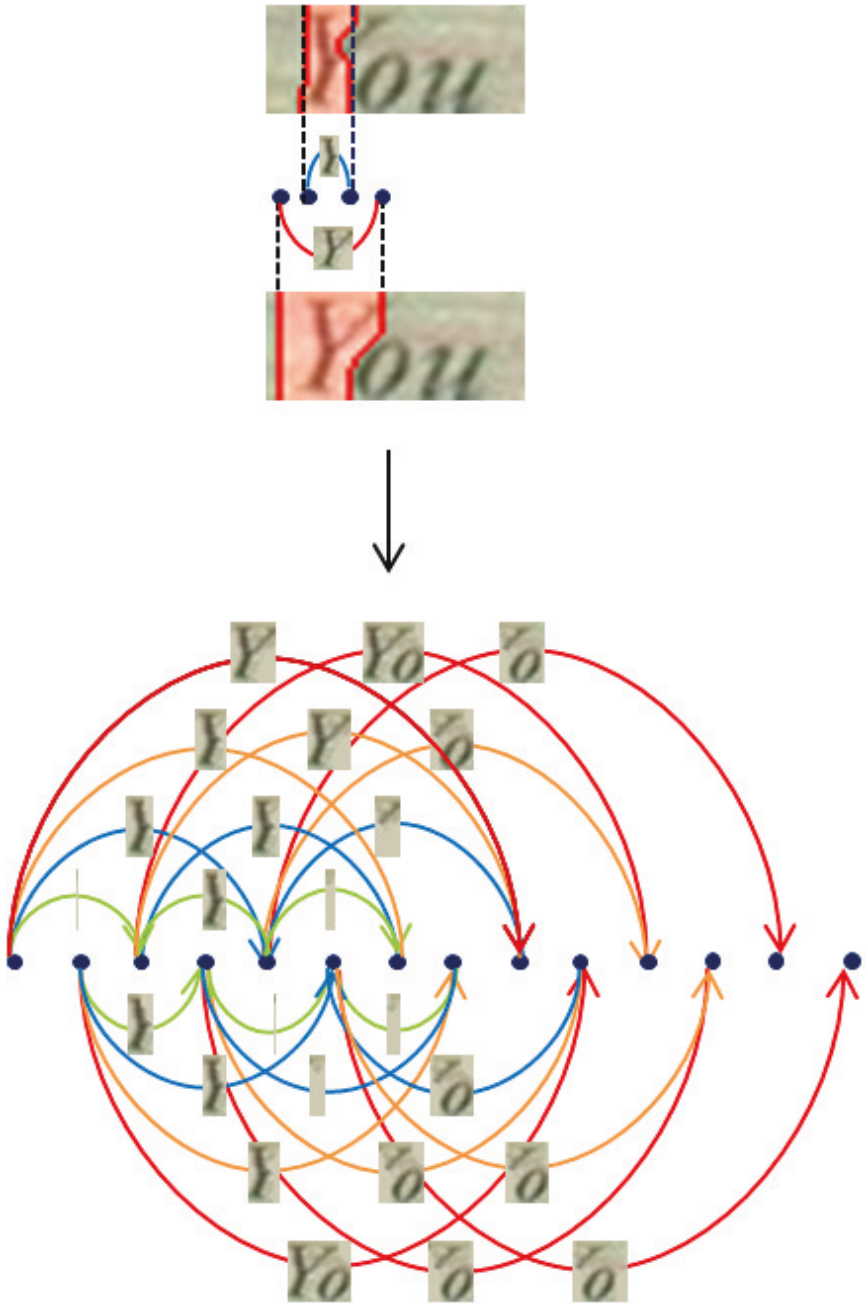

Figure 5.6: Graph model construction: the figure at the top illustrates the representation of window borders by vertices (*i.e.*, black dots) and the one at the bottom shows a part of the obtained graph (windows at scale S_1, S_2, S_3 and S_4 are represented respectively by green, blue, orange and red directed edges).

5.4.2 Integration of linguistic knowledge

In subsection 5.3, we have shown the window classification processing that is applied at several scales and positions in the text image. In the case of some misaligned windows, this step can generate confusions between characters and thus introduce errors reducing the system's performance. Fig. 5.7 illustrates an example of a window classified as a "valid" character though it corresponds to "garbage".

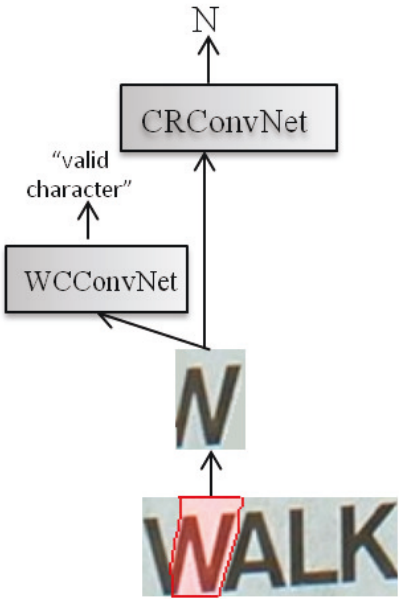

Figure 5.7: Example of a confusing window.

To remove these ambiguities, we propose to incorporate some linguistic knowledge that permits to take into consideration the lexical context. In section 4.5, we have shown that a statistical language model is well adapted to our text recognition problem and permits to improve recognition performance. Indeed, a character n-gram model is able to estimate the joint probability that a sequence of characters is observed in a given language, and we have demonstrated that when these probabilities are used, several recognition errors can be removed.

In this text recognition problem, we propose to integrate these probabilities into the graph model in order to adjust transitions between vertices. Typically, for each path within the graph, its score is computed taking into account the probability of the recognized sequence of characters (which corresponds to the sequence of edges constituting the path). This score is calculated as follows:

$$\hat{C} = \arg max \sum_{i} (log(s_i^{bor}) + log(p(w_i|c_i)) + \gamma \cdot log(p(c_i|\phi(h_n(i)))) + \delta) \qquad (5.1)$$

where $\hat{C} = c_1, c_2, ..., c_l$ is the recognized sequence of characters of length l, s_i^{bor} is the score of the borders of the window w_i, $p(w_i|c_i)$ is the classification result of w_i (namely the output of the CRConvNet, since windows labeled as "garbage" are discarded), $p(c_i|\phi(h_n(i)))$ is the probability provided by the n-gram model, γ and δ are parameters respectively encoding the weight of the language model and compensating over- and sub-segmentations.

In order to reduce character confusion errors, for each window containing a "valid" character, the best five responses of the CRConvNet are considered in the graph and a different score is computed per response. Using these scores, all paths avoiding windows labeled as "garbage" are evaluated and the optimal text is finally obtained using the Viterbi algorithm which permits to take into account positions and scales of windows, their recognition results, and some language properties provided by the language model.

The two character n-gram language models used in our experiments on DatasetI and DatasetII are the same ones as in the previous chapter (see section 4.5.2).

5.5 Experimental results

This section reports the evaluation of the proposed segmentation-free OCR system on both datasets: the "caption" texts of DatasetI and the "scene" texts of DatasetII.

After a presentation of the performance of the proposed individual window classifier, the complete OCR system is evaluated and compared to state-of-the-art methods, emphasizing the benefits and the limits of the character segmentation step by comparing the results of this segmentation-free OCR and of our first segmentation-based one. The contributions of the different processing steps (the multi-scale scanning scheme, the computation of nonlinear window borders and the integration of the linguistic knowledge) incorporated in the OCR are finally highlighted.

5.5.1 Performance of the window classifier

Our window classifier consists in combining the WCConvNet, whose task is to identify windows containing garbage and windows with "valid" characters, and the CRConvNet which recognizes classes of identified "valid" characters. Hence, this subsection focuses on the training and evaluation of both types of ConvNets (see table 5.1).

Two WCConvNets were trained to classify windows in DatasetI and DatasetII. The first one was trained on CharDatasetI and GarbDatasetI (which contains images of non-valid characters) while the second was trained on CharDatasetII and GarbDatasetII. In both experiments, the datasets were divided randomly into three subsets: a training set containing 80% of the images, a validation set and a test set, containing each 10% of the images. The training and the validation sets are used to train the WCConvNets and the test set serves to evaluate the classification performance.

Table 5.1: Classification performance of the WCConvNet and the CRConvNet : RR means Recognition Rate.

	Character RR	
	(CharDatasetI, GarbDatasetI)	(CharDatasetII, GarbDatasetII)
WCConvNet	87.99%	79.23%
CRConvNet	98.04%	85.13%

As shown in table 5.1, results obtained on DatasetI are better than those obtained on DatasetII (about 9% of difference). This is due to the high variability of "scene" texts (*i.e.*, DatasetII) compared to "caption" texts (*i.e.*, DatasetI). Indeed, the serious distortions present in natural "scene" text images make some characters looking like "garbage", thus making it difficult for the WCConvNet to distinguish them from "garbage". Notice that the WCConvNet output threshold has been chosen to emphasize the recall (the rate of valid characters correctly classified), since any valid character classified as garbage will be discarded in the graph. Fig. 5.8 shows some confusing characters in DatasetII.

Figure 5.8: Examples of confusing characters (that can be identified as garbage) of CharDatasetII.

The training of the CRConvNet was presented in the previous chapter in subsection 4.6.1. Table 5.1 reminds of the obtained performance.

5.5.2 Performance of the segmentation-free OCR

Using the different processing steps incorporated in our OCR system, the recognition performance can now be evaluated on TextDatasetI and TextDatasetII. Fig. 5.9 presents an example of a recognized text and illustrates the resulting best path within the graph model. Note that different γ and δ values were tested (see Eq. 5.1), and those yielding to best results were retained ($\gamma = 0.2$ and $\delta = 1.2$).

Experiments carried out on TextDatasetI (*cf.* table 5.2) show that our segmentation-free OCR performs well on "caption" texts and obtains 93.55% of character recognition rate, corresponding to about 81% of words correctly recognized. The proposed OCR also achieves good results on natural "scene" texts (TextDatasetII) obtaining a character recognition rate above 70%, corresponding to a word recognition rate of about 47%. The difference between the performance achieved on the two datasets can be explained by two main facts:

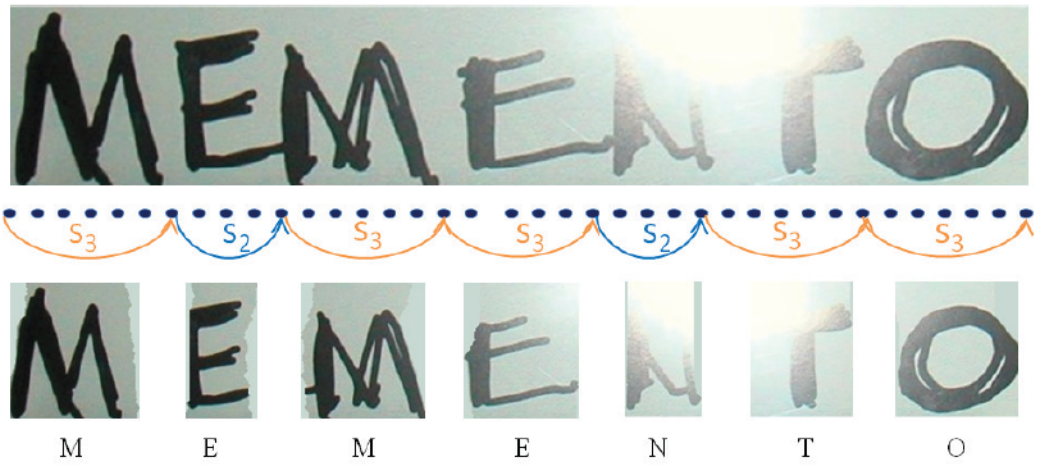

Figure 5.9: Example of recognized text: the first line shows the text image, black dots are the vertices of the graph, directed edges represent the best path obtained with the Viterbi algorithm and the last line illustrates the sliding windows identified as containing "valid" characters then recognized with the CRConvNet.

- the window classifier—namely the WCConvNet and the CRConvNet—performs better on CharDatasetI than on CharDatasetII;
- TextDatasetI is generally less complex than TextDatasetII where the task of the best path search algorithm to find the correct text is particularly difficult.

Table 5.2: Recognition performance of the segmentation-free OCR: RR means Recognition Rate.

	TextDatasetI		**TextDatasetII**	
	Character RR	Word RR	Character RR	Word RR
Segmentation-free OCR	93.55%	81.32%	70.33%	46.72%

Our recognition results were compared with those of two state-of-the-art methods [SGD09, WB10]. For the same reasons as those explained in subsection 4.6.2, this comparison was performed only on TextDatasetII. Two experiments were performed: we evaluate our OCR scheme on the 901 images selected in [SGD09] (Exp1) and on the 1,065 images selected in [WB10], using the same lexicon, created from all the words that appear in TextDatasetII as in [WB10] in the same way as mentioned in subsection 5.4.2 (Exp2). Results are reported in table 5.3 and show that our approach achieves the best word accuracy. It outperforms Saïdane *et al.*'s one [SGD09], that relies on a character segmentation step, by about 3%. Our approach also yields better results than Wang *et al.*'s one [WB10] by about 7% even though their method avoids the segmentation step and uses hand-designed features to recognize characters.

This demonstrates that our window classification, based on a combination of neural networks, and the built graph model are well adapted to our recognition problem.

The proposed OCR is also compared to two commercial OCR engines, namely ABBYY FineReader and Tesseract. Let us remind that due to some practical issues, these commercial OCRs were not trained on the same datasets as our OCR and other state-of-the-art methods. Results on TextDatasetI (*cf.* table 5.3) show that our segmentation-free OCR achieves better performance than Tesseract (+5% of character recognition rate). However it performs slightly worser that ABBYY FineReader (−1.5% of character recognition rate). This can be explained by the use of a dictionary in ABBYY FineReader absent in our OCR. Concerning experiments on TestDatasetII, ABBYY FineReader and Tesseract, evaluated by Wang et al. [WB10] and using a lexicon, created from all the words that appear in TextDatasetII, obtained about 42% and 35% of word recognition rate, while our segmentation-free OCR achieves far better performance with +24% and +31%. Hence, we can conclude that despite ABBYY FineReader achieves good results on "caption" texts, our segmentation-free OCR proves with its great ability to handle both "caption" and "scene" texts.

Table 5.3: Comparison of the proposed OCR system to state-of-the-art methods and commercial OCR engines: RR means Recognition Rate (For TextDatasetII, only word RRs are reported because they are the only performance evaluated for other existing methods).

OCR system	**TextDatasetI**		**TextDatasetII**	
			Exp1	**Exp2**
	Character RR	Word RR	Word RR	Word RR
Segmentation-free OCR	**93.55%**	**81.32%**	**57.04%**	**66.19%**
Saïdane *et al.* [SGD09]	-	-	54.13%	-
Wang *et al.* [WB10]	-	-	-	59.20%
ABBYY FineReader OCR	95.03%	87.70%	-	42.80%
Tesseract OCR	88.57%	70.01%	-	35.00%

Finally, we have compared our segmentation-free OCR to our first segmentation-based one, described in chapter 4, in order to discuss and highlight the benefits and the limits of the character segmentation step.

Table 5.4: Comparison of the segmentation-free OCR to the segmentation-based one presented in chapter 4: RR means Recognition Rate and Char character.

OCR system	**TextDatasetI**		**TextDatasetII**	
	Character RR	Word RR	Character RR	Word RR
Segmentation-free OCR	**93.55%**	**81.32%**	**70.33%**	**46.72%**
Segmentation-based OCR	95.33%	87.83%	65.33%	41.19%

Results presented in table 5.4 show that both OCRs perform well on "caption" texts

(*i.e.*, TextDatasetI) with more than 93% of good character recognition rate (less than 2% of difference in term of character recognition rate). However the segmentation-based OCR is slightly better. This proves that when the character segmentation step works well (particularly on text with small distortions like "caption" text), it enhances the following steps of the OCR and leads to better recognition performance.

On the contrary, performance achieved on "scene" texts (*i.e.*, TextDatasetII) points out a larger difference between results obtained with our segmentation-free OCR and our segmentation-based one. While the first one achieves a character recognition rate above 70% and a word recognition rate of about 47%, the second one obtains 65% of character recognition rate and 41% of word recognition rate. These results highlight the drawbacks of the character segmentation step where any error directly induces a drop in the recognition accuracy. This limit is enhanced in the special case of natural "scene" texts where images are often affected by various distortions making the segmentation very hard and thus leading to several errors such as over- and sub-segmentations. Fig. 5.10 illustrates an example of a "scene" text where the segmentation-based OCR produces several over-segmentations leading to a considerable number of errors while the segmentation-free OCR is able to recognize the correct text captured in the scene image.

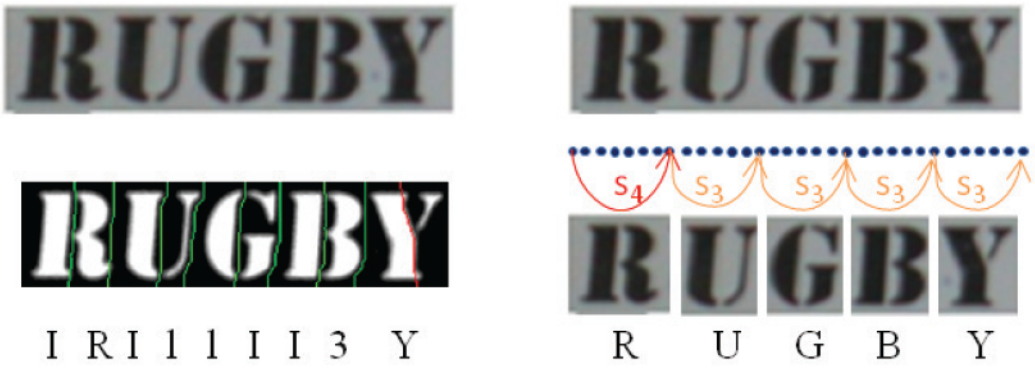

Figure 5.10: Example of a "scene" text recognized with the segmentation-based OCR (on the left) and the segmentation-free OCR (on the right): the left part illustrates the obtained segmentations within the fuzzy map and the final recognition result, and the right part shows the computed best path within the graph model and the recognized text.

5.5.3 Contributions of incorporated processing steps

After presenting the global OCR performance, we evaluate here the contribution of the different steps of the scheme. To that end, three experiments were performed to highlight the influence of three processings:

- The multi-scale scanning process (MSS) evaluated against a test using only one sliding window size,
- The nonlinear borders computing phase (NLB) evaluated against a test considering sliding windows with linear borders,
- The language model integration (LM), evaluated against a test removing this linguistic knowledge.

Results are shown in table 5.5. They confirm that multi-scale scans (MSS) are mandatory to cover different sizes of characters promoting satisfactory results. In the absence of this processing, the performance of the OCR notably decreases both on TextDatasetI and TextDatasetII to 15.74% and 0.12% of word recognition rate. The consideration of the nonlinear borders of the sliding windows (NLB) also results in an important improvement of the character recognition rate: the comparison of the performance achieved with MSS+LM (*i.e.*, without NLB) and those of the complete scheme in table 5.5 demonstrates that this processing step enables to increase the word recognition rate from about 56% to 81% in the case of TextDatasetI and from about 27% to 46% in the case of TextDatasetII. The language model (LM) also permits to improve the OCR performance on TextDatasetI and TextDatasetII by about 27% and 21% of word recognition rate. Note that the language model used in our experiments is a character tri-gram language model, since it has been shown to be the most adapted to our recognition problem (see subsection 4.6.3).

Table 5.5: Contributions of different processing steps incorporated in the proposed OCR scheme: RR stands for Recognition Rate, NLB for nonlinear borders, MSS for multi-scale scanning and the complete scheme means MSS+NLB+LM, *i.e.*, our segmentation-free OCR.

Method	**TextDatasetI**		**TextDatasetII**	
	Character RR	Word RR	Character RR	Word RR
NLB+LM	41.35%	15.74%	9.12%	0.12%
MSS+LM	74.67%	55.98%	53.41%	27.18%
MSS+NLB	72.30%	54.00%	54.32%	25.54%
Complete scheme	**93.55%**	**81.32%**	**70.33%**	**46.72%**

5.6 Conclusion

In this chapter, we have presented our first segmentation-free OCR system, one main strength of which lies in the absence of the traditional character segmentation step,

which is involved in most existing methods. The avoidance of this step is done by incorporating a multi-scale scanning scheme where sliding windows with nonlinear borders are moved to cover the characters of a given text image at their different positions and scales. The second contribution of this OCR is the designed window classifier relying on a robust machine learning model. This neural model is able to deal with various kinds of images and to learn to recognize "valid" characters and identify non-valid ones, directly from images without any preprocessing. A directed acyclic graph model has also been proposed to represent spatial constraints between windows and to enable the recognition of texts using a Viterbi algorithm. The integration of some linguistic knowledge, taking into account the lexical context, is also a strong point of our system. This permits to reduce several errors and hence to improve the system' performance.

The proposed scheme was evaluated on two datasets, TextDatasetI and TextDatasetII, highlighting the contribution of its processing steps including the multi-scale scanning scheme, the nonlinear borders computation and the language model integration. Our approach was also compared to state-of-the-art methods and commercial OCR engines. Experiments carried out have shown that our method obtains outstanding performance on both "caption" and "scene" texts, and yields the best word recognition rate among concurrent approaches.

The comparison of this segmentation-free OCR to our segmentation-based one demonstrated that both systems perform very well in the case of "caption" text images, while in the case of natural "scene" text images, the segmentation-free system achieves better results. On the one hand, these results point out the benefits of the character segmentation step that enhances the recognition performance when it operates well, particularly on "caption" texts with few distortions. On the other hand, they prove that the segmentation step directly decreases the recognition performance when it produces errors, particularly in the case of "scene" texts with strong distortions.

Even though our OCR avoids the segmentation phase by integrating a multi-scale scanning process and using a graph model, the complexity of the latter, where many paths have to be tested, remains the main weakness of this method. In the next chapter, we tackle this issue and propose a different way to avoid any character segmentation step.

Chapter 6

The recurrent connectionist approach

6.1 Introduction

In the previous chapter, we have presented a novel approach to recognize texts embedded or captured in images or videos, that avoids the critical character segmentation step by integrating a multi-scale scanning scheme. In order to deal with the sliding windows and their classification results, a graph model is built and used to determine the recognized word as the sequence of characters corresponding to the best path within the graph. Though this method achieves good performance, its main drawback remains the complexity of the graph model where many possible paths have to be tested and evaluated to select the best one. For instance, even for a small text image of size $height \times width$ with $width = 4 \times height$, the corresponding graph model consists of $\frac{width}{step} + 1 = 33$ vertices (where $step = \frac{height}{8}$) and 112 edges (31 edges of scale S_1 starting from the first 31 vertices, 29 edges of scale S_2 starting from the first 29 vertices, 27 edges of scale S_3 starting from the first 27 vertices and 25 edges of scale S_4 starting from the first 25 vertices) inducing a huge number of possible combinations (*i.e.*, paths).

In order to address this issue, this chapter proposes a connectionist temporal approach specifically designed to avoid the graph model and to learn how to automatically recognize the text without any segmentation step. The method consists of three main steps as depicted in Fig. 6.1: a multi-scale text image representation elaboration, a feature sequence classification, and the text recognition itself. In the first step, a text image is scanned at different scales as in the previous chapter, and is represented by a sequence of learnt features vectors. The second step uses a specific bidirectional recurrent neural network (Bidirectional Long-Short Term Memory, BLSTM) able to take into account dependencies between successive learnt features to classify obtained features making use of both future and past contexts. Finally, the network's outputs are analyzed and decoded to obtain the recognized text.

The following sections (6.2, 6.3, and 6.4) successively describe these different steps and their interactions within the recognition scheme. The proposed approach is eval-

uated on DatasetI and DatasetII and results are discussed in section 6.5.

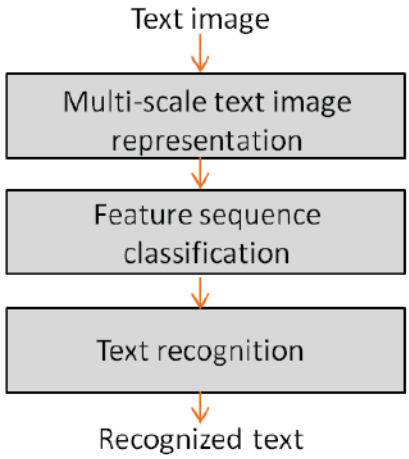

Figure 6.1: The proposed recurrent connectionist OCR scheme.

6.2 Multi-scale text image representation

The first step of our approach consists in producing a relevant representation of text images, robust to noise, deformations, geometric transformations and different kinds of distortions. In the literature, most existing methods propose to use hand-crafted features that aim at encoding prior knowledge about the image. However, these features are often specialized for a specific task or a given dataset and two main major issues remain their lack of genericity and their robustness to noise and deformations. Recently, automatically learnt features-based representations have enjoyed a considerable progress and have been applied to several fields such as object recognition [YYGH09], video action recognition [BMW+11, BMW+12] or audio classification [LLPN09]. For our text image representation problem, we propose to investigate this new direction by exploiting ConvNets, and to use learnt ConvNet as a features extractor. We will also compare these learnt features to geometric hand-crafted ones (*cf.* section 6.5).

To generate a representation, each text image is first scanned with windows of four different sizes (*cf.* section 6.2.1). Each window at a scale s is then represented by a vector X_s of m features learnt with a ConvNet (*cf.* section 6.2.2). Considering the four windows extracted at each scanning position t, a vector X^t is thus produced by concatenating X_1^t, X_2^t, X_3^t, and X_4^t corresponding to these windows. At the end of the scanning process, a sequence of learnt features vectors $[X^0, \ldots, X^t, \ldots, X^{p-1}]$ (with p the number of positions considered in the scanning scheme) is finally generated to represent each text image. Fig. 6.2 depicts these different processing steps, which are detailed in the following subsections.

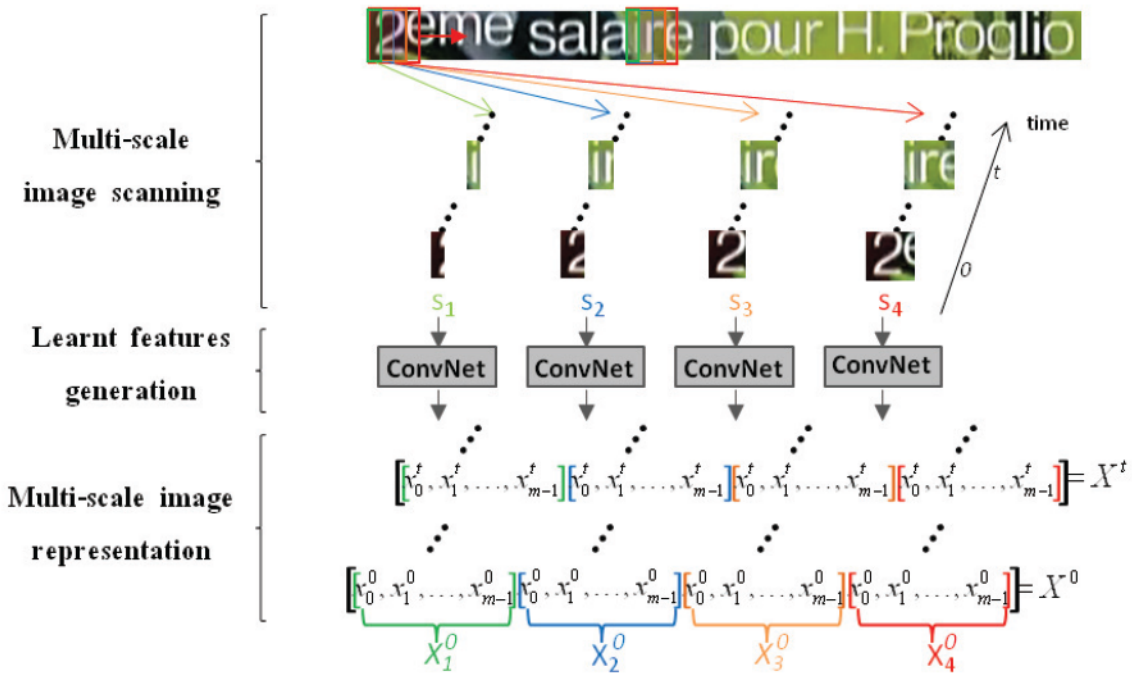

Figure 6.2: Multi-scale text image representation.

6.2.1 Multi-scale image scanning

Text images usually consist of a succession of characters having different sizes and shapes depending on their labels and fonts. Since the objective of the representation phase is to find the accurate features able to summarize the content of characters present in a text image, a multi-scale scanning scheme (which consists in the same processing as the one presented in section 5.2) is used to obtain, for each character, at least one window well aligned with it.

Text images are first scanned at various scales using four sliding windows of width $h/4$, $h/2$, $3h/4$ and h, where h is the height of the image (see Fig. 5.3). At each position t (with $t \in \{0, 1, \ldots, p-1\}$) of the scanning process, four windows at scales S_1, S_2, S_3 and S_4 are thus produced and moved with a step of $h/8$ to cover different possible positions within the text image. Furthermore, window borders are adapted to the local morphology of the image and computed using a shortest path algorithm (see Fig. 5.4).

6.2.2 Learnt features generation

After applying the multi-scale scanning scheme, several windows are obtained, from which accurate representations that preserve the information useful for the recognition task have to be extracted.

As mentioned above, we have chosen to use learnt features because of their gener-

icity and robustness to noise and deformations. To produce these features, a machine learning model able to deal with color images without any preprocessing, and to extract relevant features is required. As explained in previous chapters 4 and 5, among the possible machine learning models, ConvNets [LB95] have shown to be well adapted to our recognition task (*cf.* section 4.3.1) and are particularly able to learn to extract descriptors accurate for visual patterns recognition. For this reason, we propose to benefit from this capacity of the ConvNets to construct our features extractor.

This is done in two phases. First a ConvNet is trained in a supervised way to classify images of individual characters. Once the training phase is finished, we use the penultimate layer as a features extraction layer (*cf.* Fig. 6.3) and hence consider the activations of this layer as an accurate descriptor that summarizes the content of the input image. The choice of the penultimate layer as our features descriptor can be justified as follows. During the training phase, the ConvNet learns to extract some appropriate features for the character recognition task, and to combine these features to decide the classification result. This classification decision is taken in the output layer, while the extraction of the features is ensured by the other layers, producing a final vector of features generated at the activation of the penultimate layer; hence this latter has been chosen as a features extraction layer.

We choose to use the CRConvNet specifically designed in section 4.3.2 to recognize single characters in images as our neural-based model for the learnt features generation. Practically, for each text image, each sliding window resulting of the multi-scale scanning process is presented to the CRConvNet. This window is then represented by a vector composed of the activations of the penultimate layer, that has n_3 neurons in Fig. 6.3. This figure shows the vector of learnt features produced for a single window with the CRConvNet. Notice that, unlike in chapter 5, even misaligned windows (*i.e.*, with a non-valid character) are presented to the CRConvNet and represented by the learnt features.

In our experiments, two CRConvNets were trained to recognize characters: one for images in CharDatasetI and one for images in CharDatasetII. These CRConvNets are the same ones presented in subsection 4.3.2. Using these network architectures and considering the four scales of the scanning process, each position t in the text image is represented by a vector X^t of $4 \times n_3$ values (n_3 values per window scale) corresponding to the features produced by the ConvNet model. Finally, each text image is represented by a sequence of vectors of learnt features: $[X^0, \ldots, X^t, \ldots, X^{p-1}]$, with p the number of positions considered in the scanning process.

6.3 Feature sequence classification

Once text images are represented by sequences of automatically learnt features, the next phase consists in classifying these sequences in order to recognize the text they contain.

Inspired by the work of Graves *et al.* dedicated to handwriting recognition [GLF+09],

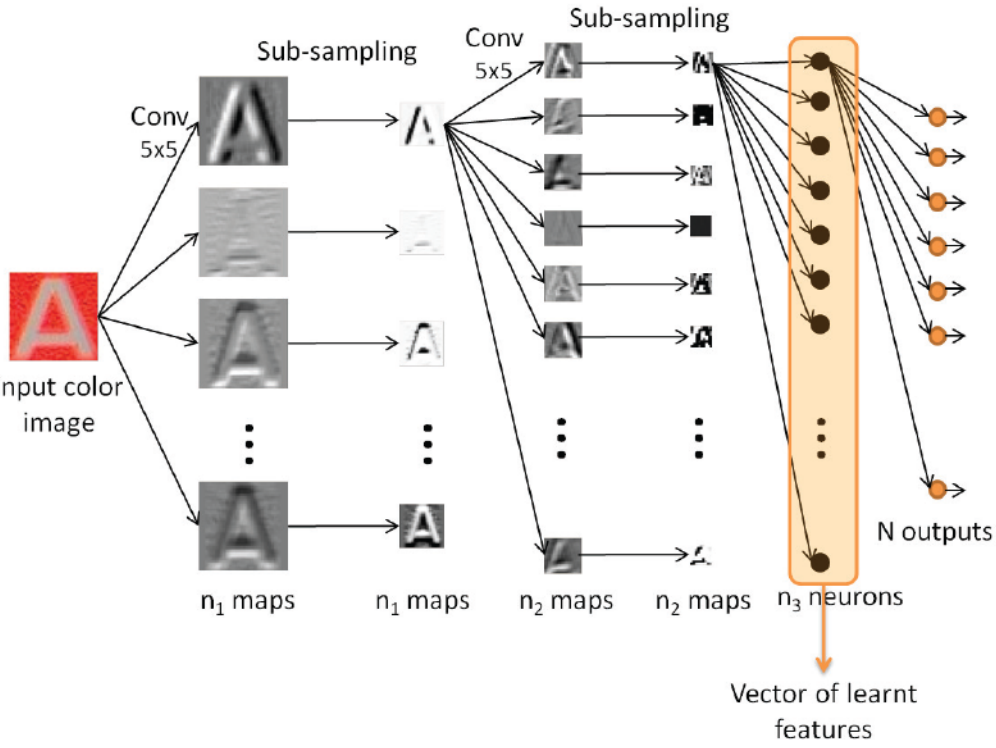

Figure 6.3: The neural-based model for features learning.

we propose to combine a particular recurrent neural network (namely a BLSTM) and a connectionist classification model (namely a connectionist temporal classification, CTC) to build a model able to learn how to classify these feature sequences. On the one hand, the BLSTM allows to handle long-range dependencies between vectors of features, permitting to consider the context of input vectors while classifying. On the other hand, the CTC enables our system to avoid any explicit segmentation into characters. The designed model is hence able to learn jointly to recognize a sequence of classes—namely characters—and to localize their positions—namely characters positions—in the unsegmented input sequence data.

This section starts by a brief introduction to recurrent neural networks together with a detailed presentation of BLSTM networks (see subsection 6.3.1). A description of the CTC model is then provided (see subsection 6.3.2). Finally, the architecture and the training of the network designed for our sequence classification problem is presented (see subsection 6.3.3).

6.3.1 Bidirectional long-short term memory

Recurrent Neural Networks (RNNs) are a particular category of ANNs which have the ability to deal with sequences of data by remembering state variables depending on

previous neural input values and using them to influence the current output. The basic idea of these networks is to introduce recurrent connections that enable the network to maintain an internal state and thus to take into account the past context. The training of these networks is done using a modified version of the back-propagation algorithm adapted to the temporal inputs/outputs. The back-propagation through time algorithm (BPTT) is the most commonly used one [WZ95]. Fig. 6.4 illustrates an example of a RNN which takes as input a sequence of vectors of data of length p and returns a sequence of outputs of the same length p. At each time step t (with $0 \leq t < p$), an input vector X^t of size m is presented to the network which returns an output vector Y^t of size n. Due to the recurrent connections in the hidden layer of the RNN, Y^t is computed taking into account not only the current input, *i.e.*, X^t, but also its context represented by the previous inputs, *i.e.*, X^{t-1}, X^{t-2}, etc.

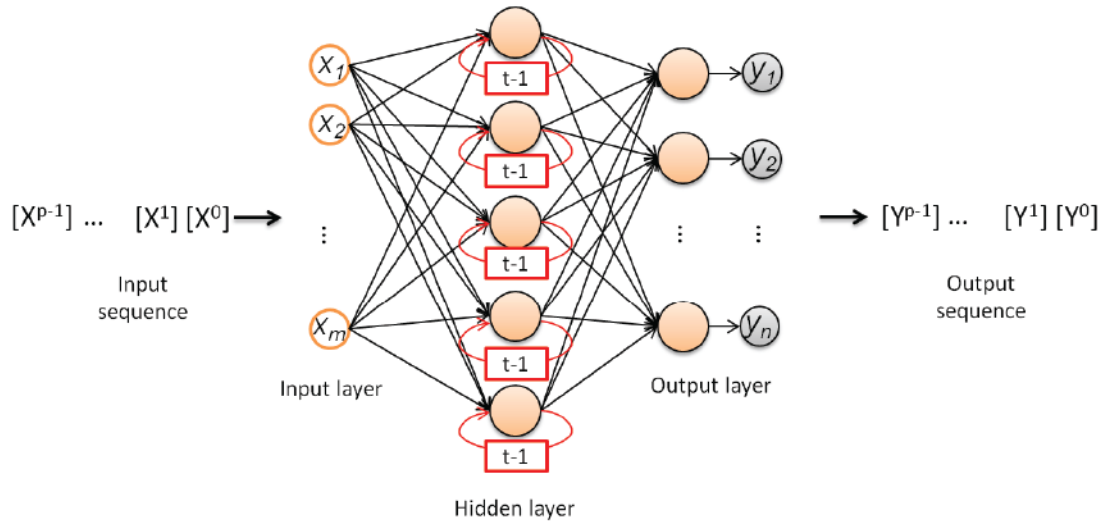

Figure 6.4: An example of a RNN architecture: recurrent connections are drawn in red.

Nevertheless, in [HS97] the authors have shown that these models are not able to handle long time lags in input sequences due to the problem of *exponential error decay*. Thus, these models become insufficient when long input sequences (over 10 or 12 time steps), such as our feature sequences, are considered. To overcome this problem, Gers *et al.* [GSS03] have proposed the Long Short-Term Memory (LSTM) model able to handle data with long range interdependencies. The first key idea of this architecture is the introduction of a special node—namely the constant error carousel (CEC)—that allows a constant "memory cell" (a recurrent weight connection which is set to 1). The second key idea consists in introducing three multiplicative gates—namely the input gate, the output gate and the forget gate—whose task is to control the access to the CEC providing analogy with write, read and reset operations. They permit the CEC to store an information over long periods of time by protecting

it from irrelevant or noisy activations. Fig. 6.5 illustrates a representation of a LSTM neuron. Integrating these two ideas, a LSTM network learns to decide moments (*i.e.*, time steps) when to remember and when to forget a previous input; hence it becomes able to retain only the useful information over a long sequence of inputs.

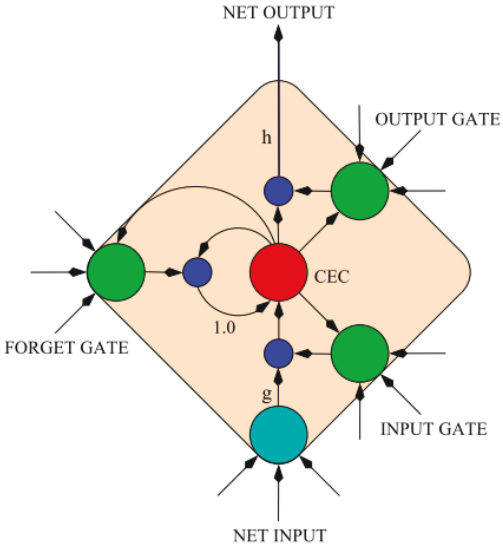

Figure 6.5: A LSTM neuron or cell [Gra08]: the CEC is represented with a red circle; the gates, represented with green ones, control the access to the cell with the multiplicative units drawn as blue small circles; g and h are the input and the output activation functions.

For the reasons explained above, we chose to use a LSTM model to classify our learnt feature sequences. However, in our task of text recognition, the future is as important as the past (*i.e.*, both previous and next letters are important to recognize the current letter). Hence, we propose to use a bidirectional LSTM (BLSTM) architecture [GS05] that allows to consider both past and future contexts. Fig. 6.6 depicts the architecture of such a BLSTM network. It consists of two separated hidden layers of LSTM neurons: the first one permits to process the forward pass making use of the past context, while the second one processes the backward pass making use of the future context. Both hidden layers are connected to the same input layer, which consists of the sequence of learnt features $[X^0, \ldots, X^t, \ldots, X^{p-1}]$ in forward or reverse order. The output layer is also connected to both hidden layers, enabling to make a decision considering past and future contexts; it returns a sequence of vectors outputs $[O^0, \ldots, O^t, \ldots, O^{p-1}]$ where $O^t = \left[o_0^t, o_1^t, \ldots, o_{n-1}^t\right]$ with n the number of character classes considered and o_i^t values between 0 and 1 (normalized with a *softmax* activation function).

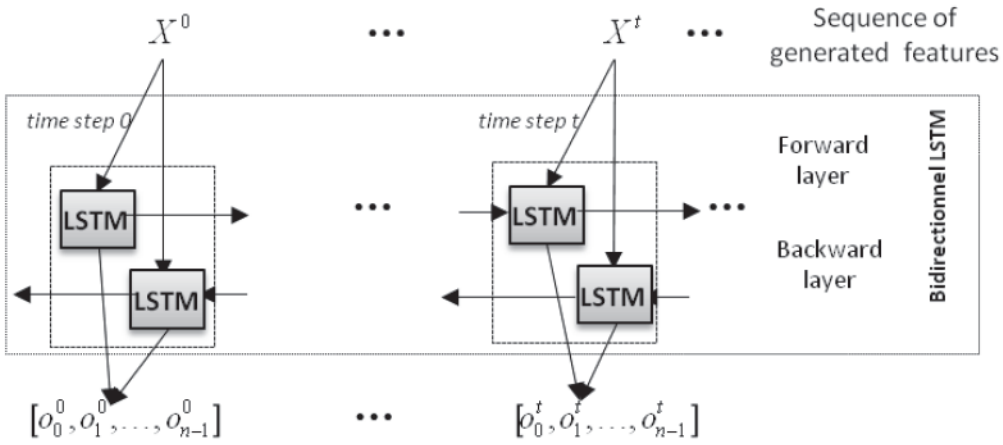

Figure 6.6: Architecture of a BLSTM network.

6.3.2 Connectionist temporal classification

Even though BLSTM networks are able to model long-range dependencies, they require pre-segmented training data allowing the network to learn to provide the correct output at each time step. However, in our text recognition problem, no character segmentation is performed and our sequences of features are not separated (*i.e.*, at a given time step t, the vector of features X^t is not associated with any class of character).

Graves *et al.* [GFGS06] have provided some remedies to this issue by introducing a specific layer or objective function—namely the connectionist temporal classification (CTC)—which allows to extend the use of RNNs to the case of non-segmented data. In [GFGS06], they showed that a BLSTM with a CTC layer outperforms HMMs and RNN-HMM hybrid schemes on a phoneme recognition task. Applied to handwritten text recognition [GLF+09], a BLSTM with a CTC layer also outperforms HMM-based methods.

Let us explain the main principles of the CTC layer. We consider a sequence labeling task that consists in assigning to each input data sequence of length p a target sequence of labels of length l, with l lower than p (for instance, the input sequence can be features representing a text image and the target sequence the text or the sequence of characters in the image). If the input sequence is not segmented (for instance, the features are not segmented with respect to the characters), the CTC enables the recurrent network to handle these data by creating a link between the BLSTM output sequence and the target one.

Fig. 6.7 illustrates this principle. If $X = [X^0, X^1, \ldots, X^{p-1}]$ is a given input sequence, $Y = [Y^0, Y^1, \ldots, Y^{p-1}]$ its corresponding network output and $C = [C^0, C^1, \ldots, C^{l-1}]$ the target sequence, the goal of the CTC is to generate an intermediate label sequence C' of length p (as long as the output sequence). To do so, Graves *et al.* proposed to use an additional class, called class "blank", which is

inserted between the labels of the sequence C. During the training phase, considering the set of training examples $\{X\}$ presented to the network, the CTC determines their corresponding intermediate sequence $\{C'\}$ by minimizing an objective function O expressed as follows:

$$O = -\sum_{(X,C)\in S} ln(P(C|X)) \tag{6.1}$$

where S is a training set containing pairs of input and target sequences (X, C), and $P(C|X)$ is the probability to observe one sequence C given its input sequence X. Since the direct minimization of this objective function requires to examine all possible sequences C', Graves *et al.* proposed an alternative minimization process inspired by the HMM's backward-forward algorithm. More details are given in [GLF+09]. Once the intermediate sequence C' is obtained, an error sequence $e = [e^0, e^1, \ldots, e^{p-1}]$ is calculated and back-propagated to update the network weights (see Fig. 6.7).

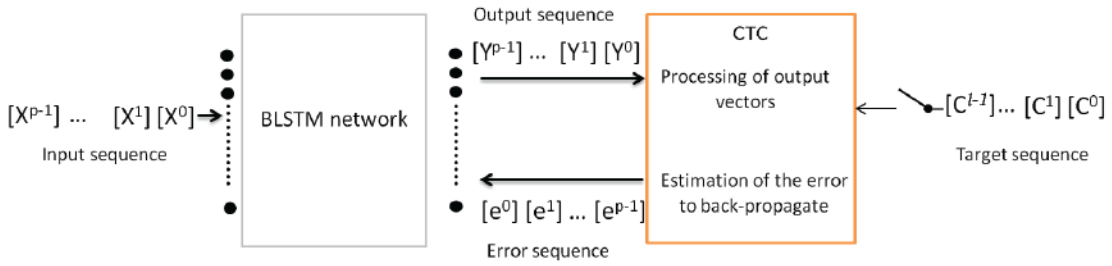

Figure 6.7: Illustration of a CTC layer linking a BLSTM network to a target sequence.

After the training phase, a decoding process permits, given a BLSTM output sequence Y, to obtain the recognized text by interpreting Y and removing "blank" in the sequence C' (see section 6.4).

In our text recognition task, a CTC layer is thus introduced in our system to connect the BLSTM's outputs to the sequence of labels, *i.e.*, the sequence of characters present in the text image. The precise description of this network architecture for our two datasets is provided below; details of the decoding operation are given in section 6.4.

6.3.3 Proposed BLSTM network architectures and training

For our recognition problem, for each dataset—TextDatasetI and TextdatasetII—several configurations were tested to determine the best network architectures.

Regarding TextDatasetI, after testing several architectures, a BLSTM network with two hidden layers (see Fig. 6.6), one for the forward pass (*i.e.*, the past context) and one for the backward pass (*i.e.*, the future context), and a CTC output layer, has been chosen. The network takes as input, at each time step, a sequence of vectors of

200 values ($4 \times n_3$ features, where $n_3 = 50$), namely the learnt features normalized between -1 and 1. The input layer is fully connected to both hidden layers, each one containing 150 LSTM cells with recurrent connections to all the other LSTM cells in the layer. The output layer, fully connected to both hidden layers, returns at each time step a vector of 42 outputs (41 classes of characters, see subsection 3.1.2, and the class "blank") normalized with a *softmax* activation function. The CTC layer takes as input a sequence consisting of these 42 BLSTM outputs values and as target the full text to be recognized C. In our experimental data, depending on the size of the text image, the sequence of inputs can contain up to 300 vectors depending on the number of positions considered in the scanning phase. The BLSTM network is trained with the classical back-propagation through time algorithm [WZ95] on a training set containing pairs of BLSTM input data (*i.e.*, sequences of vectors of learnt features representing text images X) and CTC targets (*i.e.*, the corresponding texts C).

For TextDatasetII, a similar BLSTM network architecture has been chosen. The network consists of two hidden layers of 150 LSTM cells fully connected to each other. The network takes as input a vector of 480 values (4×120 features) normalized between -1 and 1 and has an output layer which returns a vector of 37 values per time step (36 classes of characters, see subsection 3.2.2, and the class "blank"). A CTC layer is also introduced to make the link between these outputs values and the text to be recognized. The training of this network was also performed on a training set under the same conditions as the previous one (using the same back-propagation algorithm).

6.4 Text recognition

Once a BLSTM network is trained, for a given input sequence of feature vectors, the output sequence provided by the BLSTM can be analyzed to obtain the recognized text. This operation is called the decoding phase and aims at determining the labelling $\hat{l}$ that corresponds to the one with the highest conditional probability:

$$\hat{l} = \underset{l}{\operatorname{argmax}}\, p(l|\{X^0, X^1, \cdots, X^{p-1}\}) \tag{6.2}$$

where l is a possible labelling and $\{X^0, X^1, \cdots, X^{p-1}\}$ is the input sequence.

For our decoding problem, we assume, as in [GLF+09], that the best labelling corresponds to the most probable path within the sequence of the BLSTM outputs. This path is simply determined as the concatenation of the most active outputs (namely the classes with the highest conditional probability) at each time step. Given this best path, that can contain character or "blank" labels, the resulting sequence of labels can be analyzed to determine the recognized text or sequence of characters. This is done in two steps: first by removing repeated successive labels, then by removing "blanks".

For instance, for a sequence of labels:

$$\hat{l} = c_1 c_1 c_1 B B B c_2 B c_2 c_2 B B c_3 c_3 c_3 c_3 \tag{6.3}$$

where c_1, c_2, and c_3 are classes of characters, and B is the class "blank", the recognized text is:

$$\hat{C} = c_1 c_2 c_2 c_3 \tag{6.4}$$

The order of the two steps is important to preserve letters that appear twice in a text. If "blanks" were removed before repeated characters, the inappropriate text $\hat{C} = c_1 c_2 c_3$ would be obtained.

Fig. 6.8 illustrates an example of recognized text (resulting of the decoding phase) and shows its corresponding BLSTM where each recognized character is represented with a peak.

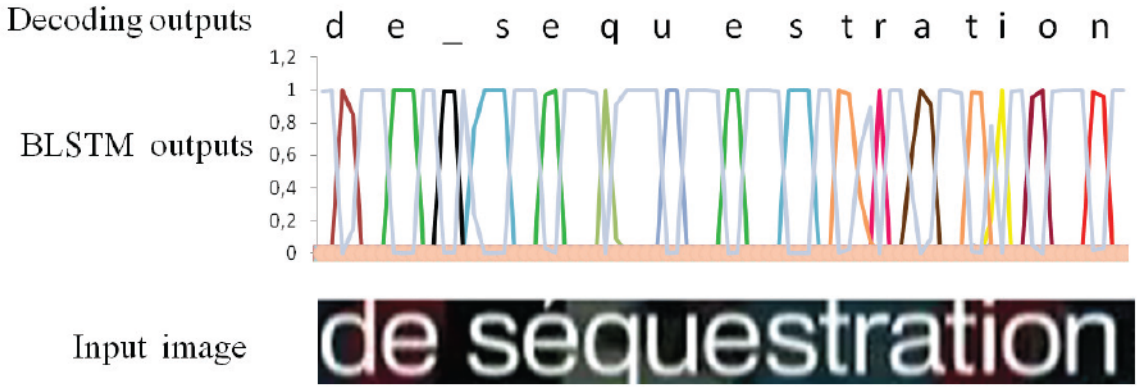

Figure 6.8: Example of a recognized text: Each class is represented with a color; the label "_" represents the class "space" and the gray curve corresponds to the class "blank".

6.5 Experimental results

We focus now on the evaluation of the proposed approach on our two datasets: the "caption" texts (DatasetI) and the "scene" texts (DatasetII). First, some experiments performed to compare learnt features-based representations to hand-crafted-based ones are presented. Then our complete OCR system is evaluated on TextDatasetI and TextDatasetII and compared to other state-of-the-art methods and to our previous approaches.

6.5.1 Contributions of the proposed learnt features

In order to evaluate the influence of our features, learnt with a neural-based model, on the text recognition process, we propose to compare the performance of our approach obtained with these features to the performance achieved using some hand-crafted ones.

Graves *et al.* [GLF+09] have addressed the problem of handwriting text recognition and proposed to represent texts by means of sequences of some geometrical hand-crafted features. Their method is applied to binary images (a black text on a white background) and consists in extracting nine features per column (*i.e.*, vertical line in the image). Each image is thus represented by a sequence of length w, the width of the image, of nine-dimensional vectors. The nine values (*i.e.*, geometrical features) of each vector, computed per column, correspond to:

- F1: the mean value of pixels intensities,
- F2: the position of the center of gravity of the pixels,
- F3: the second order vertical moment of the center of gravity,
- F4: the position of the uppermost black pixel,
- F5: the position of the lowermost black pixel,
- F6: the number of black pixels between F4 and F5,
- F7: the number of transitions from black to white between F4 and F5,
- F8: the rate of change of F4 with respect to its neighborhood columns,
- F9: the rate of change of F5 with respect to its neighborhood columns.

We choose to compare our learnt features to those nine features. We therefore perform two experiments: one that tests and evaluates the performance of our proposed connectionist approach when the BLSTM network is fed with sequences of the hand-designed features, and the second that evaluates the approach when the BLSTM network is fed with sequences of the learnt features. In both experiments, a phase of training of the BLSTM network is first established on training sets (one for the "caption" texts and one for the "scene" texts); then resulting networks are used to evaluate the performance of the method on TextDatasetI and TextDatasetII.

In the first experiment, a preliminary step is performed in order to binarize text images using the fuzzy map generated and described in subsection 4.2.1. The nine geometrical features presented above are then extracted for each column in the image, producing a sequence of length w containing vectors of nine features. Fig. 6.9 illustrates the generation of these geometrical features.

In the second experiment, sequences of learnt features are generated for each image as described in section 6.2. Using the trained CRConvNet (*cf.* subsection 4.3.2), learnt features are extracted from the multi-scale sliding windows. This extraction is done per scanning step, producing a sequence of length $w/step - 1$ (with $step = h/8$) containing vectors of 200 features for TextDatasetI, or 480 features for TextDatasetII.

Using resulting features, two BLSTM networks are trained for each dataset. The first BLSTM is fed with the sequence of geometrical features and takes 9 input values

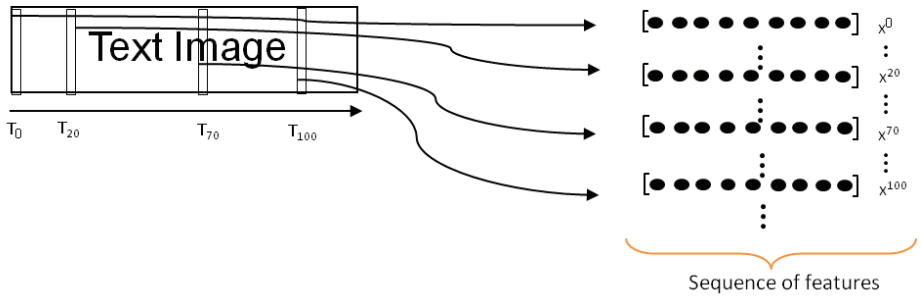

Figure 6.9: Computation of the geometrical hand-crafted features.

per time step, while the second is fed with the sequence of learnt features and takes 200 or 480 input values per time step. For DatasetI, the training phases are performed on a set of 1,399 text images extracted from TextTrainDatasetI, and for DatasetII, they are performed on a set of 1,146 images extracted from TextTrainDatasetII. Trained networks are then evaluated on TextDatasetI and TextDatasetII.

Table 6.1: Usefulness of learnt features: Reported results correspond to the character recognition rate.

Used features	**TextDatasetI**	**TextDatasetII**
Geometrical hand-crafted features	92.73%	21.87%
Learnt features	**97.35%**	**56.44%**

Performances are presented in table 6.1. Results achieved on TextDatasetI show that both BLSTM networks (trained with hand-crafted and learnt features) obtain high performance, above 90% of character recognition rate. Nevertheless, for our application, the learnt features-based process outperforms the hand-crafted-based one by about +5% of characters correctly recognized. Regarding TextDatasetII, table 6.1 shows an important difference in performance between the two types of features (about 22% versus 56% of character recognition rate). The two experiments demonstrate that the proposed learnt features are more adapted than geometrical ones for both types of texts, particularly for "scene" ones where characters can be of various fonts and shapes and can be captured on complex backgrounds making the binarization step really difficult.

6.5.2 Performance of the recurrent connectionist OCR

Using the BLSTM networks trained with learnt features, we focus now on the evaluation of our complete connectionist approach and its comparison to state-of-the-art methods.

Table 6.2 provides the results together with figures obtained with two other existing methods and two commercial OCR engines (ABBYY FineReader OCR and Tesseract OCR).

Table 6.2: Comparison of the proposed scheme to state-of-the-art methods and commercial OCR engines: RR means Recognition Rate (For TextDatasetII, only word RRs are reported because they are the only performance evaluated for other existing methods).

OCR system	**TextDatasetI**		**TextDatasetII**	
			Exp1	**Exp2**
	Character RR	Word RR	Word RR	Word RR
Connectionist OCR	**97.35%**	**87.20%**	**22.81%**	**48.83%**
Saïdane *et al.* [SGD09]	-	-	54.13%	-
Wang *et al.* [WB10]	-	-	-	59.20%
ABBYY FineReader OCR	95.03%	87.70%	-	42.80%
Tesseract OCR	88.57%	70.01%	-	35.00%

Experiments carried out on TextDatasetI show that our connectionist approach achieves an outstanding character recognition rate of 97.35% corresponding to a word recognition rate above 87%. This method obtains an outstanding word recognition rate and a higher character recognition rate than the commercial OCRs. These results highlight the interest of the proposed connectionist recurrent classification model. Regarding remaining errors (*i.e.*, the 2.65% of wrongly recognized characters), the experiments showed that most of them are related to missing spaces between words, leading to recognition errors for two consecutive words, which explains the 87.20% of word recognition rate (lower than the one obtained with ABBYY FineReader OCR). This fact (*i.e.*, that ABBYY achieves a slightly better word recognition rate with +0.5%) can also be justified by the use of a dictionary in this OCR. Comparison with commercial OCR engines on TextDatasetII shows that though our approach presents an overfitting problem, it still outperforms commercial OCR engines on "scene" texts by over +6% of word recognition rate.

For the same reasons as those explained in subsection 4.6.2, comparison to existing methods was performed only on TextDatasetII. The two same experiments as those presented in subsection 4.6.2 (Exp1, Exp2) are performed to evaluate our approach in the same conditions in order to provide meaningful comparisons. Let us just remind that Exp1 (resp. Exp2) corresponds to the experimentation with a set of 901 images extracted from TextDatasetII (resp. a set of 1,065 images extracted from TextDatasetII and using a dictionary). As shown in table 6.2, the connectionist method achieves unsatisfactory results on "scene" texts with poor word recognition rates, lower than other methods. This performance can be explained by an overfitting problem that was noticed during the training phase since the performance on the test

set corresponds to a character recognition rate of about 98% on the training examples.

Our connectionist approach was also compared to our two previous proposed scheme, *i.e.*, segmentation-based and our first segmentation-free ones. Table 6.3 presents the results of this comparison.

Experiments carried out on TextDatasetI show that though this connectionist method does not incorporate any linguistic knowledge (no language model and no dictionary), it outperforms both our segmentation-based and first segmentation-free approaches (that integrate these two linguistic components) by about respectively 2% and 4% of character recognition rate. This proves the ability of this method to avoid the character segmentation step and at the same time to obtain good performance. The fact that the word recognition rate of this method is slightly lower than the one obtained with the proposed segmentation-based approach can be explained by the missing space errors described before, and also the use of a dictionary in the latter. Regarding experiments on full TextDatasetII, as explained before for Exp1 and Exp2, we notice poor performance corresponding to lower character and word recognition rates than those of other proposed OCRs. Besides the overfitting problem, these results can be explained by several facts:

- First, the absence of the language model that has permitted to improve the word recognition rate by +21% in the case of the first segmentation-free OCR. Typically, though the BLSTM model incorporates some linguistic context (via the past and future contexts), this information remains poor compared to the one provided by a language model trained on an important corpus.

- Second, the absence of any dictionary that allows to remove some errors. Exp2 presented in table 6.2 demonstrates the utility of a dictionary that enables our approach to achieve a far better word recognition rate.

- Third, the absence of the WCConvNet whose task, in chapter 5, consisted in classifying windows into valid characters or "garbage". Indeed, when scanning "scene" texts (that often present serious distortions), several misaligned windows that can be confused with valid characters are represented by feature vectors similar to those of valid characters. When these confusions are presented to the BLSTM network without any filtering step, several ambiguities can thus be produced leading to poor recognition performance. One solution to this issue could be the incorporation of the WCConvNet features in our approach.

6.6 Conclusion

In this chapter, we have presented a connectionist approach specifically designed for the recognition of texts in images or videos. Text images are first scanned at various scales. Resulting sliding windows are then used to generate vectors of features

Table 6.3: Comparison of the connectionist OCR to previous proposed ones presented in chapters 4 and 5: RR means Recognition Rate.

OCR system	**TextDatasetI**		**TextDatasetII**	
	Character RR	Word RR	Character RR	Word RR
Connectionist OCR	**97.35%**	**87.20%**	**56.44%**	**18.60%**
Segmentation-free OCR	93.55%	81.32%	70.33%	46.72%
Segmentation-based OCR	95.33%	87.83%	65.33%	41.19%

learnt with a ConvNet. Combining a particular recurrent neural network—namely a BLSTM—and a connectionist temporal classification, a classification model is performed to deal with generated sequences of learnt features and to directly learn to recognize texts.

Our experiments have first compared learnt features to hand-crafted ones and shown that proposed ones yield the best performance both on "caption" and "scene" texts. These results emphasize one contribution of this method, which lies in the novel representation of text images generated with features learnt with a neural-based model.

Our complete OCR was also evaluated on a "caption" texts dataset and obtained promising results (exceeding 97% of characters and 87% of words correctly recognized without any linguistic knowledge, *i.e.*, neither a language model nor a dictionary), outperforming state-of-the-art methods and commercial OCRs. This performance demonstrates first the ability of the proposed method to make use of learnt features dependencies (due to the BLSTM's recurrent connections), and secondly its capacity to avoid the difficult character segmentation step by modeling (via the CTC) the link between non-segmented sequences of inputs and the recognized text. Experiments that have been carried out on a "scene" texts dataset showed a problem of generalization and highlighted the usefulness of the linguistic knowledge absent in this system. A remedy could be an integration of a language model, as for the approaches described in chapters 4 and 5.

Chapter 7

Conclusions and perspectives

This thesis addresses the issue of text recognition in multimedia documents. The aim of this task is to extract the textual clues present in images and videos in order to provide useful information to facilitate content analysis and understanding.

In this work, we have focused on the problem of text recognition in images and videos and proposed to address all the steps involved in this task. Our goal was to design novel efficient OCR systems able to deal with both "caption" and "scene" texts and to cope with different challenges including the variability of sizes, colors and fonts, the complexity of backgrounds, the difficult acquisition conditions, etc. Three OCR systems were proposed and evaluated in this thesis. The key principles of these systems and their performance achieved on two types of texts were summarized in the conclusions of chapters 4, 5, and 6.

Beyond these three OCRs that can be considered as three global and "practical" contributions of this work, we highlight here our main methodological contributions to the field of text recognition in multimedia documents.

A first contribution lies in the design of a character segmentation method that computes nonlinear separations well adapted to the local morphology of text images. The resulting segmentations enable hence a precise extraction of characters and lead to a better character recognition. In particular, the proposed method provides reliable solutions to handle touching or close characters and to deal with texts with complex backgrounds.

We have also presented an original multi-scale scanning scheme, avoiding any character segmentation step, and permitting to recognize characters directly from the text image. This scheme consists in using sliding windows of different sizes (proportional to the image height) that are moved at regular and close positions through the text image aiming at covering all character sizes and positions. Nonlinear borders are also computed for each window in order to obtain well-framed characters.

In addition to the multi-scale scanning scheme that permits to recognize characters at their appropriate position and scale in the text image, we have proposed to deal with the obtained sliding windows with a graph model able to represent spatial

constraints between windows and to manage their classification results. Due to this graph, the need for the crucial segmentation step is completely removed and full texts are recognized as sequences of characters corresponding to the most probable paths within the graph.

Furthermore, we have proposed a novel representation of text images. In contrast to the main methodology that relies on hand-crafted features, this proposal consists in representing sliding windows with features learnt with a neural-based model permitting to extract from color character images the relevant features summarizing their contents. We have shown that these features are more robust to noise, font variability and complex background than hand-crafted ones. By that way, text images can are be represented by sequence of features vectors used to feed a specific connectionist neural model (that combines a particular recurrent network—a bidirectional long-short term memory, BLSTM—and a connectionist classification—a connectionist temporal classification, CTC) able to classify these sequences and to recognize texts.

This study has also demonstrated the contribution of linguistic knowledge (*i.e.*, a language model and a dictionary) in the task of text recognition. Indeed, we have proved that when introducing the probabilities estimated by a character n-gram model, several errors related to the local character-by-character recognition can be removed taking into account the lexical context. The use of a dictionary permits also to reduce some errors, and hence improve recognition performance.

Through this work, some general principles have also emerged from our study and the carried-out experimentations. For "caption" texts, both segmentation-based and segmentation-free approaches are able to obtain outstanding performance. Regarding "scene" texts, this thesis has proved that if the segmentation-based approach obtain results equivalent to previously published methods, the segmentation-free one outperforms both state-of-the-art and commercial solutions, achieving promising results.

At the time of writing the present document, part of the work of this thesis has been implemented into two demonstration softwares, developed in Orange Labs and applied to broadcasts of several French TV channels (including TF1, France2, France3, EuroSport, M6, and BFM TV):

- An indexing engine dedicated to broadcast news: this engine detects and recognizes texts embedded in digital news videos allowing the extraction of titles of reports, names of persons, dates and places. Fig. 7.1 shows the demonstration interface presenting the results of this designed engine.

- A live-TV real-time text recognition engine: this engine combines the results of our segmentation-based OCR system and transcription obtained by a speech-to-text system to analyze a TV broadcast and extract keywords grouped into four categories: persons, terms, geography and entity. This engine is applied continuously to live-TV broadcasts including talk shows, game shows, news, etc. Fig. 7.2 illustrates the interface of this engine.

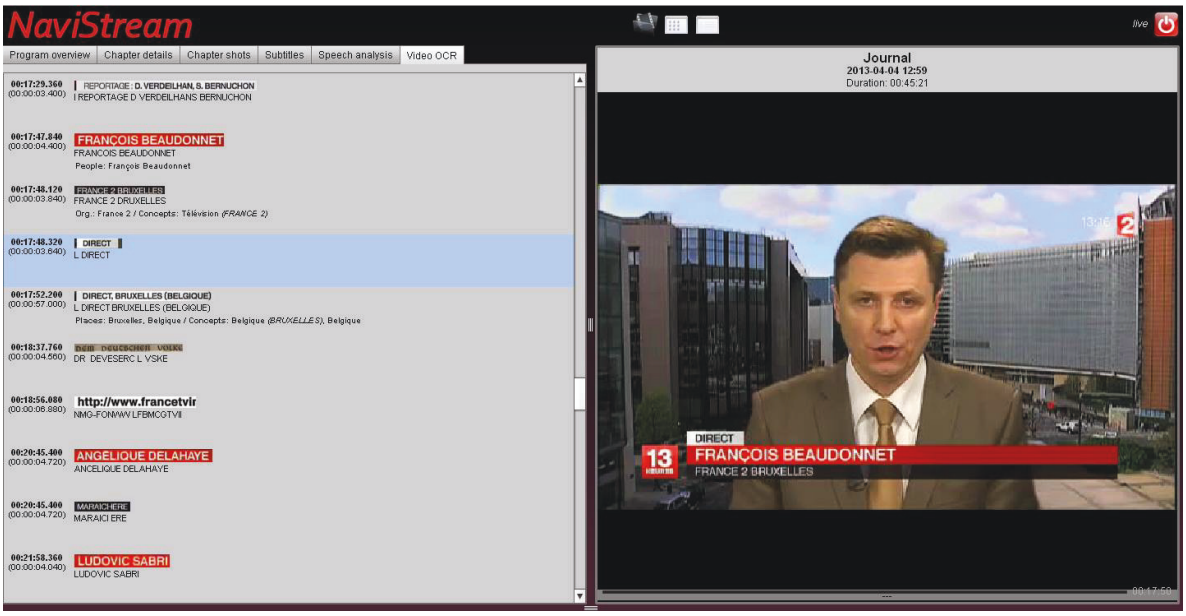

Figure 7.1: The broadcast news indexing engine: on the right, the video frame is displayed while on the left the extracted text images and their recognition results are provided.

As future extensions of this work, several research areas can be investigated.

Some of these areas are directly related to our thesis work and can be considered as next steps to do:

1. The incorporation of linguistic knowledge—namely a language model—in our connectionist approach: this can be done by introducing the probabilities of sequences of characters estimated with a n-gram model in the decoding phase of the BLSTM outputs. Hence, several errors should be removed and missed characters be recovered.

2. The integration of character confusion matrices: confusion matrices are tables that permit to evaluate the performance of a given recognizer by estimating for each class its probability to be well-recognized and those to be confused with each of the other classes. These matrices provide an important information that was not exploited in this study and that can be incorporated in our OCR systems to reduce ambiguities of recognition and thus improve performance. These confusion probabilities could be integrated when calculating word or path scores in our OCR systems.

3. The introduction of upper-case and lower-case letters: in our experiments, we have not distinguished between upper-case and lower-case letters. Since this differentiation can be useful to identify named entities, it is hence interesting

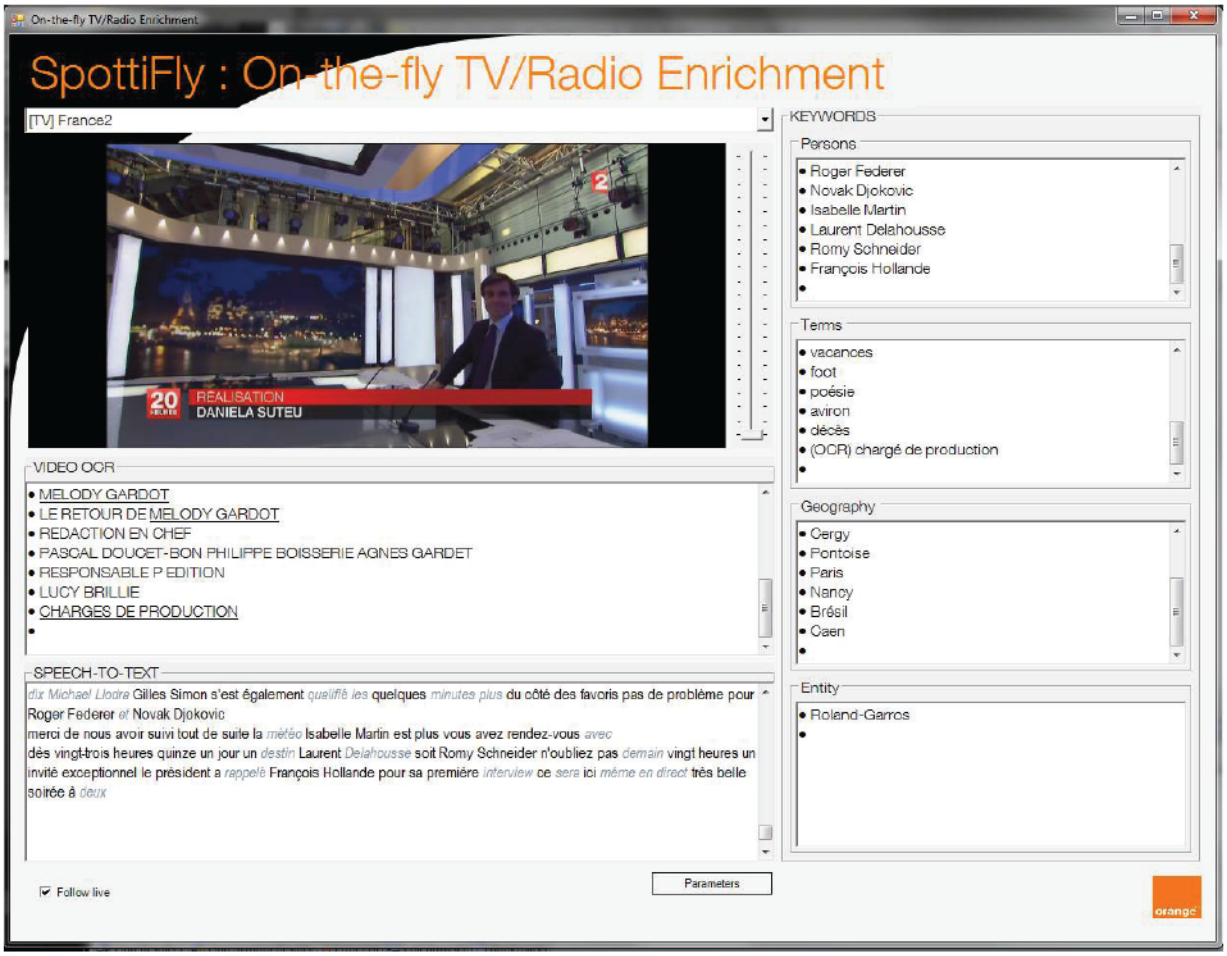

Figure 7.2: The live-TV real-time text recognition engine: the video frames are presented on the left at the top, the recognized texts (VIDEO OCR) on the left in the middle, the transcripts (SPEECH-TO-TEXT) on the left at the bottom, and the extracted keywords (KEYWORDS) on the right.

to train our ConvNets to recognize these classes and to integrate them in our OCR systems.

We can also propose some long-term prospects that might help to improve performance:

1. The super-resolution of text images: the super-resolution is a technique that consists in increasing the resolution of text images in order to obtain images of better quality where texts are easier to recognize. Several methods were proposed in the literature [LD00, WD02, BJ08]. This preprocessing can hence be studied to perform a robust method able to produce images with clearer separations between characters and thus to improve the performance of the

character segmentation step. Particularly, a method specifically adapted to "scene" texts could be designed to reduce segmentation errors on this type of texts.

2. Deep learning models for character recognition: though our recognizer relies on a deep learning model, *i.e.*, a ConvNet, and obtains good results, the use of other deep learning models, such as Restricted Boltzmann Machines (RBM) [LRB08], can be studied and compared to our recognizer in order to determine the most appropriate one for the classification task.

3. Unsupervised learning techniques, such as autoencoders or sparse encoders: these techniques, that has recently showed a great ability to resume image contents [RPCL07]. So it might be particularly interesting for our problem of text images representation. One possible improvement of our connectionist method lies in the use of unsupervised techniques to produce relevant representations to feed a recurrent connectionist model whose task is to recognize the encoded sequences of characters.

Apart from these prospects, the OCR systems developed in this work can find practical applications in several domains. For instance, in addition to the indexing engines, the proposed systems can serve to enhance a video teaching service by recognizing texts embedded in filmed slides, or help visually impaired people by reading using audio devices.

Bibliography

[BBdSM02] L.R. Bahl, P.F. Brown, P.V. de Souza, and R.L. Mercer. A tree-based statistical language model for natural language speech recognition. *Transactions on Acoustics, Speech and Signal Processing*, 37(7):1001–1008, 2002. 54

[BJ08] J. Banerjee and CV Jawahar. Super-resolution of text images using edge-directed tangent field. In *IAPR International Workshop on Document Analysis Systems*, pages 76–83. IEEE, 2008. xxvii, 15, 102

[BMW+11] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Sequential deep learning for human action recognition. *Human Behavior Unterstanding*, pages 29–39, 2011. 84

[BMW+12] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Spatio-temporal convolutional sparse auto-encoder for sequence classification. *British Machine Vision Conference*, 2012. 84

[CCC+11] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D.J. Wu, and A.Y. Ng. Text detection and character recognition in scene images with unsupervised feature learning. In *International Conference on Document Analysis and Recognition*, pages 440–445. IEEE, 2011. 27

[CG96] S.F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 310–318. Association for Computational Linguistics, 1996. 55, 56

[CGR05] T.B. Chen, D. Ghosh, and S. Ranganath. Video-text extraction and recognition. In *IEEE Region 10 Conference*, volume 1, pages 319–322, 2005. x, 7, 25

[CL02] R.G. Casey and E. Lecolinet. A survey of methods and strategies in character segmentation. *Pattern Analysis and Machine Intelligence*, 18(7):690–706, 2002. 7, 17, 47

[CL11] C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27, 2011. 59

[COB02] D. Chen, J.-M. Odobez, and H. Bourlard. Text segmentation and recognition in complex background based on Markov random field. In *International Conference on Pattern Recognition*, volume 4, pages 227–230. IEEE, 2002. 12

[COB04] D. Chen, J.-M. Odobez, and H. Bourlard. Text detection and recognition in images and video frames. *Pattern Recognition Letters*, 37(3):595–608, 2004. 7

[CSL02] H.S. Chang, S. Sull, and S.U. Lee. Efficient video indexing scheme for content-based retrieval. *Transactions on Circuits and Systems for Video Technology*, 9(8):1269–1279, 2002. 1

[CY04] X. Chen and A.L. Yuille. Detecting and reading text in natural scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 366–373. IEEE, 2004. 9

[CZKA02] Y.L. Chang, W. Zeng, I. Kamel, and R. Alonso. Integrated image and speech analysis for content-based video indexing. In *International Conference on Multimedia Computing and Systems*, pages 306–313, 2002. xvii, 1

[DAS01] C. Dorai, H. Aradhye, and J.-C. Shim. End-to-end video text recognition for multimedia content analysis. In *International Conference on Multimedia and Expo*, pages 601–604. IEEE Computer Society, 2001. 27, 58

[DG08] M. Delakis and C. Garcia. Text detection with convolutional neural networks. In *International Conference on Computer Vision Theory and Applications*, volume 2, pages 290–294, 2008. 7, 32

[DLR+77] A.P. Dempster, N.M. Laird, D.B. Rubin, et al. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977. 43

[EGS11] K. Elagouni, C. Garcia, and P. Sébillot. A comprehensive neural-based approach for text recognition in videos using natural language processing. In *International Conference on Multimedia Retrieval*, 2011. 8

[Fat07] R. Fattal. Image upsampling via imposed edge statistics. *ACM Transactions on Graphics*, 26(3):95, 2007. 13

[FE73] M.A. Fischler and R.A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers,*, 100(1):67–92, 1973. 28

[FF09] X. Fan and G. Fan. Graphical models for joint segmentation and recognition of license plate characters. *IEEE Signal Processing Letters,*, 16(1):10–13, 2009. 23

[FREM04] S. Farsiu, D. Robinson, M. Elad, and P. Milanfar. Advances and challenges in super-resolution. *International Journal of Imaging Systems and Technology*, 14(2):47–57, 2004. 13

[GD04] C. Garcia and M. Delakis. Convolutional Face Finder: A neural architecture for fast and robust face detection. *Pattern Analysis and Machine Intelligence*, 26(11):1408–1423, 2004. 50

[GESF04] J. Gllavata, R. Ewerth, T. Stefi, and B. Freisleben. Unsupervised text segmentation using color and wavelet features. *Image and Video Retrieval*, pages 1967–1967, 2004. 11

[GFGS06] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *International Conference on Machine Learning*, pages 369–376. ACM, 2006. 90

[GG84] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984. 12

[GLF+09] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *Pattern Analysis and Machine Intelligence*, 31(5):855–868, 2009. xxii, 86, 90, 91, 92, 94

[Gra08] A. Graves. *Supervised sequence labelling with recurrent neural network*. PhD thesis, Technische universität München, 2008. xiii, 89

[GS05] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005. 89

[GSS03] F.A. Gers, N.N. Schraudolph, and J. Schmidhuber. Learning precise timing with LSTM recurrent networks. *The Journal of Machine Learning Research*, 3:115–143, 2003. 88

[HKA10] M. Halima, H. Karray, and A. Alimi. A comprehensive method for Arabic video text detection, localization, extraction and recognition. *Advances in Multimedia Information Processing-PCM*, pages 648–659, 2010. 25

[HMZ09] X. Huang, H. Ma, and H. Zhang. A new video text extraction approach. In *International Conference on Multimedia and Expo*, pages 650–653. IEEE, 2009. 20

[HS97] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 88

[HYZ02] X.S. Hua, P. Yin, and H.J. Zhang. Efficient video text recognition using multiple frame integration. In *International Conference on Image Processing*, volume 2, pages 397–400, 2002. x, 7, 15, 16

[ITK10] M. Iwamura, T. Tsuji, and K. Kise. Memory-based recognition of camera-captured characters. In *IAPR International Workshop on Document Analysis Systems*, pages 89–96. ACM, 2010. 25

[JHF+05] S. Jun, Y. Hotta, K. Fujimoto, Y. Katsuyama, and S. Naoi. Grayscale feature combination in recognition based segmentation for degraded text string recognition. *International Workshop on Camera-Based Document Analysis and Recognition*, 2005. 22

[JIKKJ04] K. Jung, K. In Kim, and A. K Jain. Text information extraction in images and video: a survey. *Pattern Recognition Letters*, 37(5):977–997, 2004. 2

[Jol05] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2005. 22

[JSS99] M.C. Jung, Y.C. Shin, and S.N. Srihari. Machine printed character segmentation method using side profiles. In *IEEE International Conference on Systems, Man, and Cybernetics*, volume 6, pages 863–867. IEEE, 1999. 25

[JSVR05] C. Jacobs, P.-Y. Simard, P. Viola, and J. Rinker. Text recognition of low-resolution document images. In *International Conference on Document Analysis and Recognition*, pages 695–699. IEEE, 2005. 27

[Kat87] S. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35(3):400–401, 1987. 56

[KGJV83] S. Kirkpatrick, C.D. Gelatt Jr, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983. 12

[KHE05] S. Kopf, T. Haenselmann, and W. Effelsberg. *Robust character recognition in low-resolution images and videos*. Universität Mannheim/Institüt für Informatik, 2005. x, 7, 18, 19, 24, 28

[KN95] R. Kneser and H. Ney. Improved backing-off for m-gram language modeling. In *International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 181–184. IEEE, 1995. 56

[KSIA04] Y. Kusachi, A. Suzuki, N. Ito, and K. Arakawa. Kanji recognition in scene images without detection of text fields - robust against variation of viewpoint, contrast, and background texture. In *International Conference on Pattern Recognition*, volume 1, pages 457–460, 2004. 22, 28

[LB95] Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, pages 255–258, 1995. xix, 13, 49, 86

[LBBH98] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. of the IEEE*, 86(11):2278–2324, 1998. 50

[LBWX10] M. Li, M. Bai, C. Wang, and B. Xiao. Conditional random field for text segmentation from images with complex background. *Pattern Recognition Letters*, 31(14):2295–2308, 2010. 12

[LD99] H. Li and D. Doermann. Text enhancement in digital video using multiple frame integration. In *International Conference on Multimedia*, pages 19–22. ACM, 1999. 15

[LD00] H. Li and D. Doermann. Superresolution-based enhancement of text in digital video. In *International Conference on Pattern Recognition*, volume 1, pages 847–850. IEEE, 2000. ix, 14, 102

[LDK00] H. Li, D. Doermann, and O. Kia. Automatic text detection and tracking in digital video. *IEEE Transactions on Image Processing*, 9(1):147–156, 2000. 7

[LKF10] Y. LeCun, K. Kavukcuoglu, and C. Farabet. Convolutional networks and applications in vision. In *IEEE International Symposium on Circuits and Systems*, pages 253–256. IEEE, 2010. 50

[LLP02] S.W. Lee, D.J. Lee, and H.S. Park. A new methodology for gray-scale character segmentation and recognition. *Pattern Analysis and Machine Intelligence*, 18(10):1045–1050, 2002. 45

[LLPN09] H. Lee, Y. Largman, P. Pham, and A. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. *Advances in neural information processing systems*, 22:1096–1104, 2009. 84

[LPM07] J. Lim, J. Park, and G.G. Medioni. Text segmentation in color images using tensor voting. *Image and Vision Computing*, 25(5):671–685, 2007. 7

[LPS+03] S.M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young. Icdar 2003 robust reading competitions. *International Journal on Document Analysis and Recognition*, 2:682–687, 2003. 5, 35

[LRB08] N. Le Roux and Y. Bengio. Representational power of restricted boltzmann machines and deep belief networks. *Neural Computation*, 20(6):1631–1649, 2008. 103

[LS96] R. Lienhart and F. Stuber. Automatic text recognition in digital videos. *Image and Video Processing*, pages 2666–2675, 1996. xvii, 1, 7

[LSA94] S. Liang, M. Shridhar, and M. Ahmadi. Segmentation of touching characters in printed document recognition. *Pattern Recognition*, 27(6):825–840, 1994. 17

[LW02] R. Lienhart and A. Wernicke. Localizing and segmenting text in images and videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(4):256–268, 2002. x, 16

[LWC+10] H.T. Lue, M.G. Wen, H.Y. Cheng, K.C. Fan, C.W. Lin, and C.C. Yu. A novel character segmentation method for text images captured by cameras. *ETRI journal*, 32(5):729–739, 2010. x, 21

[MAJ11] A. Mishra, K. Alahari, and CV Jawahar. An MRF model for binarization of natural scene text. In *International Conference on Document Analysis and Recognition*, pages 11–16, 2011. ix, 10, 12

[MBPLM08] F. Manerba, J. Benois-Pineau, R. Leonardi, and B. Mansencal. Multiple moving object detection for fast video content description in compressed domain. *Journal on Advances in Signal Processing*, 2008:5, 2008. xvii, 1

[MGS05] S. Marinai, M. Gori, and G. Soda. Artificial neural networks for document analysis and recognition. *Pattern Analysis and Machine Intelligence,*, 27(1):23–35, 2005. 27

[MS99] C.D. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT press, 1999. 29, 54

[MS01] S.K. Mitra and G.L. Sicuranza. *Nonlinear image processing*. Academic Press, 2001. 14

[MS06] K. Malczewski and R. Stasinski. Optical character recognition of low resolution text sequences from hand-held device supported by super-resolution. In *International Symposium on Multimedia Signal Processing and Communications*, pages 61–64. IEEE, 2006. 14

[MTG06] C. Mancas-Thillou and B. Gosselin. Character segmentation-by-recognition using log-Gabor filters. In *International Conference on Pattern Recognition*, volume 2, pages 901–904, 2006. 7, 22

[MTM05] C. Mancas-Thillou and M. Mirmehdi. Super-resolution text using the Teager filter. In *International Workshop on Camera-Based Document Analysis and Recognition*, pages 10–16. Citeseer, 2005. x, 14, 15

[MZJ+07] G. Miao, G. Zhu, S. Jiang, Q. Huang, X. Changsheng, and W. Gao. A real-time score detection and recognition approach for broadcast basketball video. In *International Conference on Multimedia and Expo*, pages 1691–1694. IEEE, 2007. 18, 28

[NGP11] K. Ntirogiannis, B. Gatos, and I. Pratikakis. Binarization of textual content in video frames. In *International Conference on Document Analysis and Recognition*, pages 673–677, 2011. 10

[Nib85] W. Niblack. *An introduction to digital image processing*. Strandberg Publishing Company, 1985. ix, 9, 10

[NIOA05] K. Negishi, M. Iwamura, S. Omachi, and H. Aso. Isolated character recognition by searching features in scene images. In *International Workshop on Camera-Based Document Analysis and Recognition*, pages 140–147, 2005. 23, 25

[Nov66] P.S. Novikov. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet Physics-Doklady*, volume 10, 1966. 38

[ODT+09] A. Ohkura, D. Deguchi, T. Takahashi, I. Ide, and H. Murase. Low-resolution character recognition by video-based super-resolution. In *International Conference on Document Analysis and Recognition*, pages 191–195. IEEE, 2009. x, 26

[Ots75] N. Otsu. A threshold selection method from gray-level histograms. *Automatica*, 11:285–296, 1975. ix, 9, 10

[PLK+10] J. Park, G. Lee, E. Kim, J. Lim, S. Kim, H. Yang, M. Lee, and S. Hwang. Automatic detection and recognition of Korean text in outdoor signboard images. *Pattern Recognition Letters*, 31(12):1728–1739, 2010. 26

[PSMT97] A.J. Patti, M.I. Sezan, and A. Murat Tekalp. Superresolution video reconstruction with arbitrary sampling lattices and non-zero aperture time. *IEEE Transactions on Image Processing,*, 6(8):1064–1076, 1997. 14

[PSST11] T.Q. Phan, P. Shivakumara, B. Su, and C.L. Tan. A gradient vector flow-based method for video character segmentation. In *International Conference on Document Analysis and Recognition*, pages 1024–1028. IEEE, 2011. x, 7, 19

[PST10] T.Q. Phan, P. Shivakumara, and C.L. Tan. A skeleton-based method for multi-oriented video text detection. In *IAPR International Workshop on Document Analysis Systems*, pages 271–278. ACM, 2010. 7

[QLWS07] X. Qian, G. Liu, H. Wang, and R. Su. Text detection, localization, and tracking in compressed video. *Signal Processing: Image Communication*, 22(9):752–768, 2007. 7

[RHW86] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. 48

[Ros58] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958. 48

[RPCL07] M.A. Ranzato, C. Poultney, S. Chopra, and Y. LeCun. Efficient learning of sparse representations with an energy-based model. *Advances in Neural Information Processing Systems*, 19:1137–1144, 2007. xxvii, 103

[SBS+11] P. Shivakumara, S. Bhowmick, B. Su, C.L. Tan, and U. Pal. A new gradient based character segmentation method for video text recognition. In *International Conference on Document Analysis and Recognition*, pages 126–130, 2011. x, 18

[SCS09] T. Som, D. Can, and M. Saraclar. HMM-based sliding video text recognition for Turkish broadcast news. In *International Symposium on Computer and Information Sciences*, pages 475–479, 2009. 27, 29

[SG07a] Z. Saidane and C. Garcia. Automatic scene text recognition using a convolutional neural network. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 100–106, 2007. 7, 22, 27, 50

[SG07b] Z. Saidane and C. Garcia. Robust binarization for video text recognition. In *International Conference on Document Analysis and Recognition*, volume 2, pages 874–879, 2007. 13

[SG08] Z. Saidane and C. Garcia. An automatic method for video character segmentation. *Image Analysis and Recognition*, pages 557–566, 2008. 21, 22

[SGD09] Z. Saidane, C. Garcia, and J.-L. Dugelay. The image text recognition graph (iTRG). In *International Conference on Multimedia and Expo*, pages 266–269, 2009. 7, 22, 28, 61, 62, 78, 79, 96

[SKH+99] T. Sato, T. Kanade, E.K. Hughes, M.A. Smith, and S. Satoh. Video OCR: indexing digital news libraries by recognition of superimposed captions. *Multimedia Systems*, 7(5):385–395, 1999. 16, 18, 21, 28

[SKHS98] T. Sato, T. Kanade, E.K. Hughes, and M.A. Smith. Video OCR for digital news archive. In *International Workshop on Content-Based Access of Image and Video Database*, pages 52–60. IEEE, 1998. 9, 21

[SLP+08] Y. Song, A. Liu, L. Pang, S. Lin, Y. Zhang, and S. Tang. A novel image text extraction method based on k-means clustering. In *International Conference on Computer and Information Science*, pages 185–190. IEEE, 2008. 11

[SPB12] N. Sharma, U. Pal, and M. Blumenstein. Recent advances in video based document processing: A review. In *IAPR International Workshop on Document Analysis Systems*, pages 63–68, 2012. 8

[SPLT11] P. Shivakumara, T.Q. Phan, S. Lu, and C.L. Tan. Video character recognition through hierarchical classification. In *International Conference on Document Analysis and Recognition,*, pages 131–135. IEEE, 2011. x, 24

[SSHP97] J. Sauvola, T. Seppanen, S. Haapakoski, and M. Pietikainen. Adaptive document binarization. In *International Conference on Document Analysis and Recognition*, volume 1, pages 147–152. IEEE, 1997. ix, 9, 10

[SSJ03] P.Y. Simard, D. Steinkraus, and C.P. John. Best practices for convolutional neural networks applied to visual document analysis. In *International Conference on Document Analysis and Recognition*, volume 2, pages 958–963, 2003. 50

[Sto02] A. Stolcke. SRILM-an extensible language modeling toolkit. In *International Conference on Spoken Language Processing*, volume 3, pages 901–904, 2002. 56

[SW05] C.G.M. Snoek and M. Worring. Multimedia event-based video indexing using time intervals. *IEEE Transactions on Multimedia*, 7(4):638–647, 2005. xvii, 1

[SXWZ12] C. Shi, B. Xiao, C. Wang, and Y. Zhang. Adaptive graph cut based binarization of video text images. In *IAPR International Workshop on Document Analysis Systems*, pages 58–62. IEEE, 2012. 13

[TCJY07] J. Tse, D. Curtis, C. Jones, and E. Yfantis. An OCR-independent character segmentation using shortest-path in grayscale document images. In *International Conference on Machine Learning and Applications*, pages 142–147. IEEE, 2007. 19

[TFG05] C. Thillou, S. Ferreira, and B. Gosselin. An embedded application for degraded text recognition. *EURASIP Journal on applied signal processing*, 2005:2127–2135, 2005. 29

[UMS08] S. Uchida, H. Miyazaki, and H. Sakoe. Mosaicing-by-recognition for video-based text recognition. *Pattern Recognition*, 41(4):1230–1240, 2008. 26

[WB10] K. Wang and S. Belongie. Word spotting in the wild. In *European Conference on Computer Vision*, pages 591–604, 2010. 23, 28, 61, 62, 78, 79, 96

[WD02] C. Wolf and D. Doermann. Binarization of low quality text using a Markov random field model. In *International Conference on Pattern Recognition*, volume 3, pages 160–163. IEEE, 2002. 12, 102

[Wer90] P.J. Werbos. Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990. 49

[WJ04] C. Wolf and J.-M. Jolion. Extraction and recognition of artificial text in multimedia documents. *Pattern Analysis and Applications*, 6(4):309–326, 2004. 10

[WJC02] C. Wolf, J.M. Jolion, and F. Chassaing. Text localization, enhancement and binarization in multimedia documents. In *International Conference on Pattern Recognition*, volume 2, pages 1037–1040. IEEE, 2002. 14

[WJW04] R. Wang, W. Jin, and L. Wu. A novel video caption detection approach using multi-frame integration. In *International Conference on Pattern Recognition*, volume 1, pages 449–452. IEEE, 2004. 16

[WK11] T. Wakahara and K. Kita. Binarization of color character strings in scene images using k-means clustering and support vector machines. In *International Conference on Document Analysis and Recognition*, pages 274–278, 2011. 11

[WLMH09] J.J. Weinman, E. Learned-Miller, and A.R. Hanson. Scene text recognition using similarity and a lexicon with sparse belief propagation. *Pattern Analysis and Machine Intelligence*, 31(10):1733–1746, 2009. 29

[Wu96] X. Wu. YIQ vector quantization in a new color palette architecture. *IEEE Transactions on Image Processing*, 5(2):321–329, 1996. 11

[WZ95] R. J. Williams and D. Zipser. Gradient-based learning algorithms for recurrent networks and their computational complexity. *Backpropagation: Theory, architectures and applications*, pages 433–486, 1995. 88, 92

[Yag91] R.R. Yager. Connectives and quantifiers in fuzzy sets. *Fuzzy sets and systems*, 40(1):39–75, 1991. 44

[YBYK11] Y. Yoon, K.D. Ban, H. Yoon, and J. Kim. Blob extraction based character segmentation method for automatic license plate recognition system. In *International Conference on Systems, Man, and Cybernetics*, pages 2192–2196. IEEE, 2011. x, 20

[YEYT11] T. Yamazoe, M. Etoh, T. Yoshimura, and K. Tsujino. Hypothesis preservation approach to scene text recognition with weighted finite-state transducer. In *International Conference on Document Analysis and Recognition*, pages 359–363, 2011. 8

[YGH04] Q. Ye, W. Gao, and Q. Huang. Automatic text segmentation from complex background. In *International Conference on Image Processing*, volume 5, pages 2905–2908. IEEE, 2004. ix, 12, 13

[YHGZ05] Q. Ye, Q. Huang, W. Gao, and D. Zhao. Fast and robust text detection in images and video frames. *Image and Vision Computing*, 23(6):565–576, 2005. 7

[YPX09] J. Yi, Y. Peng, and J. Xiao. Using multiple frame integration for the text recognition of video. In *International Conference on Document Analysis and Recognition*, pages 71–75, 2009. x, 7, 16

[YW05] M. Yokobayashi and T. Wakahara. Segmentation and recognition of characters in scene images using selective binarization in color space and GAT correlation. In *International Conference on Document Analysis and Recognition*, pages 167–171, 2005. 7, 25

[YW06] M. Yokobayashi and T. Wakahara. Binarization and recognition of degraded characters using a maximum separability axis in color space and GAT correlation. In *International Conference on Pattern Recognition*, volume 2, pages 885–888. IEEE, 2006. 25

[YYGH09] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1794–1801. IEEE, 2009. 84

[ZC03] D.Q. Zhang and S.F. Chang. A Bayesian framework for fusing multiple word knowledge models in videotext recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 528–533, 2003. 8, 24, 28

[ZLT10] Z. Zhou, L. Li, and C.L. Tan. Edge based binarization for video text images. In *International Conference on Pattern Recognition*, pages 133–136, 2010. 10

INSA de RENNES
Service des Formations

AVIS DU JURY SUR LA REPRODUCTION DE LA THESE SOUTENUE

Titre de la thèse:
Combining neural-based approaches and linguistic knowledge for text recognition in multimedia documents

Nom Prénom de l'auteur : ELAGOUNI KHAOULA

Membres du jury :
- Monsieur MÉRIALDO Bernard
- Monsieur LÉZORAY Olivier
- Monsieur MORIN Emmanuel
- Madame SÉBILLOT Pascale
- Monsieur GARCIA Christophe
- Monsieur MAMALET Franck
- Monsieur VIARD-GAUDIN Christian

Président du jury : Christian VIARD-GAUDIN

Date de la soutenance : 28 Mai 2013

Reproduction de la these soutenue

☒ Thèse pouvant être reproduite en l'état
☐ Thèse pouvant être reproduite après corrections suggérées

Fait à Rennes, le 28 Mai 2013

Signature du président de jury

Le Directeur,

M'hamed DRISSI

Résumé

Les travaux de cette thèse portent sur la reconnaissance des indices textuels présents dans des images et des vidéos. Dans ce cadre, nous avons conçu des prototypes d'OCR (*optical character recognition*) capables de reconnaître tant des textes incrustés que des textes de scène acquis n'importe où au sein d'images ou de vidéos (sur des panneaux, des banderoles, des affiches...). Nous nous sommes intéressée à la définition d'approches robustes à la variabilité des textes (fonte, couleur, taille...) et aux conditions d'acquisition (fond complexe, occlusion, luminosité non uniforme, faible résolution...). Plus précisément, nous avons proposé deux types de méthodes dédiées à la reconnaissance de texte :

- une approche fondée sur une segmentation en caractères qui recherche des séparations non linéaires entre les caractères adaptées à la morphologie de ces derniers ;

- deux approches se passant de la segmentation en intégrant un processus de *scanning* multi-échelles ; la première utilise un modèle de graphe pour reconnaître les textes tandis que la seconde intègre un modèle connexionniste récurrent spécifiquement développé pour gérer les contraintes spatiales entre les caractères reconnus.

Outre les originalités liées à chacune des approches, deux contributions supplémentaires de ce travail de thèse résident dans la définition d'une reconnaissance de caractères fondée sur un modèle de classification neuronale et l'intégration de certaines connaissances linguistiques permettant de tirer profit du contexte lexical.

Les différentes méthodes conçues ont été évaluées sur deux bases de documents : une base de textes incrustés dans des vidéos et une base publique de textes de scène (base ICDAR 2003). Les expérimentations menées ont permis de montrer la robustesse des approches et de comparer leurs performances à celles de l'état de l'art, mettant ainsi en évidence les avantages et les limites de chaque méthode.

Mots-clés : Reconnaissance de texte, reconnaissance de caractères, segmentation, réseau de neurones, modèle de langue, *scanning* multi-échelles, modèle de graphe, classification connexionniste

Abstract

This thesis focuses on the recognition of textual clues in images and videos. In this context, OCR (optical character recognition) systems, able to recognize caption texts as well as natural scene texts captured anywhere in the environment have been designed. Novel approaches, robust to text variability (different fonts, colors, sizes, etc.) and acquisition conditions (complex background, non uniform lighting, low resolution, etc.) have been proposed. In particular, two kinds of methods dedicated to text recognition are provided:

- A segmentation-based approach that computes nonlinear separations between characters well adapted to the local morphology of images;

- Two segmentation-free approaches that integrate a multi-scale scanning scheme. The first one relies on a graph model, while the second one uses a particular connectionist recurrent model able to handle spatial constraints between characters.

In addition to the originalities of each approach, two extra contributions of this work lie in the design of a character recognition method based on a neural classification model and the incorporation of some linguistic knowledge that enables to take into account the lexical context.

The proposed OCR systems were tested and evaluated on two datasets: a caption texts video dataset and a natural scene texts dataset (namely the public database ICDAR 2003). Experiments have demonstrated the efficiency of our approaches and have permitted to compare their performances to those of state-of-the-art methods, highlighting their advantages and limits.

Keywords: Text recognition, character recognition, segmentation, neural network, language model, multi-scale scanning, graph model, connectionist classification

N° d'ordre : 13ISAR 13 / D13 - 13

Institut National des Sciences Appliquées de Rennes

20, Avenue des Buttes de Coësmes . CS 14315 . F-35043 Rennes Cedex

Tel : 02 23 23 82 00 – Fax : 02 23 23 83 96