Détection, localisation et typage de texte dans des images de documents hétérogènes par Réseaux de Neurones Profonds

Bastien Moysset

N°d'ordre NNT : 2018LYSEI044

THESE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de
l'INSA LYON

Ecole Doctorale N° ED512
Informatique et Mathématiques de Lyon
(InfoMaths)

Spécialité de doctorat : Informatique

Soutenue publiquement le 28/05/2018 par :
Bastien Moysset

Détection, localisation et typage de texte dans des images de documents hétérogènes par Réseaux de Neurones Profonds

---

Devant le jury composé de :

CORD Matthieu	Professeur , Sorbonne Université	Président du jury
FROMONT, Elisa	Professeur , Université de Rennes 1	Rapporteure
HEUTTE, Laurent	Professeur , Université de Rouen	Rapporteur
COUPRIE, Camille	Docteur , Facebook AI Research	Examinatrice
WOLF, Christian	Maître de Conférences HDR , INSA-LYON	Directeur de thèse
KERMORVANT, Christopher	Docteur , TEKLIA	Co-directeur de thèse

Remerciements

Je tiens, en premier lieu, à remercier chaleureusement Christopher Kermorvant. C'est lui qui, dans un premier temps, m'a motivé et convaincu de devenir ingénieur recherche chez A2iA. C'est aussi lui qui, dans un second temps, m'a convaincu d'entreprendre ce travail de thèse. C'est enfin lui qui est venu m'apporter son encadrement toujours pertinent et sa confiance lors de la fin de cette thèse.

Je suis aussi très reconnaissant envers mon directeur de thèse, Christian Wolf, pour son temps et sa disponibilité, pour son écoute, pour ses conseils. Il a su constamment rester présent et impliqué dans ma thèse malgré la distance. J'ai eu beaucoup de chance de l'avoir pour m'encadrer et l'en remercie.

J'ai été ravi de travailler aux côtés de Jérôme Louradour et de commencer cette thèse sous sa supervision. Il m'a transmis son savoir, ses méthodes, ses techniques et j'ai beaucoup appris à son contact. Je le remercie pour ça.

Je remercie particulièrement les personnes ayant participé au jury de ma thèse pour leurs commentaires avisés et pour avoir accepté d'examiner mes travaux. Elisa Fromont et Laurent Heutte pour avoir accepté de rapporter mon manuscrit et pour leurs remarques de qualité mais aussi les examinateurs Matthieu Cord et Camille Couprie.

L'entreprise A2iA m'a offert le cadre de travail idéal pour réaliser ces travaux. Je suis conscient de la chance qu'a été de travailler dans une entreprise qui encourage la publication d'articles scientifiques et offre autonomie et liberté dans les réalisations tout en proposant des thématiques de recherche très intéressantes et concrètes et l'opportunité de travailler sur des technologies à la pointe de l'innovation.

J'ai en particulier été ravi de travailler aux côtés de scientifiques de valeur au sein de l'équipe recherche d'A2iALab. Merci à Théodore pour l'inspiration et l'exemple qu'il m'a donné. C'est en partie en l'observant que j'ai eu envie d'entreprendre cette thèse. Merci à Ronaldo et Faouzi pour les discussions enrichissantes à travers les années mais aussi à Vu, Xi, Kamel et Ana.

Je remercie le laboratoire Liris pour m'avoir accepté en son sein et à ses membres pour leur disponibilité lors de mes passages à Lyon. Je remercie aussi toutes les personnes, à A2iA ou au Liris, qui m'ont apporté un support technique et administratif de qualité au cours de cette thèse.

Enfin, je tiens à remercier ceux qui en faisant partie de ma vie personnelle, en dehors du bureau, participent à mon équilibre de vie et influencent ainsi favorablement ma vie professionnelle. C'est d'abord grâce à eux que j'ai pu mener ce projet à bien. Merci à ma famille. Merci aux amis parisiens, Arnaud, Damien, Robin, Bonnot, Emma, pour leur soutien au bar'bu ou ailleurs. Merci

à mes partenaires de voyage, Yoan, Flocon, Mathilde, Guillaume, Maxime et Grégoire, pour m'avoir permis de m'évader et de vivre des moments fantastiques. Merci aussi aux copains du rugby pour leur camaraderie et leur bonne humeur.

A tous, merci.

Bastien

Résumé

Lire automatiquement le texte présent dans les documents, qu'ils soient imprimés ou manuscrits, permet de rendre accessibles les informations qu'ils contiennent. Pour réaliser la transcription de pages complètes, la détection et la localisation des lignes de texte est une étape cruciale. Si les méthodes traditionnelles de détection de lignes se basent sur des approches de traitement d'images, elles peinent à généraliser à des jeux de données très hétérogènes.

Pour cela, nous proposons dans cette thèse une approche par réseaux de neurones profonds. Ces techniques ont démontré ces dernières années de bonnes capacités à apprendre des tâches complexes et à généraliser correctement sur des ensembles variés. Comme les caractéristiques d'entrée sont les pixels, aucun pré-traitement spécifique à l'image n'est nécessaire. Ces approches par réseaux de neurones ont été appliquées à de nombreuses applications dont la reconnaissance de l'écriture et la détection des objets dans les scènes naturelles.

Notre objectif est de proposer une méthode permettant de détecter les lignes de texte dans les images de documents en prenant en compte les spécificités de la tâche que sont un nombre de données d'entraînement réduit, un large nombre de petits objets à détecter par image et un besoin de précision des prédictions afin de ne pas détériorer la reconnaissance du contenu des lignes.

Nous avons d'abord proposé une approche de segmentation mono-dimensionnelle des paragraphes de texte en lignes à l'aide d'une technique inspirée des modèles de reconnaissance, où la méthode de classification temporelle connexionniste (CTC) est utilisée pour aligner implicitement les séquences. Cette technique a pour avantage de ne pas nécessiter l'annotation de la position des lignes, seul connaître le nombre de lignes présentes dans le paragraphe est nécessaire.

Ensuite, nous proposons un réseau qui prédit directement les valeurs des coordonnées de la position des boîtes englobant les lignes de texte. L'ajout d'un terme de confiance à ces boîtes hypothèses permet, après seuillage, d'être capable de localiser un nombre variable d'objets. Nous proposons une prédiction locale des objets qui permet de partager les paramètres entre les localisations et, ainsi, de multiplier les exemples d'objets différents vus par chaque prédicteur de boîte lors de l'entraînement. Cela permet de compenser la taille restreinte des jeux de données utilisés. Pour récupérer les informations contextuelles permettant de prendre en compte les informations liées à la structure du document, nous ajoutons, entre les couches convolutionnelles de notre réseau, des couches récurrentes LSTM multi-dimensionnelles.

Nous proposons trois stratégies de reconnaissance pleine page du texte. La détection directe des boîtes englobant les lignes, la détection des coins opposés de ces boîtes et leur appariement et la détection des côtés gauches des lignes de texte. Dans ce dernier cas, c'est le reconnaisseur de texte qui est chargé d'apprendre à identifier la fin de la ligne. Ces stratégies permettent de tenir

compte du besoin important de précision au niveau des prédictions des positions.

Nous montrons, sur la base hétérogène Maurdor, la performance de notre approche pour des documents pouvant être manuscrits et imprimés, écrits en français, en anglais et en arabe et nous nous comparons favorablement à des méthodes issues de l'état de l'art. La visualisation des concepts appris par nos neurones permet de mettre en exergue la capacité de nos couches récurrentes à apporter l'information contextuelle.

Abstract

Being able to automatically read the texts written in documents, both printed and handwritten, makes it possible to access the information they convey. In order to realize full page text transcription, the detection and localization of the text lines is a crucial step. Traditional methods tend to use image processing based approaches, but they hardly generalize to very heterogeneous datasets.

In this thesis, we propose to use a deep neural network based approach. Deep learning techniques have been proven during the last few years able to learn complex tasks and to efficiently generalize on heterogeneous datasets. Because the input features are the raw image pixels, no image specific pre-processing is needed. Neural networks approaches have been applied to various applications including text recognition and object detection in natural scenes.

We aim at proposing a method that enables to detect text lines in document images and that take into account the specificity of the task that are a reduced amount of available training data, a large number of small objects to detect in a single image and the need to precisely locate the objects in order not to deteriorate the recognition of the line content.

We first propose a mono-dimensional segmentation of text paragraphs into lines that uses a technique inspired by the text recognition models. The connexionist temporal classification (CTC) method is used to implicitly align the sequences. The advantage of this technique is that it does not need an explicit line position annotation ; only knowing the number of lines that are present in the dataset is needed.

Then, we propose a neural network that directly predicts the values corresponding to the coordinates of the boxes bounding the text lines. Adding a confidence prediction to these hypothesis boxes enables, after thresholding, to locate a varying number of objects. We propose to predict the objects locally in order to share the network parameters between the locations and to increase the number of different objects that each single box predictor sees during training. This compensates the rather small size of the available datasets. In order to recover the contextual information

that carries knowledge on the document layout, we add multi-dimensional LSTM recurrent layers between the convolutional layers of our networks.

We propose three full page text recognition strategies. The direct detection of the text line bounding boxes, the detection of opposite corners of these boxes and their pairing, and the detection of the left sides of the boxes. In the last scenario, the text recognizer is in charge of learning to identify the end of the line. These three strategies enable to tackle the need of high preciseness of the text line position predictions.

We show on the heterogeneous Maurdor dataset how our methods perform on documents that can be printed or handwritten, in French, English or Arabic and we favourably compare to other state of the art methods. Visualizing the concepts learned by our neurons enables to underline the ability of the recurrent layers to convey the contextual information.

Table des matières

Chapitre 1

Introduction

1.1 Problématique

Dans le monde actuel, l'accumulation et l'utilisation de données sont prégnantes. Cela est en parti dû au fait que, lorsque ces données sont au format numérique, il est possible d'en traiter de grandes quantités à l'aide de systèmes informatiques. Ces traitements peuvent être réalisés de manière automatisée et, ainsi, permettre des gains en productivité. Il est par exemple possible de retrouver des mots clés de manière rapide.

Un nombre important de documents sont toujours au format papier. Cela peut être lié au fait que ces documents datent d'une époque où l'informatisation était moins généralisée mais cela peut aussi correspondre à des documents récents écrits à la main, à des formulaires remplis ou à d'autres types de documents échangés entre personnes de manière physique. Les informations présentes dans ces documents ne peuvent être utilisées efficacement sans être transcrites du format analogique dans lequel elles sont écrites à un format numérique. Faire réaliser cette transcription par des opérateurs s'avère long et coûteux. Les techniques d'analyse et de reconnaissance automatique des documents ont pour but d'automatiser cette tâche et de permettre à un plus grand nombre de ces données d'être rendues utilisables.

Depuis plus de soixante ans, la recherche s'est penchée sur ces problématiques. Tout d'abord pour la reconnaissance de caractères isolés puis pour de la reconnaissance de mots ou de lignes. Afin d'améliorer les performances et de pouvoir traiter des bases plus hétérogènes incluant des polices variées ou, a fortiori, de l'écriture manuscrite, des techniques utilisant l'apprentissage automatique, et en particulier des réseaux de neurones, ont été proposées pour la reconnaissance du texte.

Ainsi, les tâches d'analyse de documents ont participé activement au développe-

ment des réseaux de neurones. En particulier, le neocognitron proposé par Fukushima [Fukushima and Miyake, 1982] et à l'origine des réseaux convolutionnels vise à classifier des images de chiffres. La popularisation de ces réseaux convolutionnels par LeCun et al. [LeCun et al., 1989] et la proposition de leur apprentissage par rétro-propagation des gradients ont été initialement développées pour la reconnaissance de chiffres manuscrits et on notera que la base MNIST [LeCun, 1998] est toujours utilisée par la communauté en apprentissage automatique.

Plus récemment, et avant le renouveau insufflé par les succès de Krizhevsky et al. [Krizhevsky et al., 2012] pour de la classification d'objets, des techniques de réseaux convolutionnels récurrents ont rencontré leurs premiers résultats probants pour une tâche de reconnaissance de lignes de texte manuscrites [Graves and Schmidhuber, 2009] [Grosicki and El Abed, 2009] et ont rapidement été utilisées par la communauté d'analyse de documents.

Si la reconnaissance de l'écriture, tant au niveau caractère qu'au niveau ligne, a fortement bénéficié de l'utilisation de l'apprentissage automatique et de l'apprentissage profond, peu de travaux ont été réalisés dans ce sens pour la détection des lignes de texte. De plus, au sein de la communauté de l'analyse de documents, une majorité des campagnes d'évaluation fournissant des jeux de données important ont été réalisées pour la reconnaissance du texte [Tong et al., 2014] [Sánchez et al., 2014] et pas pour l'analyse de la structure des documents.

Or, la détection et la localisation des lignes de texte est une étape importante en vue de la reconnaissance pleine page [Kim, 1997]. En effet, comme illustré dans la Figure 1.1 qui représente une chaîne classique de reconnaissance de texte pleine page, la localisation des lignes est extraite des images de pages entières, éventuellement après une étape de détection des paragraphes, avant de reconnaître le texte inclus dans ces lignes. Cela signifie que tout texte non correctement segmenté par l'étape de détection des lignes sera irrémédiablement perdu et ne pourra pas être reconnu.

De plus, comme illustré par les résultats de la campagne d'évaluation Maurdor [Brunessaux et al., 2014], et par des études complémentaires [Moysset et al., 2014b], la localisation du texte est responsable d'une part importante des erreurs lorsque l'on travaille avec des bases fortement hétérogènes comportant des documents variés et plusieurs types d'écriture. Par conséquent, la détection des lignes de texte reste un problème loin d'être résolu. C'est dans ce cadre que nous travaillons et utilisons les capacités de généralisation des techniques d'apprentissage pour la localisation des lignes de texte dans des documents hétérogènes afin de proposer un système robuste de reconnaissance de texte pleine page.

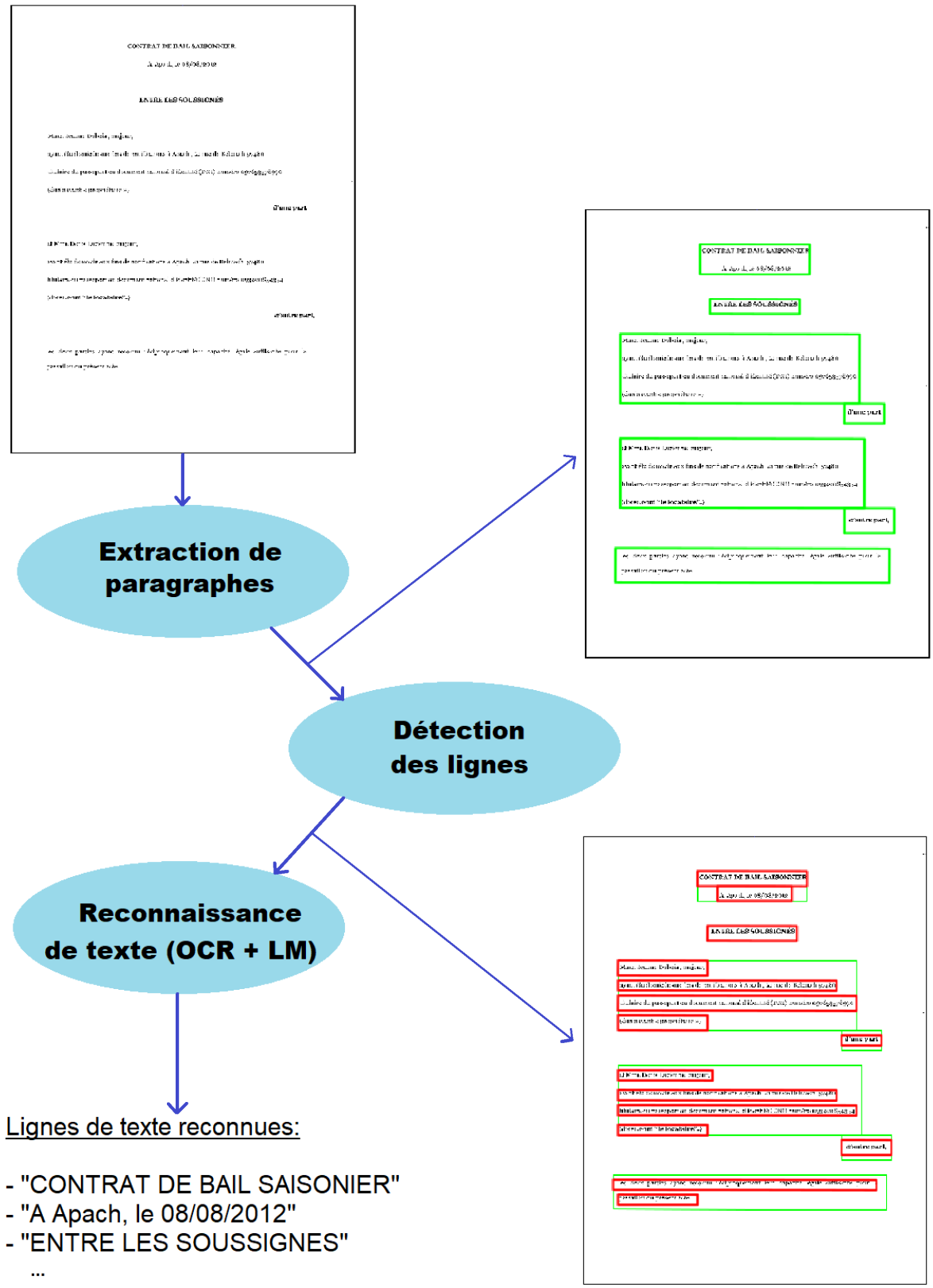


Fig. 1.1 – Une chaîne typique de reconnaissance de texte pleine page qui illustre les étapes de localisation des paragraphes, de détection des lignes de texte et de reconnaissance du contenu.

1.2 Cadre de la thèse

Cette thèse a été effectuée avec le Laboratoire d'InfoRmatique en Image et Systèmes d'information (LIRIS) de Lyon.

Elle a été réalisée au sein de la société A2iA [1], entreprise fondée en 1991 et spécialisée dans le traitement de documents pour des applications comme le traitement automatique de chèques bancaires, la reconnaissance et la classification de formulaires, de courriers ou la lecture d'archives historiques.

Les activités de recherche de A2iA [2] sont illustrées par la participation à des programmes de recherche français ou internationaux comme Pacte, Maurdor ou Himanis et par la participation à diverses campagnes d'évaluation des performances des systèmes comme, par exemple, les campagnes de reconnaissance de textes manuscrits Rimes [Grosicki and El-Abed, 2011], OpenHart [Tong et al., 2014] ou HTRtS [Sánchez et al., 2014].

Les produits de l'entreprise étant appliqués à des problèmes réels, la position des lettres ou des lignes de texte dans les documents est inconnue et proposer un système de reconnaissance pleine page est nécessaire. De plus, comme ces produits doivent traiter des documents venant de sources très hétérogènes et pas forcément connues, un algorithme robuste et performant est souhaitable et souhaité pour réaliser la localisation des lignes de texte afin de réaliser une reconnaissance pleine page du texte.

1.3 Objectifs et contributions

Le cadre dans lequel se déroule ce travail, et les spécificités de la tâche de localisation des lignes de texte nous amènent à définir un certain nombre de problématiques auxquelles nous chercherons à répondre :

— *Comment détecter efficacement les lignes de texte dans des documents hétérogènes provenant de sources différentes ?*
— *Comment apprendre à détecter un nombre variable d'objets ?*
— *Comment généraliser efficacement à partir d'un nombre réduit d'exemples ?*
— *Comment permettre de détecter un nombre important de petits objets ?*
— *Comment être capable de prendre en compte le contexte pour bénéficier de celui-ci dans*

1. www.a2ia.com
2. www.a2ialab.com

des documents fortement structurés ?
— *Comment prédire les objets détectés de manière suffisamment précise pour reconnaître correctement le texte contenu ?*
— *Comment visualiser et interpréter ce que le réseau a appris ?*

Pour répondre à ces différentes problématiques, plusieurs contributions ont été apportées lors de cette thèse. Elles nous permettent de proposer un système robuste de reconnaissance pleine page de texte :

— L'utilisation de réseaux de neurones pour prédire directement les coordonnées des boîtes englobant les lignes de texte et l'utilisation d'un terme de confiance dans le fait que ces boîtes hypothèses existent permettent de détecter un nombre d'objets variable. Cette utilisation d'une approche par apprentissage pour la localisation des lignes permet d'apprendre à détecter des lignes dans des documents hétérogènes. Prendre comme caractéristiques d'entrée les pixels de l'image, permet d'éviter de faire des suppositions sur l'apparence des caractères.
— Nous proposons d'utiliser des réseaux plus petits et locaux afin de partager les paramètres entre les localisations et, ainsi, d'être capable de prédire un nombre plus important d'objets tout en facilitant l'apprentissage avec un nombre limité de données.
— L'insertion de couches récurrentes multi-dimensionnelles nous permet de transmettre les informations contextuelles entre les différentes localisations.
— Différentes stratégies de reconnaissance pleine page sont proposées, basées sur l'observation de la précision des prédictions en fonction des coordonnées prédites. En particulier, nous proposons un système où le détecteur des lignes prédit la position des côtés gauches des objets et où le reconnaisseur de texte est chargé d'apprendre où se trouve la droite de la ligne.
— Nous visualisons les images qui activent le plus les neurones de notre réseau et nous analysons la prise en compte du contexte par les couches récurrentes à travers l'observation de la rétro-propagation des gradients jusque dans l'image d'entrée.

1.4 Organisation de la thèse

Cette thèse est composée, outre cette introduction, de huit chapitres.

- Tout d'abord, le chapitre 2 présente un aperçu de l'état de l'art. Une revue des différents procédés de détection de lignes développés durant les dernières décennies, et majoritairement basées sur des méthodes de traitements d'images, est effectuée. Sont ensuite décrites les différentes

manières de détecter des objets dans des scènes naturelles. Ces techniques, bien que appliquées à des tâches différentes de celle qui nous intéresse, partagent certains de nos objectifs tels que le fait de devoir détecter plusieurs objets d'apparence variable. De plus, d'avantage de systèmes utilisant des réseaux de neurones, et donc apparentées aux techniques que nous décrivons dans cette thèse, ont été proposés ces dernières années dans ce domaine. Enfin, nous y décrivons les techniques permettant de prendre en compte les spécificités de la tâche de détection de lignes de texte que sont la nécessité de prendre en compte le contexte et l'utilisation de bases de données de tailles réduites.

- Le chapitre 3 décrit des travaux préliminaires effectués afin de segmenter les paragraphes en lignes de texte. La segmentation est effectuée de manière mono-dimensionnelle selon l'axe vertical en utilisant un réseau de neurones convolutionnel et récurrent, utilisant des 2D-LSTMs. L'entraînement est effectué grâce à la technique de la CTC qui permet d'éviter le besoin de connaître la position explicite de chaque ligne ; seul le nombre de lignes présentes dans le paragraphe est nécessaire. Cette technique partage les technologies utilisées pour la reconnaissance de l'écriture, tant au niveau du réseau de neurones qu'au niveau de la technique d'entraînement et d'alignement. Ce chapitre permet donc également d'introduire ce réseau de reconnaissance de texte.

- Ensuite, le chapitre 4 décrit en détails les deux techniques de détection d'objets dans des scènes naturelles que sont YOLO et MultiBox et qui s'apparentent le plus à la méthode que nous proposons dans cette thèse. Ces techniques, ou des adaptations de ces techniques, ont des résultats très compétitifs sur les tâches pour lesquelles elles ont été développées. Nous soulignons les similarités de ces deux techniques au niveau de la conception des réseaux de neurones utilisés pour directement prédire les coordonnées des objets et la confiance dans le fait que les objets hypothèses existent ; mais également au niveau des fonctions de coût utilisées pour leur entraînement. Nous analysons aussi leur différence majeure qu'est l'association des boîtes références aux boîtes hypothèses et expliquons pourquoi l'appariement utilisé par YOLO n'est pas utilisable pour une tâche comme la détection de lignes de texte où de nombreux petits objets peuvent être présents. Ce chapitre permet d'introduire les notations que nous utiliserons dans ce travail.

- Nous proposons dans le chapitre 5 un modèle de détection d'objets textuels qui tient compte des spécificités de notre tâche que sont un ensemble d'entraînement réduit et un nombre important d'objets à détecter. Dans ce chapitre, sont décrites les contributions que nous avons apportées dans la construction du réseau de neurones. Ces contributions ont été apportées dans le but de traiter efficacement les problèmes associés à la tâche de détection de lignes dans des documents. En particulier, la taille des filtres convolutionnels est choisie pour être adaptée à des objets de formes majoritairement horizontales et le nombre de ces filtres est réduit afin de réduire le nombre de paramètres et de faciliter l'entraînement avec un nombre réduit d'exemples. Dans ce but de faciliter l'entraînement, une prédiction locale est introduite à travers une couche convolutionnelle finale de taille 1×1, les différentes localisations de l'image partagent leurs paramètres et voient

d'avantage d'exemples de lignes lors de l'entraînement. Cette approche locale entraîne une perte d'information de contexte puisque les sorties ont des champs réceptifs plus petits que la page. Nous récupérons cette capacité à prendre en compte le contexte en intercalant des couches de récurrence LSTM multi-dimensionnelles entre nos couches convolutionnelles.

- Le chapitre 6 décrit trois stratégies de reconnaissance de texte dans des pages entières. Ces trois stratégies se servent des méta-architectures décrites dans le chapitre 5. En revanche, elles se distinguent par le nombre de coordonnés prédites. La technique de détection d'objets par régression des coordonnées des boîtes englobant les lignes de texte introduite dans le chapitre précédent est utilisée pour détecter, tour à tour, les coins, les côtés ou la totalité des boîtes englobant les lignes de texte. Ces trois propositions différentes sont justifiées par l'impact du nombre de coordonnées prédites sur la précision des prédictions et par l'impact négatif de prédictions approximatives sur la reconnaissance du texte. Ainsi, outre la prédiction directe des boîtes complètes, nous proposons de détecter séparément les coins inférieurs gauches et supérieurs droits de ces boîtes et de les apparier. Nous proposons également une stratégie visant à ne détecter que les côtés gauches de ces boîtes et à charger le reconnaisseur de texte, réentraîné, d'apprendre à ignorer les éventuels objets présents à droite des lignes.

- Les résultats expérimentaux obtenus par les systèmes proposés sont donnés dans le chapitre 7. Après avoir décrit les jeux de données et les métriques utilisés, nous analysons les résultats de nos différentes approches à la fois géométriquement en évaluant les positions des boîtes obtenues, en mesurant si la reconnaissance du texte présent dans ces boîtes est correcte et visuellement. Une étude ablative est effectuée afin de vérifier l'impact de nos contributions principales que sont l'ajout des LSTMs et l'utilisation d'une couche locale mais aussi afin de justifier certains des choix d'implémentation les plus critiques.

- Si les résultats expérimentaux permettent de justifier la pertinence de notre approche, il nous a paru intéressant de visualiser ce qui avait été appris par notre réseau de neurones. Pour cela, nous visualisons dans le chapitre 8 ce qu'apprennent les différents neurones. Cette visualisation est effectuée de deux manières complémentaires. D'une part, nous visualisons les morceaux d'images d'entrée, parties des images complètes, qui maximisent l'activation ou la désactivation des neurones de nos différentes couches convolutionnelles et visualisons ainsi les discriminations effectuées par le réseau en fonction de la profondeur du neurone dans le réseau. D'autre part, nous visualisons la transmission des informations contextuelles grâce aux couches récurrentes en observant la rétro-propagation de gradients correspondant à une sortie donnée jusque dans l'image d'entrée.

- Finalement, le chapitre 9 permet de conclure et de proposer des pistes de recherche complémentaires.

Si ce document peut être lu dans l'ordre d'écriture qui s'efforce de décrire les méthodes utilisées

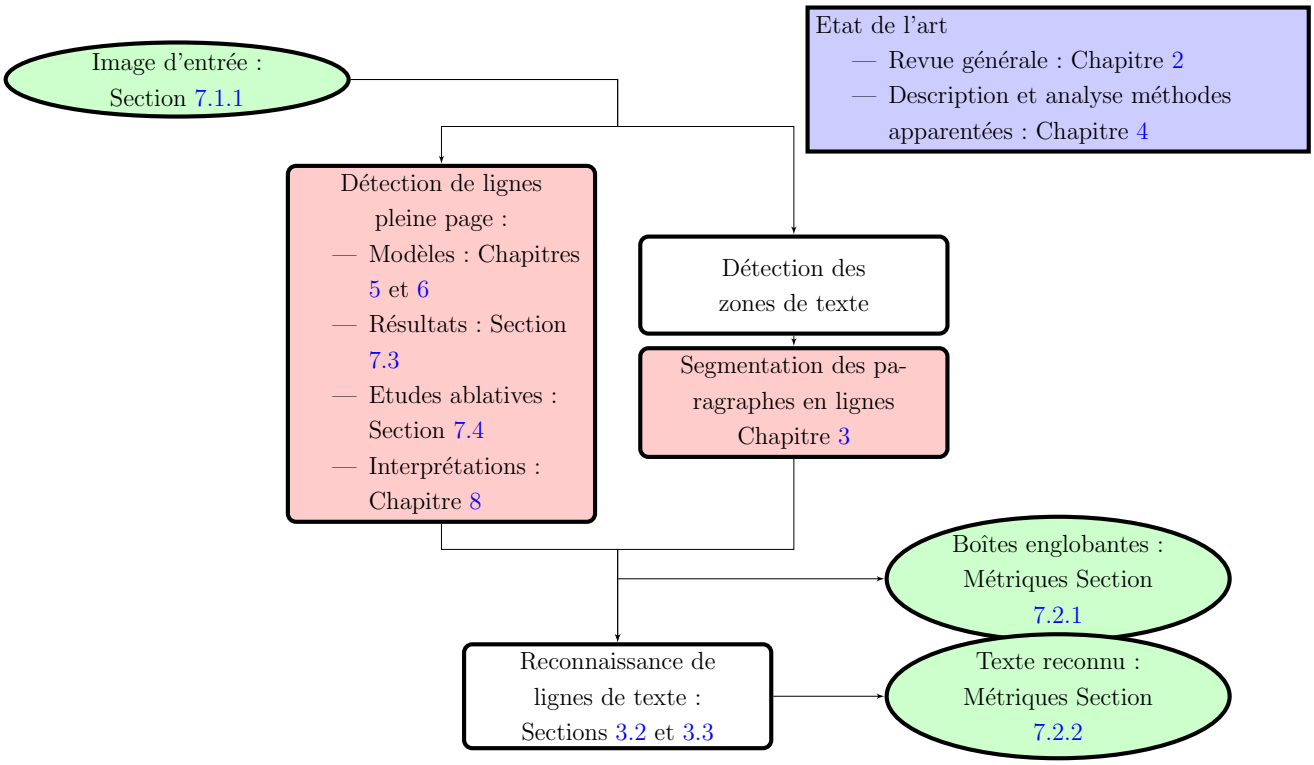


Fig. 1.2 – Chaîne de reconnaissance de l'écriture pleine page indiquant les sections correspondantes. En rouge sont indiquées les sections avec nos principales contributions, en bleu celles décrivant l'état de l'art et en vert le cadre expérimental.

les unes après les autres, un lecteur intéressé par certaines contributions pourra se référer à la Figure 1.2 qui indique les sections de ce document dans lesquelles sont décrites chaque étape de notre reconnaissance pleine page. En particulier, les chapitres 2 et 4 présentent un état de l'art. Nos contributions principales sont décrites dans les chapitres 5 et 6 alors que les résultats sont analysés dans les Chapitres 7 et 8.

Chapitre 2

État de l'art

La détection d'objets dans les images, dans un sens très large, a été traitée par des sous-communautés différentes de la vision par ordinateur. Les approches proposées en analyse de documents ont été souvent différentes des approches utilisées pour la détection d'objets dans des scènes naturelles. L'apparition des techniques à base de réseaux de neurones et d'apprentissage profond suscitent un mouvement de convergence vers la cohérence des communautés.

Nous décrirons en Section 2.1 les méthodes traditionnelles utilisées pour localiser les lignes de texte dans les documents. Puis, nous étudierons en Section 2.2, les approches utilisées pour la détection d'objets dans les scènes naturelles et en particulier, dans la Section 2.2.4, les approches à base de réseaux de neurones auxquelles s'apparente la méthode que nous proposerons dans cette thèse. Nous discuterons, en Section 2.3 des techniques de réseaux de neurones relatives à nos contributions. En particulier, nous discuterons de la prise en compte du contexte en Section 2.3.1 et de l'utilisation de petits ensembles d'entraînements en Section 2.3.2.

2.1 Détection de lignes de texte dans des images de documents

De nombreuses méthodes ont été, depuis plusieurs décennies, proposées pour analyser les documents et extraire les objets relatifs à la composition des pages [Likforman-Sulem et al., 2007] [Eskenazi et al., 2017]. Les styles de documents étudiés ont été très variés, des documents imprimés aux lettres manuscrites en passant par les documents historiques et les formulaires. Les objectifs de cette analyse ont également été différents et les objets extraits en conséquence. Blocs de texte, lignes de texte, mots ou caractères, analyse des objets entourant le texte tels que logos, tableaux ou figures.

Dans ce cadre, de nombreuses méthodes ont été développées et cette partie en décrira les grandes lignes. La diversité des méthodes reflète la diversité des problèmes tels que traiter des documents imprimés ou manuscrits, des formulaires ou bien des lettres, ou des documents historiques mais aussi la diversité des niveaux d'analyse, détection de blocs de texte, de lignes de texte ou bien de mots. S'ils partagent un objectif commun, notons que notre méthode ne s'apparente donc pas technologiquement à la majorité de ces approches qui sont basées sur des techniques de traitements d'image.

Historiquement, les approches utilisées pour détecter les lignes de texte ont pu être divisées en deux classes. Les approches ascendantes, décrites en Section 2.1.1, agrègent progressivement de petites composantes pour arriver aux objets et les approches descendantes, décrites en Section 2.1.2 qui divisent progressivement l'image complète pour arriver aux objets. Enfin, nous décrirons en Section 2.1.3 les techniques de détection de texte à base d'apprentissage automatique, en général plus récentes. C'est dans ce cadre que notre travail prend place, ces techniques ont principalement été développées pour la détection de mots dans des scènes naturelles.

2.1.1 Approches ascendantes

Les techniques ascendantes de détection de lignes de texte groupent de petits éléments pour former les lignes de texte. Elles se décomposent en deux familles. La première, se présente sous forme de filtrages et vise à grouper les pixels de l'image alors que la seconde part d'une décomposition en composantes connexes et groupe celles-ci.

Première famille d'approches ascendantes : filtrages

Les approches ascendantes appliquées à la segmentation des lignes de texte dans des documents peuvent commencer avec les techniques de morphologie mathématique, ces techniques sont également appelées *smearing*. L'algorithme RLSA (*Run-Length Smoothing Algorithm*) [Wong et al., 1982] en est un bon exemple puisqu'il remplit horizontalement les espaces entre pixels noirs proches. Les objets détectés sont relatifs aux composantes connexes obtenues. Ces objets peuvent être des mots, des lignes ou des paragraphes de texte en fonction du filtre choisi pour la dilatation. Cela revient à dire que l'on groupe les pixels noirs suffisamment proches les uns des autres. D'autres filtres que le filtre horizontal ont été utilisés [Khayyat et al., 2012] pour des taches différentes. Des techniques similaires ont utilisé des successions d'opérations morphologiques de type ouverture et fermeture [Papavassiliou et al., 2010].

Cette approche a été élargie aux images en niveaux de gris [Shi and Govindaraju, 2004]. Et plusieurs techniques ont utilisé des filtres de natures différentes pour fusionner les pixels, lettres

ou mots appartenant à la même ligne. C'est le cas de Shi et al. [Shi et al., 2009] qui proposent une méthode adaptée à l'orientation variable des lignes de texte, en particulier manuscrites, en utilisant une série de filtres en forme d'ellipses ayant des orientations différentes. Dans cette technique, l'image floutée est seuillée et les composantes extraites sont les lignes de texte.

Li et al. [Li et al., 2008] utilisent également une image floutée, à l'aide de *density probability function*, mais utilise une technique de *levelset* pour segmenter les différents objets. Cette technique a été adaptée en utilisant la fonction d'énergie de Mumford-Shah dans le *levelset* [Du et al., 2009]. Lemaitre et al. [Lemaitre and Camillerapp, 2006] appliquent un filtre de Kalman à une image en basse résolution, floutée, pour trouver les lignes.

Deuxième famille d'approches ascendantes : Composantes connexes

D'autres approches ascendantes se basent sur les composantes connexes comme élément de base et groupent ces composantes connexes pour former les lignes de texte.

Cela peut être fait à l'aide d'heuristiques basées sur la position relative des composantes connexes, leurs tailles et surfaces respectives [Boulid et al., 2016] ou sur les directions données par leur lignes de base [Feldbach and Tonnies, 2001]. Rabaev et al. [Rabaev et al., 2013] permettent de travailler avec des images historiques fortement endommagées en utilisant directement des images en niveaux de gris. Les composantes connexes sont obtenues à différents seuils et sont conservées si elles ont une forme correcte avant d'être groupées.

Cela peut aussi être fait en créant un graphe entre les composantes connexes proches les unes des autres ; cette proximité étant définie en utilisant les K plus proches voisins [O'Gorman, 1993], à partir d'un diagramme de Voronoi qui sépare les aires d'influence des différentes composantes connexes [Kise et al., 1998] ou à partir d'un *Minimum Spanning Tree* [Abuhaiba et al., 1995] [Yin and Liu, 2009] . Ces graphes sont ensuite coupés pour former les lignes de texte à partir de règles basées sur les angles entre eux ou sur les distances qui les séparent ou d'autres techniques de *clustering*. Des graphes similaires peuvent aussi être coupés afin de former les lignes à l'aide d'algorithmes de théorie des graphes [Davis et al., 2015] [Kumar et al., 2006].

Plusieurs méthodes se basent sur la transformée de Hough pour trouver les alignements entre les centroïdes de ces composantes connexes [Likforman-Sulem et al., 1995] [Louloudis et al., 2006] [Louloudis et al., 2009].

Lim et al. [Lim et al., 2007] et Nguyen et al. [Nguyen et al., 2010] utilisent une technique non supervisée de *tensor voting* pour extraire des caractéristiques, qui après sélections selon des critères choisis permettent d'extraire les composantes connexes de texte dans des scènes naturelles.

Enfin, Ryu et al. [Ryu et al., 2014] assemblent des parties de composantes connexes pour former les lignes en minimisant une fonction de coût de manière itérative, en fusionnant ou séparant successivement les ensembles de parties de composantes connexes.

2.1.2 Approches descendantes

Les approches descendantes, quant à elles, prennent comme élément de base la page entière et divisent celle-ci progressivement. Elles voient leur apparition avec des méthodes de type X-Y cut [Nagy et al., 1992] pour lesquelles on cherche des vallées sans écritures, alternativement de manière horizontale et verticale, pour séparer les objets.

Avec une approche analogue, [Baird, 1994] cherche dans l'image les rectangles blancs de taille maximale. Ces blocs, en fonction de leurs formes et de leurs tailles, définissent des inter-paragraphes, des inter-lignes, des inter-mots ou des inter-caractères.

Les méthodes à base de projections de profils [Messelodi and Modena, 1999], pour lesquelles un histogramme de la présence des pixels est construit horizontalement permettent de s'adapter à d'éventuels bruits de binarisation ou à des caractères se touchant. Cet histogramme est ensuite seuillé pour séparer les lignes des interlignes. Cette technique a été adaptée à la segmentation de lignes manuscrites [Zahour et al., 2004] [Zahour et al., 2007] en effectuant cette projection sur des colonnes du texte et pas sur toute la largeur de la page afin de permettre de prendre en charge l'inclinaison des lignes. Weliwitage et al. [Weliwitage et al., 2005] segmentent les lignes manuscrites en raffinant les résultats de la projection de profils en évitant de traverser les ascendants et descendants alors que Bulacu et al. [Bulacu et al., 2007] proposent de contourner ces ascendants et descendants.

De nombreuses techniques tendent à améliorer cette idée en essayant de trouver un passage avec le moins de pixels noirs entre les lignes. C'est le cas pour Saabni et al. [Saabni and El-Sana, 2011] qui utilisent une technique à base de *seam carving* où une *Signed Distance Transform* est appliquée à l'image et utilisée comme carte d'énergie dans laquelle chaque pixel a pour valeur sa distance minimale à un pixel de fond. Le *seam carving* est appliqué sur cette carte d'énergie pour trouver les chemins passant par le centre des lignes. Arvanitopoulos et al. [Arvanitopoulos and Süsstrunk, 2014] adaptent cette technique aux images de documents historiques en couleurs en utilisant le *seam carving* pour tracer un sillon entre les lignes consécutives. Surinta et al. [Surinta et al., 2014] utilisent des techniques de *path planning* pour lier des points à gauches et à droites de l'image en limitant le nombre de pixels noirs traversés. Similairement, Stafylakis et al. utilisent un algorithme de Viterbi [Stafylakis et al., 2008] pour lier les bords de l'image avec des états de transition pénalisant la traversée de pixels noirs et les déplacements verticaux. Nicolaou et al. [Nicolaou and Gatos, 2009] tracent progressivement un sillon entre les

lignes d'un côté de l'image vers l'autre en suivant les gradients indiqués par une image floutée avec un filtre horizontal. Basu et al. [Basu et al., 2007] utilisent une technique à base de *waterflow* où les pixels noirs de l'image binarisée correspondent aux obstacles à la propagation du flux. Enfin, Brodic et al. [Brodić and Milivojević, 2016] étendent cette technique en complexifiant la fonction de propagation du flux. Les zones non touchées par le flux sont les lignes de texte.

Ces techniques ont en commun de chercher un chemin entre les lignes et sont donc adaptées à de la segmentation de paragraphes ou de pages avec une seule colonne. Cependant, elles rencontrent des problèmes pour la segmentation pleine page de documents avec des mises en page plus complexes.

2.1.3 Approches par apprentissage automatique

Les techniques présentées dans la section précédente, aussi bien ascendantes que descendantes, ont pour limite la variabilité des documents réels et nécessitent souvent d'importants efforts d'adaptation à une nouvelle base, que ce soit pour obtenir une binarisation (et donc une décomposition en composantes connexes) satisfaisante ou pour trouver les hyper-paramètres permettant de bien séparer les lignes entre elles. Pour pallier ces difficultés, des techniques à base d'apprentissage automatique ont récemment été utilisées avec pour but d'apprendre la variabilité des situations. Un modèle est appris en fonction des données. C'est dans ce cadre que nos travaux se situent.

Avant l'utilisation de méthodes de *deep learning*, et de manière similaire à ce qui a été proposé pour la détection d'objets naturels, les modèles graphiques ont été utilisés pour la détection d'objets textuels, en particulier à partir de *Conditional Random Fields* (CRF [Lafferty et al., 2001]). Ainsi, Pan et al. [Pan et al., 2011] apprennent un CRF pour identifier les composantes connexes de texte alors que Hebert et al. [Hebert et al., 2011] utilisent un CRF pour identifier les pixels de l'image d'entrée qui correspondent à une zone de ligne.

On notera cependant que l'utilisation de réseaux de neurones apparaît, dès 2001, proposé par Jung [Jung, 2001] qui utilise un *Multi-Layer Perceptron* pour détecter du texte avec des tailles, styles et couleurs différents. Les réseaux ont pour entrée des caractéristiques de texture et pour sortie la présence ou l'absence de texte.

De même, Delakis et al. [Delakis and Garcia, 2008], appliquent un réseau de neurones convolutionnel à une image de scène naturelle avec une fenêtre glissante et classifient chaque localisation comme appartenant à une ligne, à un interligne ou au fond.

Plus récemment, certaines techniques utilisent des réseaux de neurones pour identifier le texte au niveau pixelique. Baechler et al. [Baechler et al., 2013] et Fischer et al. [Fischer et al., 2014]

utilisent un perceptron multi-couches pour discriminer, à partir de *features* liées au voisinage proche de chaque pixel, entre les pixels de texte et les pixels appartenant au fond ou à des interlignes dans des documents historiques. Ces techniques sont étendues avec des techniques d'apprentissage profond par Chen et al. [Chen and Seuret, 2017] qui utilisent un réseau de neurones convolutionnel pour classer les pixels suivant s'ils sont du texte ou d'autres objets. Vo et al. [Vo and Lee, 2016] utilisent un réseau complètement convolutionnel (*Fully Convolutional network*, FCN [Long et al., 2015]) dans un but similaire. Zhang et al. [Zhang et al., 2016] utilisent aussi un FCN pour détecter les pixels de texte, dans des images de scènes naturelles, la particularité étant que le FCN est multi-échelles. He et al. [He et al., 2016], toujours pour de la détection de texte dans des scènes naturelles, classifient aussi les pixels avec un réseau convolutionnel. Cette classification est améliorée par l'utilisation d'un apprentissage multi-tâche où la classification texte/non-texte est effectuée parallèlement à la classification du caractère présent à cet endroit et par l'utilisation d'une étape de post-traitement à base d'augmentation de contraste.

Plus récemment, et concurremment de notre travail, des techniques à base d'apprentissage profond ont aussi été utilisées pour directement détecter les positions des lignes de texte, essentiellement dans des images de scènes naturelles. Dans ces approches, comme dans nos travaux, la position des objets textuels est prédite par le réseau à l'aide d'une régression sur cette position.

Gupta et al. [Gupta et al., 2016] utilisent une adaptation de la méthode YOLO, qui sera décrite en détails dans la Section 4, pour détecter les lignes de texte et apportent une base synthétique pour l'entraînement alors que Jaderberg et al. [Jaderberg et al., 2016] utilisent EdgeBoxes [Zitnick and Dollár, 2014] pour détecter les boites englobant les mots et raffinent les prédictions avec une régression locale dans un deuxième temps.

Plusieurs techniques se sont focalisées sur la possibilité de pouvoir prédire autre chose que des boîtes horizontales, et ainsi de pouvoir s'adapter à l'inclinaison des textes, souvent présente dans les images de scènes naturelles dans lesquelles les textes sont inclinés à cause de la perspective. C'est notamment le cas de Ma et al. [Ma et al., 2017] qui adaptent Faster RCNN [Ren et al., 2015] en ajoutant des ancres inclinées. C'est aussi le cas de Liu et al. [Liu and Jin, 2017] qui étendent SSD [Liu et al., 2016] à la détection de lignes de mots dans des scènes naturelles en prédisant les 8 coordonnées de quadrilatères au lieu des 4 coordonnées nécessaires à la position de rectangles horizontaux.

He et al. [He et al., 2017] prédisent la position du centre des objets et la position relative des 4 sommets du quadrilatères englobant le mot. Enfin, Shi et al. [Shi et al., 2017] utilisent SSD, donc une régression multi-échelles pour détecter la position des lettres. Des mots inclinés peuvent ensuite être créés en groupant les lettres. Tian et al. [Tian et al., 2016] poussent cette décomposition plus loin en ne prédisant que le haut et le bas éventuel des caractères pour chaque position dans la dernière carte de caractéristiques de la partie convolutionnelle du réseau. On y

notera la présence de réseaux récurrents de type LSTMs sur la dimension horizontale pour prendre en compte le contexte.

Ces techniques sont directement inspirées de techniques de détection d'objets dans des scènes naturelles, celles-ci seront discutées dans la Section 2.2 suivante.

Enfin, certaines approches tendent à fusionner la tâche de détection du texte avec la tâche de reconnaissance du texte. C'est notamment le cas des techniques qui utilisent des modèles d'attention pour prédire la position des objets textuels. Bluche et al. [Bluche et al., 2016b] utilisent un réseau convolutionnel avec des MD-LSTMs pour déterminer quelle partie de l'image est utile pour reconnaître une lettre. Les différentes lettres du paragraphe sont lues les unes après les autres grâce à la mémoire transmise par une récurrence. Dans [Bluche, 2016], ce même modèle d'attention est utilisé pour donner la position des lignes du paragraphe et les reconnaître, permettant ainsi une segmentation et une reconnaissance jointe.

Enfin, Bartz et al. [Bartz et al., 2017] effectuent une reconnaissance pleine page en utilisant un *spatial transformer network* (STN [Jaderberg et al., 2015]) pour localiser les mots dans des scènes naturelles. Ce STN est appris de manière indirectement supervisée à travers la propagation de gradients venant de la tâche de reconnaissance du texte présent dans les boîtes.

2.2 Détection d'objets dans des images naturelles

Si beaucoup de méthodes très variées ont été proposées pour la détection de lignes de texte, les méthodes les plus récentes comme la notre ou celles décrites en Section 2.1.3 s'inspirent directement de méthodes pensées pour détecter d'autres types d'objets et en particulier les objets présents dans des scènes naturelles. Pour cela, nous décrirons dans cette section la variété de méthodes proposées et mettront l'emphase sur les techniques utilisant des réseaux de neurones, et en particulier en Section 2.2.4 sur les techniques utilisant une régression directe des coordonnées des boîtes englobant les objets.

2.2.1 Techniques traditionnelles à base de traitements d'image

Similairement à ce que nous avons détaillé pour les tâches d'analyse de documents, la détection d'objets dans des images naturelles a initialement utilisé des approches basées sur le traitement d'images. Cela a été le cas, par exemple, à travers des appariement de caractéristiques locales comme SIFT [Lowe, 2004] qui détecte des points d'intérêt, ou en utilisant des ondelettes [Papageorgiou et al., 1998], en utilisant des filtres orientés [Jones and Viola, 2001]

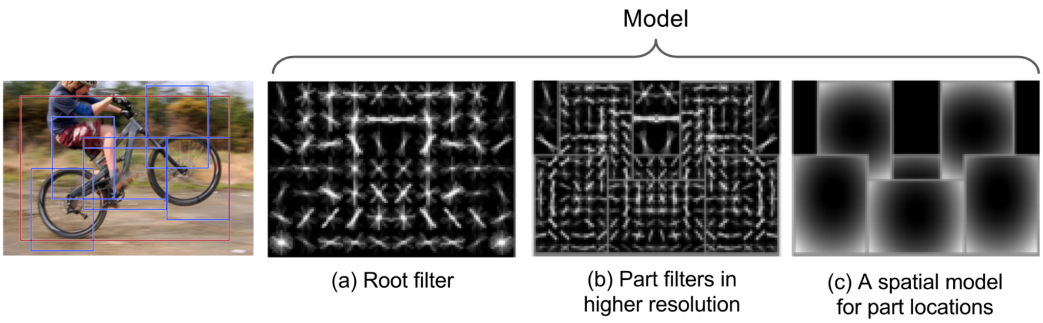


Fig. 2.1 – Illustration du fonctionnement des modèles déformables. Image issue de [Felzenszwalb et al., 2010].

[Viola and Jones, 2004] pour la détection de visages ou pour détecter des piétons à l'aide d'histogrammes d'orientation des gradients (HOG)[Dalal and Triggs, 2005].

2.2.2 Utilisation de l'apprentissage automatique pré-*Deep Learning*

Les méthodes à base d'apprentissage automatique ont également été utilisées. Cela peut remonter à l'utilisation de modèles analogues à des régresions linéaires par Fischler et al. [Fischler and Elschlager, 1973]. Plus récemment, c'est le cas de Feltzenszwalb et al. [Felzenszwalb et al., 2010] qui introduisent des modèles déformables qui reconnaissent des objets à partir de la combinaison de caractéristiques HOG correspondant aux différentes parties des objets et résolvent un problème combinatoire pour associer ces parties. Ces modèles déformables sont illustrés en Figure 2.1.

Carreira et al. [Carreira and Sminchisescu, 2010] et Endres et al. [Endres and Hoiem, 2010] utilisent des algorithmes de segmentation de graphes avec différentes initialisations pour créer des objets candidats ; le graphe correspondant aux liens entre pixels voisins appartenant à une image. Ces objets candidats sont ensuite triés à partir de leurs caractéristiques selon leur probabilité de correspondre à un objet.

Ce concept de calculer la probabilité qu'un objet hypothèse soit un objet réel, sans savoir la classe de cet objet, concept aussi appelé *objectness*, est aussi utilisé par Alexe et al. [Alexe et al., 2010] [Alexe et al., 2012] en utilisant une combinaison de caractéristiques et un classifieur bayésien.

2.2.3 Réseaux de neurones pour la détection d'objets

Comme pour de nombreuses tâches d'apprentissage automatique, et en particulier de vision par ordinateur, la détection d'objets dans des scènes naturelles a été particulièrement transformé par la résurgence de l'utilisation des réseaux de neurones ayant suivi la compétition ILSVRC 2012 et les travaux de Krizevski et al. [Krizhevsky et al., 2012].

Nous noterons néanmoins que des réseaux de neurones récurrents avaient été précédement utilisés pour détecter les points caractéristiques des visages et les suivre par Behnke [Behnke, 2005].

L'utilisation de réseaux de neurones pour détecter les objets peut être faite en utilisant une fenêtre glissante sur l'image et en classifiant si l'une des classes d'intérêt est présente à la positon donnée, à différente échelles [Sermanet et al., 2013]. Cette approche a pour désavantage d'être particulièrement gourmande en temps de calcul puisque la classification doit être faite à chaque position, pour chaque échelle et chaque forme d'objet différente.

Pour cette raison, Uijlings et al. [Uijlings et al., 2013] proposent de classifier des superpixels ou des groupements de superpixels issus d'une segmentation de graphe basée sur les textures de l'image [Felzenszwalb and Huttenlocher, 2004]. La classification est effectuée à l'aide d'une machine à vecteurs de support (SVM) que Farabet et al. [Farabet et al., 2013] [Farabet, 2013] remplacent par un réseau convolutionnel pour effectuer cette classification.

La technique EdgeBoxes [Zitnick and Dollár, 2014] propose des objets candidats à partir des contours des objets et considère que plus le nombre de contours inclus est grand, plus il est probable que cela contienne un objet.

Enfin, Pinheiro et al. [Pinheiro et al., 2015, Pinheiro et al., 2016] utilisent une déconvolution pour prédire la position des objets, directement au niveau pixel.

2.2.4 Régression directe des coordonnées des objets à l'aide de réseaux de neurones

La détection des boîtes a également été faite en utilisant le modèle convolutionnel pour prédire directement les coordonnées des boîtes englobant les objets. Cela a été proposé initialement par Szegedy et al. [Szegedy et al., 2013] pour la localisation d'un unique objet dans l'image, suivi par une proposition de Erhan et al. [Erhan et al., 2014] pour une généralisation à un nombre variable d'objets. C'est cette technique, nommée MultiBox, qui se rapproche le plus de notre méthode et elle sera détaillée plus en détails et analysée en Section 4. Elle introduit une prédiction de la confiance concernant l'existence de l'objet, confiance prédite parallèlement aux coordonnées des

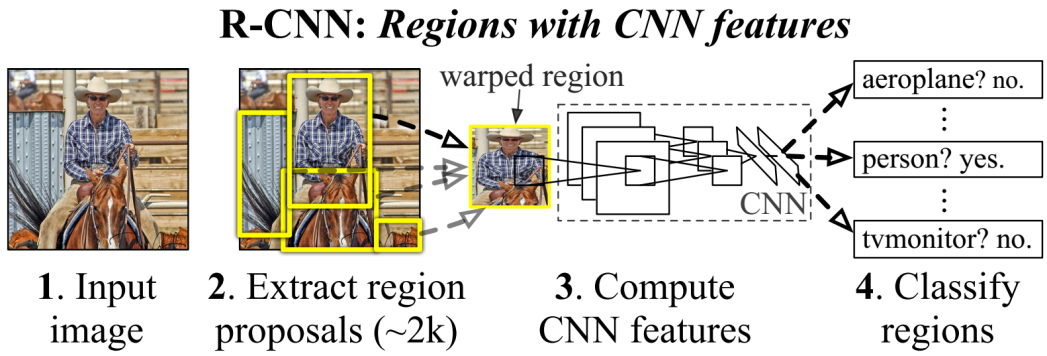


Fig. 2.2 – Illustration du fonctionnement du modèle R-CNN. Image issue de [Girshick et al., 2014].

boîtes. Szegedy et al. [Szegedy et al., 2015b] et SSD [Liu et al., 2016] étendent cette technique en utilisant de plus gros réseaux de neurones et une approche multi-échelles.

YOLO [Redmon et al., 2016] prédit également les coordonnées des boîtes mais prédit également, de manière simultanée, la classe de l'objet.

R-CNN [Girshick et al., 2014], illustré en Figure 2.2, utilise des objets candidats obtenus à partir de superpixels et classifie ces propositions, les triant au passage. Une régression est appliquée pour modifier la position exacte de ces objets. Cette technique a été étendue par Fast R-CNN [Girshick, 2015] qui rend la technique plus rapide en partageant l'extraction des caractéristiques bas niveau correspondant aux premières couches convolutionnelles entre les positions. Faster R-CNN [Ren et al., 2015] y associe des réseaux de neurones pour prédire les objets candidats à la manière de MultiBox. Cette approche Faster R-CNN est étendue par l'ajout de notions multi-échelles par Lin et al. [Lin et al., 2016].

R-FCN [Dai et al., 2016], illustré en Figure 2.3, étend l'approche Faster R-CNN en utilisant des réseaux complètement convolutionnels, qui permettent de traiter des images de tailles variables, afin de prédire des parties des objets. Mordan et al. [Mordan et al., 2017] permet à ces parties d'être déformables.

Similairement à notre méthode, IoN [Bell et al., 2016] utilise des récurrences spatiales pour transmettre des informations contextuelles suivant les quatre directions. Cependant, les transitions cachées de ces couches récurrentes sont fixées à l'identité alors que notre modèle utilise des couches de LSTMs multi-directionnelles complètes et entraînables.

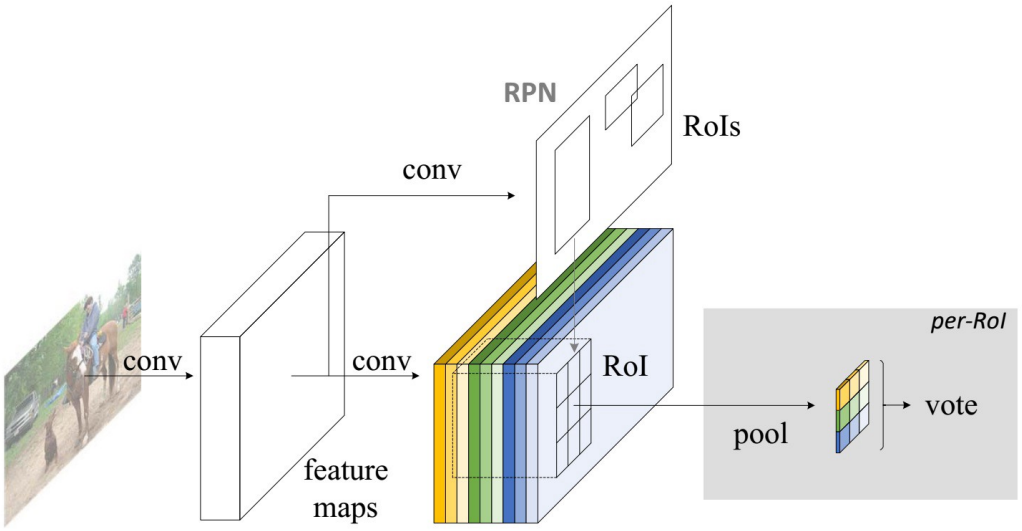


Fig. 2.3 – Illustration du fonctionnement du modèle R-FCN. Image issue de [Dai et al., 2016].

2.3 Techniques d'apprentissage profond pertinentes pour les défis liés à la reconnaissance de texte pleine page

Les techniques présentées en Section 2.2.4 partagent le fait d'utiliser des réseaux de neurones pour prédire directement les coordonnées des boîtes englobant les objets. C'est dans ce cadre que nous travaillons. Par contre, nous n'adressons pas le problème de détection d'objets dans des scènes naturelles mais celui de la détection de lignes de texte dans des documents. Ce problème est différent. En particulier, le caractère très structuré des documents incite à vouloir prendre en compte les interactions entre les objets, l'état de l'art de la prise en compte de ce contexte sera détaillé en Section 2.3.1. Une autre particularité de la tâche de détection d'objets textuels est le caractère plus réduit des bases de données utilisées. Les différentes approches utilisées pour apprendre avec de petites bases seront détaillées en Section 2.3.2.

2.3.1 Prendre en compte le contexte

Les couches convolutionnelles permettent de transmettre des informations de contexte à travers la superposition des champs réceptifs mais ce contexte reste local puisque restreint auxdits champs réceptifs.

D'autres travaux ont exploité des modèles graphiques probabilistes comme les *Conditional Random Fields* (CRF) [Krähenbühl and Koltun, 2011] pour capturer les dépendances structurelles dans les images [Zheng et al., 2015]. Néanmoins ces CRFs ont pour défaut d'avoir, en général, des inférences exactes non tractables.

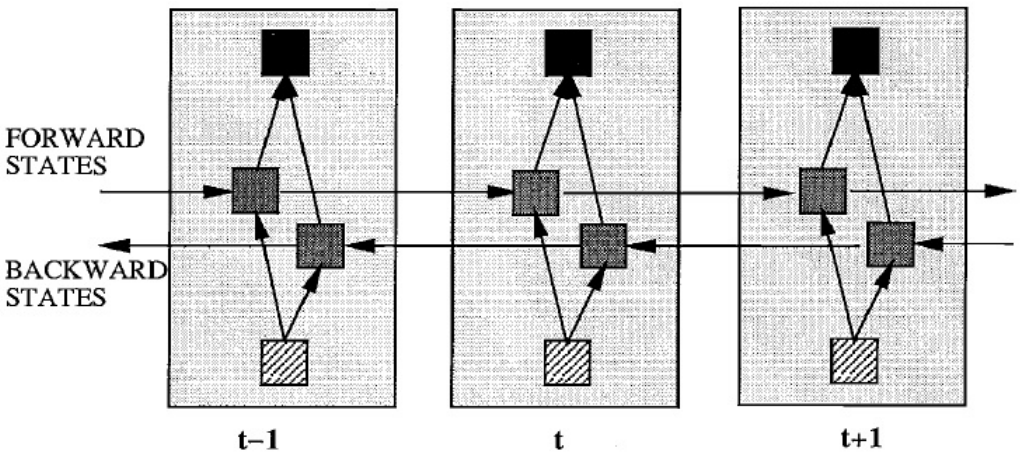


Fig. 2.4 – Illustration du fonctionnement d'une cellule récurrente bi-directionnelle. Image issue de [Schuster and Paliwal, 1997].

L'une des contributions de notre travail est l'ajout de LSTM multi-dimensionelles entre les couches convolutionnelles de notre réseau de détection des boîtes englobant les objets. Ces couches récurrentes, dont l'utilisation sera décrite en détails en Section 5.4 ont été proposées par Graves et al. [Graves and Schmidhuber, 2009]. Elles sont inspirées des LSTMs mono-dimensionelles [Hochreiter and Schmidhuber, 1997], et des récurrences bi-directionnelles [Schuster and Paliwal, 1997] illustrées en Figure 2.4, qui traitent des signaux temporels et, à l'aide de portes et d'une mémoire interne, permettent de transmettre des informations à la fois à court et à long terme en diminuant les problèmes d'évanouissement des gradients. Ces LSTMs multi-dimensionelles reçoivent des informations à la fois des zones adjacentes verticalement et horizontalement.

Les *Grid LSTMs* proposées par Kalchbrenner et al. [Kalchbrenner et al., 2015], et dont la version 3D s'applique à des signaux spatiaux, modifie cette LSTM en lui permettant de prendre en compte à la fois l'information spatiale mais aussi l'information venant des autres couches du réseau.

Les LSTMs convolutionelles [Xingjian et al., 2015] permettent de prendre un environnement convolutionnel pour calculer les activations et portes de notre LSTM alors que les *Gated Recurrent Units* (GRU) [Chung et al., 2014] simplifient cette LSTM en supprimant des portes, et en en fusionnant d'autres. Elle ont aussi été adaptées en deux dimensions [Huang et al., 2016].

Afin de paralléliser plus efficacement les calculs, Visin et al. [Visin et al., 2015] proposent de remplacer les LSTMs multi-dimensionelles par une succession de LSTMs mono-dimensionnelles verticales et horizontales. Dans une optique similaire, Stollenga et al. [Stollenga et al., 2015] proposent une gestion pyramidale des données afin de pouvoir calculer simultanément tous les éléments d'un même plan. Une variation de l'image d'entrée permet d'utiliser ces LSTMs pyramidales

pour calculer la LSTM 2D originale [Doetsch et al., 2016].

Les modèles d'attention [Cho et al., 2015] sont une autre forme de transmission de contexte en explicitant au réseau où l'information intéressante se situe.

2.3.2 Apprendre avec de petites bases

Une autre particularité de la tâche de détection de lignes de texte est le plus petit volume de données disponibles pour l'apprentissage supervisé. Plusieurs méthodes ont été proposées afin de faciliter ces entraînements. Tout d'abbord les couches convolutionnelles proposées par Fukushima et al. [Fukushima, 1979] et ensuite entraînées par rétropropagation de gradients [LeCun et al., 1989] permettent de partager les paramètres du réseau entre les différentes positions du réseau. Permettre le partage de ces paramètres permet de mutualiser et donc de faciliter leur entraînement puisque une certaine invariance spatiale existe dans les formes pouvant être présentes dans l'image. C'est en partie ce qui inspirera la modification de notre modèle pour le rendre local, ce qui sera détaillé en Section 5.3.

Une autre manière de gérer le manque de données est d'éviter le sur-apprentissage à l'aide de régularisation, c'est à dire en encourageant lors de l'entraînement le réseau à garder les valeurs des paramètres réduites. Cela peut être fait avec la technique traditionnelle de *weight decay* [Krogh and Hertz, 1992] qui va mettre une pénalité sur les paramètres proportionelle à leur norme et donc les inciter à garder des valeurs faibles. Le *Dropout* [Hinton et al., 2012] qui annule aléatoirement la valeur de certains neurones durant l'entraînement pour éviter les co-adaptations entre neurones va également avoir des effets régularisants.

Le choix de la fonction d'optimisation peut aussi permettre de faciliter l'entraînement. C'est le cas de la descente de gradient stochastique [Bottou, 1998] [Kiwiel, 2001], qui calcule les gradients et applique les mises à jours des paramètres correspondantes sur de petits sous-ensembles de l'ensemble d'entraînement. D'autres techniques, permettant d'avoir des taux d'apprentissage variables et différents sur les différents paramètres du réseau ont également été proposés afin d'accélérer les entraînements et de faciliter la convergence vers un minimum satisfaisant. C'est par exemple le cas de RmsProp [Tieleman and Hinton, 2012], d'AdaGrad [Duchi et al., 2011] ou d'Adam [Kingma and Ba, 2014].

Enfin, mentionnons les techniques d'apprentissage semi-supervisé ou non supervisé qui permettent d'entraîner les réseaux sur la tâche cible avec un nombre limité de données en profitant de données non annotées ou d'annotations correspondant à des tâches différentes.

Cela peut être fait en utilisant des techniques d'auto-encodeur [Bengio et al., 2009] où le réseau est chargé de reconstruire l'image qu'il a en entrée, le but étant de compresser, sur une couche inter-

médiaire, l'information présente dans le signal d'entrée. Cette information compressée peut ensuite être utilisée pour la tâche cible. Les auto-encodeurs variationnels [Kingma and Welling, 2013] bruitent le signal de l'auto-encodeur et vise à rendre celui-ci, et donc l'information compressée, plus robuste. D'autres approches, dites de transfert d'apprentissage [Yosinski et al., 2014], initialisent l'entraînement avec des paramètres pré-entraînés sur d'autres bases de données et éventuellement d'autres tâches. Elles s'appuient sur le fait que les caractéristiques extraites par les couches de convolutions, en particulier les premières, sont similaires pour différentes tâches de reconnaissance d'images. Il est aussi possible d'adapter l'entraînement à partir d'une tâche proche [Ganin et al., 2016] ou de joindre les apprentissages dans une approche multi-tâches [Bilen and Vedaldi, 2016] pour mutualiser les apprentissages.

Chapitre 3

Segmentation mono-dimensionnelle d'images de paragraphes en lignes de texte

Lors de travaux préliminaires [Moysset et al., 2015b], nous avons réalisé la segmentation de paragraphes de texte en lignes. Pour ce faire, nous avons utilisé un système similaire à celui utilisé pour la reconnaissance de l'écriture [Pham et al., 2014] avec une séquence de lignes de texte et d'interlignes en lieu et place des caractères utilisés pour la reconnaissance de l'écriture. Cette méthode est motivée par le fait que la position des paragraphes ainsi que le nombre de lignes dans ces paragraphes sont disponibles dans l'annotation de la base Maurdor que nous utilisons (et qui sera détaillée en section 7.1.1). Un avantage clé de cette méthode, par rapport aux autres méthodes à base d'apprentissage automatique, réside dans le fait qu'aucune annotation supplémentaire vis à vis de la position des lignes de texte n'est nécessaire.

Dans la Section 3.1, nous détaillerons le problème que nous cherchons à résoudre et motiverons le fait d'utiliser une approche similaire à ce qui est utilisé pour la reconnaissance de texte dans cette thèse. Nous décrirons ensuite ce système de reconnaissance de texte en détaillant en Section 3.2 comment un réseau de neurones est utilisé pour prédire les caractères et en décrivant dans la Section 3.3 le principe de la CTC qui nous permet d'entraîner le réseau en alignant les séquences références et hypothèses de manière implicite.

Dans la Section 3.4, nous décrirons comment est utilisé notre modèle de segmentation de paragraphes en lignes de texte et nous analyserons les résultats obtenus en section 3.5.

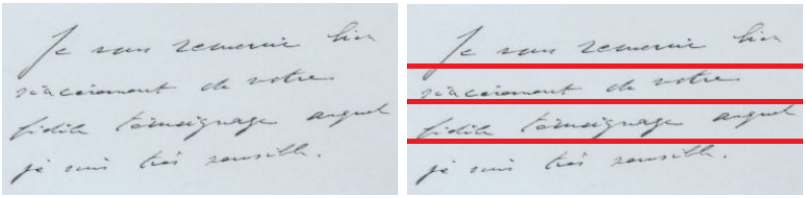


Fig. 3.1 – Illustration du problème de segmentation d'un paragraphe (gauche) en lignes de texte(droite).

3.1 Présentation du problème

Dans ce chapitre, nous cherchons à résoudre un problème de segmentation de paragraphes en lignes de texte. Cette segmentation est un des problèmes classiques d'analyse de documents et une étape traditionnelle de la reconnaissance pleine page, comme cela a été illustré dans la Figure 1.1.

En s'affranchissant de l'étape de détection des paragraphes, on simplifie le problème puisque l'on peut se contenter d'une segmentation mono-dimensionnelle selon l'axe vertical. Une illustration du problème que l'on cherche à résoudre peut être vue en Figure 3.1.

Néanmoins, deux difficultés majeures persistes. La première est liée à l'hétérogénéité des paragraphes que l'on cherche à segmenter. En effet, ces paragraphes peuvent être manuscrits ou imprimés, dans des langues différentes et sur des fonds et supports variés.

L'apprentissage automatique peut permettre de prendre en charge cette hétérogénéité mais requiert une quantité importante de données annotées. Or, si nous disposons de la position des paragraphes sur la base Maurdor que nous utilisons, il n'y a pas d'annotation au niveau de la position des lignes, ce qui représente la deuxième difficulté.

Comme nous disposons de l'annotation du texte contenu dans les paragraphes, et que les caractères de retour à la ligne ("\n") sont présents dans ces annotations, nous disposons cependant de l'annotation du nombre de lignes présentes dans les paragraphes.

On va donc chercher à résoudre un problème de prédiction de séquences où les éléments de la séquence ne sont pas explicitement localisés. Cela signifie que le système devra apprendre implicitement cette localisation.

Si on ajoute une prédiction d'interlignes dans cette séquence, on remarque que la problématique que nous cherchons à résoudre est fortement analogue au problème de reconnaissance de lignes de texte. En effet, dans ce cas aussi, on a des images de tailles variables, dans lesquelles on

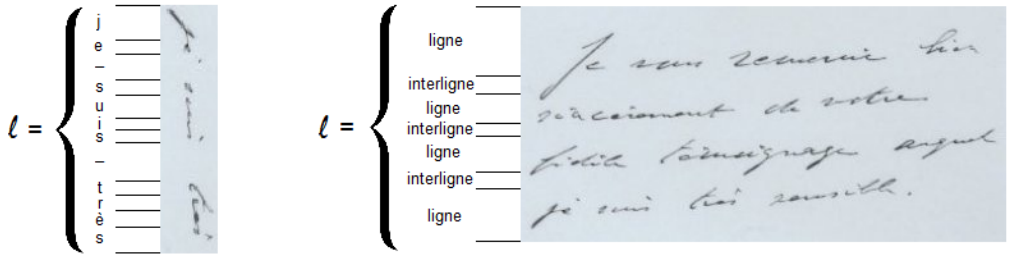


Fig. 3.2 – Illustration de l'analogie entre les problèmes de reconnaissance de ligne de texte (gauche) et de segmentation de paragraphe en lignes de texte (gauche). Sont illustrés les séquences $\boldsymbol{l}$ à prédire dans les deux cas. La segmentation explicite, donnée par les traits noirs n'est pas disponible et elle devra donc implicitement être apprise.

cherche à prédire des séquences mono-dimensionnelles de tailles variables d'objets, des caractères en l'occurence, objets dont la position n'est pas explicitement donnée. Donc, on retrouve le fait que pour pouvoir reconnaître ces caractères, le système doit implicitement en apprendre les positions. Cela revient à apprendre une segmentation implicite de l'image, cette segmentation est horizontale dans le cas de la reconnaissance de texte et verticale dans le cas de la segmentation de paragraphes. On peut visualiser cette analogie dans la Figure 3.2.

En se servant de cette analogie des problèmes, nous pouvons nous inspirer des solutions proposées au problème de reconnaissance de l'écriture pour résoudre le problème de segmentation des paragraphes en lignes de texte.

3.2 Reconnaissance de l'écriture : Prédiction des caractères

La reconnaissance de lignes de texte que nous utilisons dans cette thèse, et qui inspire le modèle de segmentation des paragraphes en lignes de texte proposé dans ce chapitre utilise un réseau de neurones récurrents à base de couches de 2D-LSTMs (*Two Dimensional Long Short-term Memory*) similaire à celui utilisé par Pham et al. [Pham et al., 2014]. Utiliser un réseau à base de 2D-LSTMs permet de prendre en compte des informations venant de toute l'image pour effectuer une prédiction locale. Par exemple, les tailles d'écriture et les polices sont souvent constantes dans une même ligne et la forme d'une lettre peut dépendre de ses voisines.

Ce réseau de neurones utilisé pour la reconnaissance des lignes est très fortement inspiré de celui présenté par Graves et al. [Graves and Schmidhuber, 2009] mais utilise les contibutions de [Pham et al., 2014] qui proposent une adaptation de certains paramètres, en particulier au niveau des tailles des filtres des couches convolutionnelles et d'ajouter pendant l'entraînement du *Dropout*

[Hinton et al., 2012] après chaque couche de récurrence.

Le réseau prend en entrée une image de taille variable, en niveaux de gris. Cette image va passer dans un réseau composé de trois couches de 2D-LSTMs. Les 2D-LSTMs seront décrites en détails dans la section 5.4.1. En particulier, il est important de noter que les récurrences sont effectuées dans les quatre directions (cf. Figure 5.3).

On notera que, la première couche de 2D-LSTM prend en entrée les valeurs d'un carré de 2 pixels de large. Cela correspond à la couche de *Tiling* indiquée dans le tableau 3.1. Après chacune des deux premières couches de 2D-LSTMs, une couche de convolution est ajoutée dans chacune des quatre directions de la récurrence. Des paramètres différents sont utilisés pour chacune des directions. On somme ensuite les cartes de caractéristiques (*feature maps*) des quatre directions et on applique a cette somme une non-linéarité sous la forme d'une tangente hyperbolique.

Après la troisième couche de 2D-LSTMs, dans chacune des quatre directions, une couche entièrement connectée est ajoutée. Le nombre de neurones en sortie de ces couches complètement connectées est égal au nombre N de labels de sortie (ie. le nombre de caractères dans l'alphabet utilisé). Les cartes de caractéristiques obtenues en sortie de ces quatre couches entièrement connectées sont sommées. Enfin, on transforme les cartes de caractéristiques en deux dimensions en vecteurs de caractéristiques en une dimension en sommant sur la direction relative à notre séquence, c'est à dire relative à la hauteur de l'image d'entrée pour la tâche de reconnaissance et relative à la largeur de l'image d'entrée pour notre tâche de segmentation des paragraphes en lignes de texte. On applique une fonction *Softmax* ($\sigma(z_j) = e^{z_j} / \sum_{n=1}^{N} e^{e_n}$ avec N le nombre de vecteurs de caractéristiques et $j \in \{1,...,N\}$) afin de rendre les prédictions du réseau homogènes à des probabilités.

Finalement, est obtenu une séquence de prédictions de sorties $\boldsymbol{y} = \{\boldsymbol{y_1},...,\boldsymbol{y_T}\}$ où pour chaque position horizontale j, $\boldsymbol{y_j} = \sigma(z_j)$. Le nombre T de positions, et donc également de prédictions, est variable. Il dépend de la taille de l'image d'entrée, variable, et de la taille des filtres et *strides* du réseau indiqués dans le tableau 3.1. Pour chaque position j de la séquence de prédictions, le vecteur de prédictions $\boldsymbol{y_j}$ est composé de N valeurs ($\boldsymbol{y_j} = \{y_j(1),...,y_j(N)\}$). Ces valeurs qui peuvent être assimilées à des probabilités sont associées aux N éléments de l'alphabet utilisés.

Cette architecture est illustrée en Figure 3.3. Les paramètres du réseau relatifs au nombre de neurones sur chaque couche cachée et à la taille des filtres sont indiqués dans le tableau 3.1 où l'on trouve aussi le nombre de paramètres libres par couche. On notera que le réseau contient un total de 134 147 paramètres libres, ce qui est plutôt faible au regard des standards de la communauté travaillant avec des réseaux profonds.

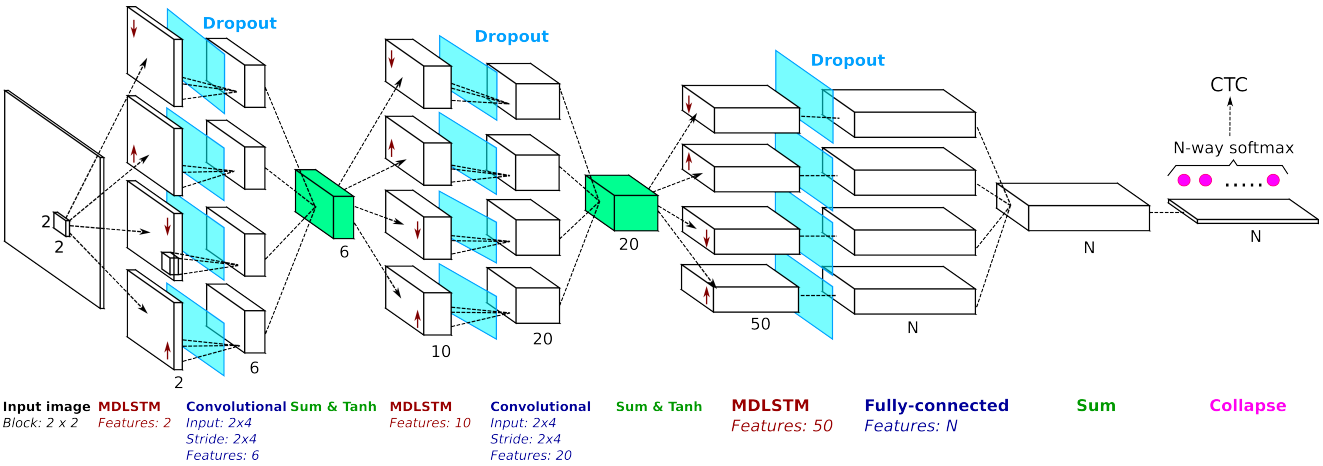


Fig. 3.3 – Illustration de l'architecture du réseau de neurones utilisé pour la reconnaissance de l'écriture et pour la segmentation de paragraphes en lignes de texte.

TABLE 3.1 – Nombre de neurones sur chaque couche cachée (et la couche de sorties) pour le réseau utilisé dans les tâches de reconnaissance de texte et de segmentation des paragraphes en lignes.

Couche :	Taille des filtres	Taille des *strides*	Nombre de neurones cachés	Nombre de paramètres libres
(0) Tiling	2×2	2×2	4	0
(1) LSTM			2	360
(2) Convolution	2×4	2×4	6	384
(3) LSTM			10	5400
(4) Convolution	2×4	2×4	20	6400
(5) LSTM			50	121000
(6) Linéaire			N	51xN

3.3 Reconnaissance de l'écriture : Entraînement et alignement dynamique

Ces réseaux de neurones utilisés pour la reconnaissance de lignes de texte sont entraînés par descente de gradients stochastique (SGD). La fonction de coût est définie dans la section 3.3.1 et du *Dropout* est appliqué sur chacune des couches de 2D-LSTMs avec une probabilité de 0,5. La technique du *Dropout* [Hinton et al., 2012] consiste à mettre aléatoirement, durant l'apprentissage, la moitié des neurones à zéro et sert à éviter la co-adaptation entre les neurones. Cela a pour effet de limiter le sur-apprentissage et d'aider le système à mieux généraliser.

3.3.1 Description de l'alignement avec la *Connectionist Temporal Classification* (CTC)

Pour entraîner ce réseau, un alignement est nécessaire puisque la séquence de sorties $\boldsymbol{y} = \{\boldsymbol{y_1},...,\boldsymbol{y_T}\}$ du réseau est d'une taille T différente (plus grande) de la taille K de la séquence de vérités terrains $\boldsymbol{l} = \{l_1,...,l_K\}$, c'est à dire de la séquence de lettres à reconnaitre. Pour cela, la technique de *Connectionist Temporal Classification* (CTC) [Graves et al., 2006] est utilisée.

On rappelle que chaque élément $\boldsymbol{y_i}$ de la séquence de prédiction est composé de N valeurs ($\boldsymbol{y_i} = \{y_i(1),...,y_i(N)\}$). Ces valeurs peuvent être assimilées à des probabilités puisqu'elles sont situées après une couche de *Softmax* et sont associées aux N éléments de l'alphabet utilisés. Les éléments de la séquence $\boldsymbol{l}$ sont choisis dans cet alphabet.

On peut définir l'ensemble des trajets $\boldsymbol{\pi} = \{\pi_1,...,\pi_K\}$ qui permettent d'associer les éléments de $\boldsymbol{l}$ aux éléments de $\boldsymbol{y}$. π_i étant l'index du label associé à la position i du chemin. Ces différents chemins $\boldsymbol{\pi}$ possibles sont illustrés par le graphe présenté en Figure 3.4.

La CTC va associer une probabilité P_π à chaque chemin $\boldsymbol{\pi}$ en multipliant les probabilités des labels associées à chaque position du chemin, comme indiqué dans l'équation 3.1.

$$P_\pi = \prod_{t=1}^{T} y_t(\pi_t) \tag{3.1}$$

La probabilité P_l totale de la séquence $\boldsymbol{l}$ est ensuite obtenue en sommant les probabilités P_π de tous les chemins possibles :

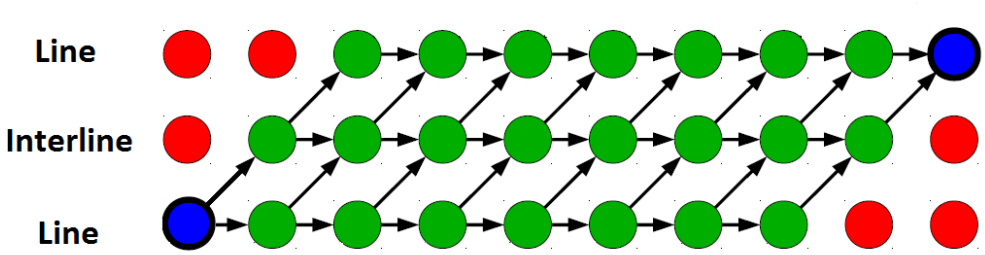


Fig. 3.4 – Illustration des trajets possibles lors de l'alignement CTC, dans le cadre de la détection de lignes. En bleu les points de départ et d'arrivée du graphe, en vert les états par lesquels le passage est autorisé et en rouge, les états interdits. L'axe horizontal est relatif à la séquence des différentes *frames* de l'image.

$$P_l = \sum_{\forall \boldsymbol{\pi}=\boldsymbol{l}} P_\pi = \sum_{\forall \boldsymbol{\pi}=\boldsymbol{l}} \prod_{t=1}^{T} y_t(\pi_t) \tag{3.2}$$

Pendant l'entraînement, on a un ensemble de couples $\{\boldsymbol{y}, \boldsymbol{l}\}$ et on va va chercher à modifier $\boldsymbol{y}$ pour maximiser la vraisemblance de $\boldsymbol{l}$ sachant $\boldsymbol{y}$. On peut donc minimiser la fonction de coût $C(\boldsymbol{y},\boldsymbol{l})$ de l'Equation 3.3. Comme tout est dérivable, cela peut se faire avec les techniques classiques d'optimisation par descente de gradient.

$$C(\boldsymbol{y},\boldsymbol{l}) = -ln(P_l) = -ln\left(\sum_{\forall \boldsymbol{\pi}=\boldsymbol{l}} \prod_{t=1}^{T} y_t(\pi_t)\right) \tag{3.3}$$

3.3.2 Ajout de "blancs" dans les séquences de labels pour la reconnaissance

Durant la phase d'inférence, pour chaque position t, seule la prédiction la plus forte va être conservée. Pour la segmentation de lignes de texte, toutes ces prédictions sont utiles puisqu'elles permettent de définir la position de la ligne ; plus de détails seront donnés en section 3.4.3. Mais pour la tache de la reconnaissance de l'écriture, la séquence de texte est ce qui nous intéresse. Comme $T \gg K$, plusieurs prédictions successives vont correspondre au même label. On agglutine donc les prédictions successives similaires pour retrouver la séquence voulue. Dans l'exemple suivant, avec a et b nos deux labels, on a :

$$aabbab \rightarrow abab$$

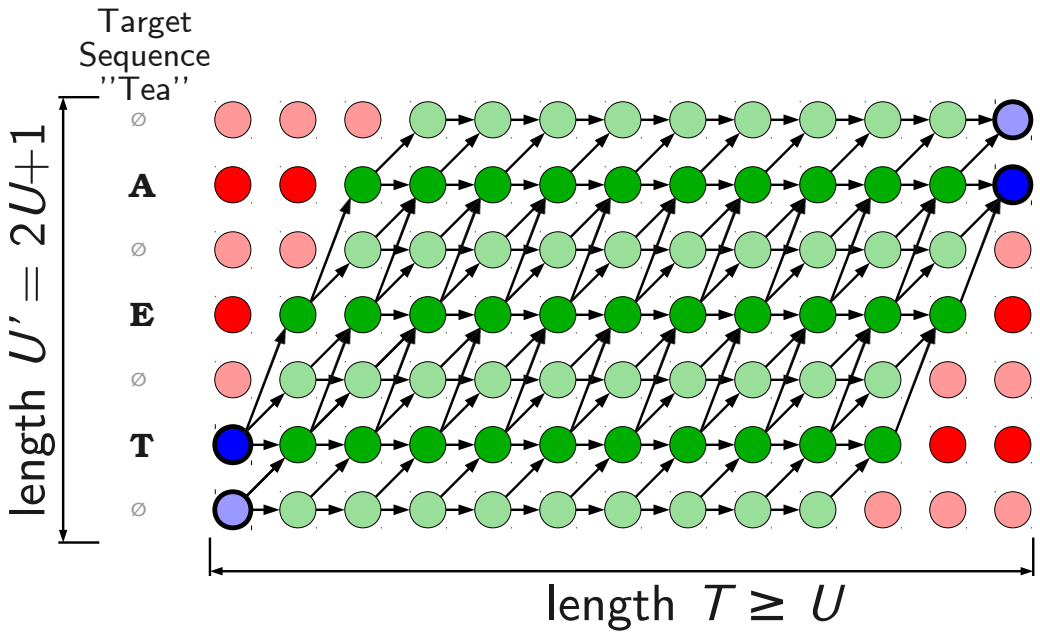


Fig. 3.5 – Illustration des trajets possibles lors de l'alignement CTC lorsque l'on ajoute des blancs.

Cependant, un problème survient lorsque la séquence voulue contient deux labels similaires consécutifs. En effet, un seul label pourra être produit à cause de l'agglutination. La solution traditionnelle à ce problème [Graves et al., 2006] est d'ajouter un label supplémentaire appelé "blanc" utilisé pour marquer les frontières entre les lettres. Une sortie est ajoutée au réseau et correspond à ce label. Lors de l'alignement, pour une vérité terrain $\boldsymbol{l}$ donnée, les chemins $\boldsymbol{\pi}$ pourront optionnellement passer par des prédictions de "blanc" entre les labels de la séquence $\boldsymbol{l}$. Ceci est illustré en figure 3.5 ou le label "blanc" est dénoté par $\varnothing$.

Lorsque des caractères similaires sont présents, comme illustré figure 3.6, ce passage par le label "blanc" devient obligatoire entre les labels consécutifs identiques lors de la construction des différents chemins $\boldsymbol{\pi}$ possibles.

Au décodage, ces labels "blanc" sont supprimés après l'agglutination et permettent donc au système de reconnaissance de prédire des labels consécutifs similaires. Toujours en notant a et b nos deux labels et $\varnothing$ le label "blanc", on a :

$$aa\varnothing\varnothing bb\varnothing ba \rightarrow abba$$

On notera que la présence de ces caractères "blanc" se traduit par la prédiction de pics très localisés pour la prédiction des autres labels (le label "blanc" est de très loin le caractère le plus prédit) et que, appliqué à une tâche de reconnaissance de l'écriture avec des réseaux récurrents, il permet de faciliter grandement la convergence des réseaux [Hannun et al., 2014] puisque il permet de faciliter l'alignement au début de l'entraînement [Bluche et al., 2016a]. Les auteurs

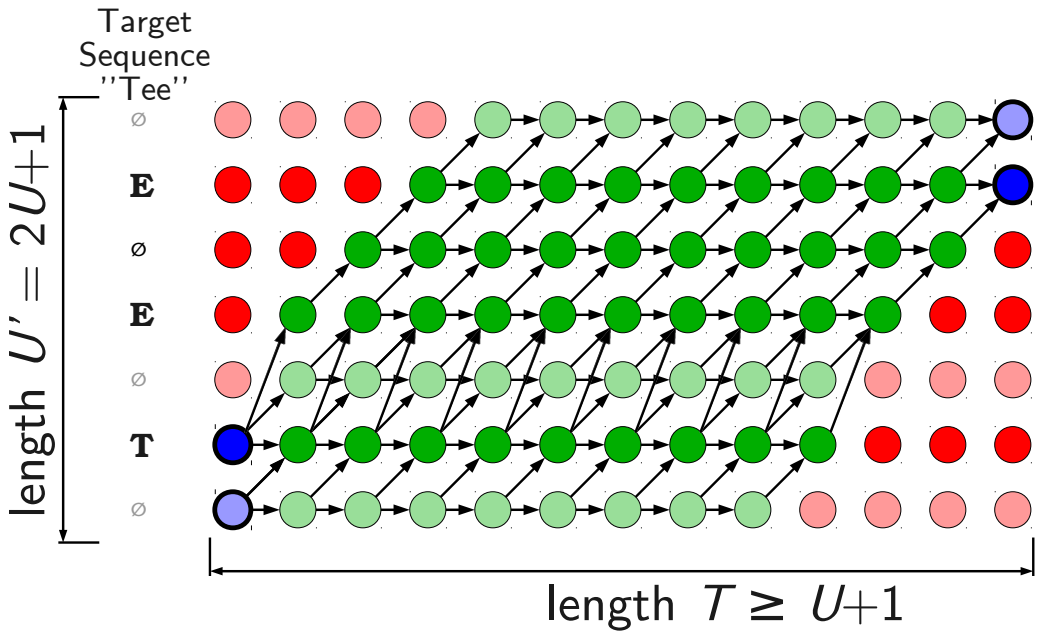


Fig. 3.6 – Illustration des trajets possibles lors de l'alignement CTC lorsque l'on ajoute des blancs, en présence de caractères consécutifs similaires.

[Graves et al., 2006] suggèrent aussi que ce caractère "blanc" pourrait permettre de représenter les espace sans caractères ou entre les caractères, formes de bruits qui ne correspondent à aucune classe.

3.4 Segmentation de paragraphes en lignes

Nous proposons d'utiliser un système similaire à celui utilisé pour la reconnaissance du texte et décrit dans les sections précédentes afin de réaliser une segmentation mono-dimensionnelle de nos paragraphes en lignes de texte.

3.4.1 Création des labels

L'objectif de notre technique est de prédire une séquence correspondant à la présence de lignes ou d'interlignes dans l'image de paragraphe. Le nombre de lignes (et donc le nombre d'interlignes) présentes dans chaque paragraphe nous est donné dans les annotations de la base Maurdor [Brunessaux et al., 2014] par la présence de caractères de fins de lignes ("\n") dans les annotations des textes contenus dans chaque paragraphe.

Utiliser une technique de reconnaissance de séquences au lieu d'une technique de classification,

avec l'alignement CTC, nous évite la nécessité d'annoter la position de chaque ligne. Le haut et le bas de paragraphes peuvent paraître similaires à des interlignes. Pour cette raison, il est légitime de se demander si on doit commencer (respectivement finir) nos séquences par un label interligne. On peut aussi se demander si la présence d'un label "blanc" est nécessaire ou si il peut être remplacé par la présence de labels interligne. Pour ces raisons, plusieurs générations de données différentes sont possibles. Elles sont illustrées en figure 3.8 et seront évaluées en section 3.5.

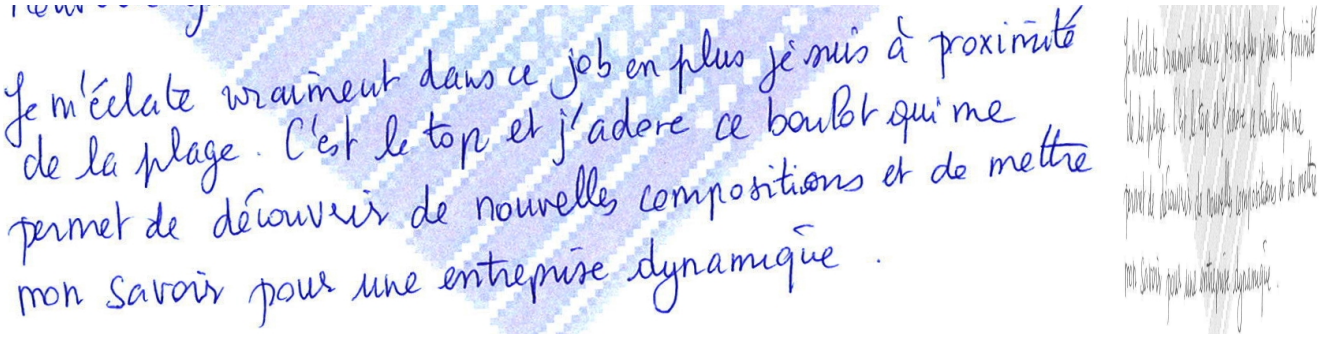
3.4.2 Pré-traitements

Plusieurs transformations de l'image d'entrée sont effectuées avant d'envoyer celle-ci au réseau de neurones qui va effectuer la segmentation des paragraphes. Un exemple du résultat de ces transformations est donné en figure 3.7. Tout d'abord, l'image est normalisée dans une résolution fixe de 300 points par pouce (dpi) pour tenir compte de documents venant de sources d'acquisition diverses. L'image est transformée en niveaux de gris, c'est aussi souvent le cas en reconnaissance de l'écriture [Pham et al., 2014]. Cela permet de réduire le nombre de caractéristiques en entrée du réseau et est peu dommageable pour la reconnaissance.

Comme notre segmentation est mono-dimensionnelle, elle souffre particulièrement lorsque les lignes de texte ne sont pas horizontales. Afin de régler ce problème, nous pré-traitons les paragraphes avec l'algorithme de Bloomberg et al. [Bloomberg et al., 1995] qui va trouver l'angle selon lequel la projection des pixels noirs va être maximale. L'image est redressée en fonction de cet angle pour que les lignes deviennent le plus horizontales possible.

Ensuite, et toujours comme dans la tâche de reconnaissance de l'écriture, les valeurs des caractéristiques de l'image sont normalisées en moyenne et en variance. Pour les paragraphes en arabe, une symétrie horizontale est effectuée afin que toutes les images de notre jeu de données voient leurs lignes de texte alignées du même côté (à gauche).

Enfin, de manière très spécifique à cette tâche, l'image est normalisée en largeur de manière anisotropique. La largeur de l'image va être fortement réduite à une valeur fixe alors que la hauteur va rester inchangée. Cette réduction horizontale de la taille de l'image est nécessaire pour réduire la taille globale de l'image et donc le temps de calcul. Garder une taille verticale inchangée permet de garder suffisamment de résolution dans cette direction qui est la plus critique puisque c'est celle selon laquelle on veut segmenter. Une valeur de 200 pixels s'est, expérimentalement, avérée être un bon compromis entre performances et temps de calculs. On remarque en figure 3.7b que après ces pré-traitements, et malgré la contraction horizontale effectuée, les lignes restent visuellement bien séparées et identifiables pour une image typique de paragraphe de document.



(a) (b)

Fig. 3.7 – a) Image initiale du paragraphe. b) Image, pré-traitée, donnée en entrée du réseau pour la segmentation des paragraphes en lignes.

3.4.3 Post-Traitements

Pour chaque position verticale, le réseau va prédire la présence de ligne, d'interligne ou de "blanc". Puisque le réseau est récurrent et que l'entraînement est effectué avec la CTC pour aligner, le réseau n'apprend pas la position exacte des lignes. Il est possible que le réseau converge vers une prédiction systématique du haut des lignes, du bas des lignes ou du centre de celles ci. La taille des interlignes par rapport aux lignes peut également varier. Pour cette raison, et comme illustré en Figure 3.8, on va prendre pour chaque ligne hypothèse, à la fois les positions successives pour lesquelles la sortie ligne a été prédite mais également les positions adjacentes ou il a été prédit interligne ou "blanc". Cela signifie que les zones prédites comme interlignes ou "blanc" vont être incluses à la fois dans les boites des lignes qui leurs sont supérieures et des lignes qui leur sont inférieures. Ceci est particulièrement intéressant pour les paragraphes manuscrits puisque d'une part, les ascendants et les descendants de lignes successives se recouvrent fréquemment, et que d'autre part, les réseaux récurrents utilisés lors de la reconnaissance de l'écriture se montrent robustes à la présence de ces ascendants et descendants venant de lignes voisines.

3.5 Expériences et résultats

Plusieurs expériences ont été effectuées. Elles visent à comprendre le comportement de notre système, à effectuer certains choix au niveau de la conception de notre système et, enfin, à en comparer les performances par rapport à d'autres techniques de segmentation de paragraphes en lignes.

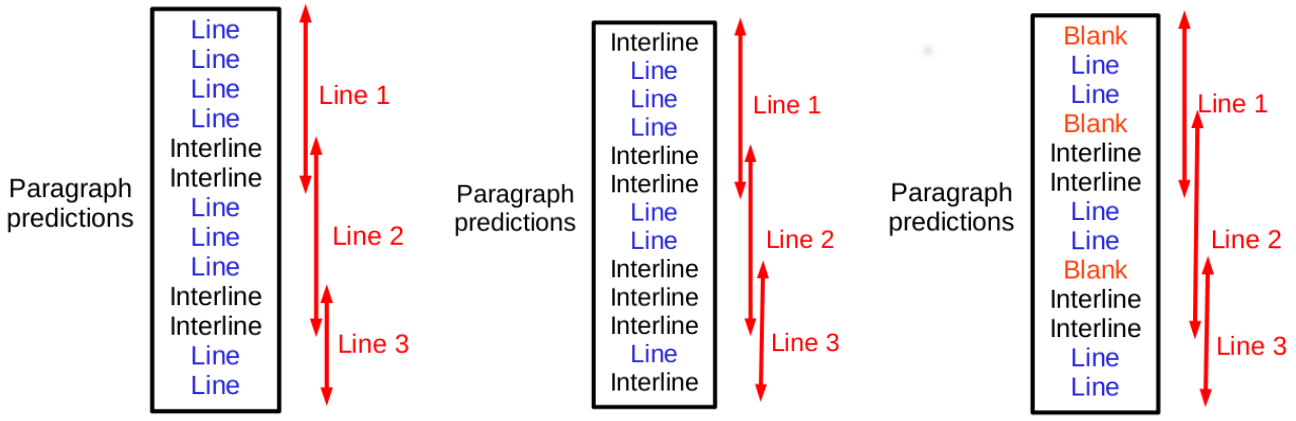


Fig. 3.8 – Illustration des stratégies de construction des labels et des résultats associés : a) En commençant et en finissant par le label ligne. b) En commençant et en finissant par le label interligne. c) En ajoutant une sortie pour le label "blanc". Et illustration de la construction des boîtes de lignes résultats à partir de ces prédictions.

3.5.1 Cadre des expériences

Nos expériences ont été faites sur les paragraphes de la base Maurdor [Brunessaux et al., 2014] qui sera détaillée de manière plus approfondie en section 7.1.1. Cette base contient à la fois des documents en français, en anglais et en arabe ; à la fois imprimés et manuscrits.

Deux métriques seront utilisées dans cette section :

D'abord, le taux de reconnaissance mot (WER), qui sera détaillé en section 7.2, mesure la qualité de la segmentation en lignes effectuée à l'aune de la capacité d'un système de reconnaissance de texte à en reconnaître le contenu. Il sera utilisé dans les tableaux 3.2 et 3.3 pour comparer les stratégies de construction des labels et dans les tableau 3.7 et 3.8 pour comparer notre technique à des méthodes plus traditionnelles.

Ensuite, une métrique spécifique à cette section, où l'erreur correspond à la proportion de paragraphes dans lesquels le bon nombre de lignes n'a pas été trouvé. Cette métrique ne dépend pas de la position des lignes, juste du nombre de lignes détectées. Elle est directement liée à la fonction de coût utilisée pour l'entraînement de nos réseaux de neurones et sera utilisée dans les tableaux 3.4, 3.5 et 3.6 pour comparer différents réseaux entre eux.

3.5.2 Modélisation des interlignes

Comme mentionné en section 3.4.1, plusieurs choix sont possibles pour la construction des labels utilisés lors de l'entraînement des réseaux. Dans le tableau 3.2 sont montrés les résultats de deux réseaux de neurones. L'un possède deux sorties, respectivement pour modéliser la présence d'une ligne et d'un interligne. Au second, un caractère "blanc" (cf. section 3.3.2) a été ajouté. On observe que, comme pour la reconnaissance de l'écriture [Bluche et al., 2016a], utiliser le "blanc" permet d'améliorer légèrement les performances.

TABLE 3.2 – Comparaison des taux d'erreur mots de systèmes entraînés avec et sans label "blanc" sur l'ensemble de validation de la base Maurdor Français manuscrit et imprimé.

Réseau	Français manuscrit	Français imprimé
RNN sans label "blanc"	21.43%	8.01%
RNN avec label "blanc"	19.45%	7.58%

Le tableau 3.3 compare deux réseaux. Le premier a utilisé comme labels lors de l'apprentissage des séquences commençant et terminant par les labels ligne. Le deuxième a utilisé des séquences commençant et terminant par les labels interligne. On observe que le premier, sans interlignes sur les bords des séquences d'entraînement, est meilleur. Nous pouvons supposer que cela est dû au fait que le réseau a de cette manière plus de liberté pour interpréter les marges supérieures et intérieures. La classe interligne du réseau n'est pas perturbée avec des hauts et des bas d'apparences très variées.

TABLE 3.3 – Comparaison des taux d'erreur mots de systèmes entraînés avec et sans label interligne ajoutés au début et à la fin des séquences d'entraînement. Evaluation sur les bases de validation de Maurdor Français manuscrit et imprimé.

Réseau	Français manuscrit	Français imprimé
Séquences d'entraînement commençant et terminant par des labels ligne	19.45%	7.58%
Séquences d'entraînement commençant et terminant par des labels interligne	29.15%	19.21%

3.5.3 Spécialisation du réseau

Les paragraphes de la base Maurdor sont homogènes en langue et en type de texte (imprimé ou manuscrit). Les pages qui les contiennent sont aussi divisés en 5 catégories de documents détaillées dans le tableau 7.2. Si les réseaux mentionnés précédemment ont été entraînés sur l'ensemble de ces

TABLE 3.4 – Comparaison des proportions de paragraphes avec le nombre correct de lignes détectées entre des réseaux entraînés sur des types d'écriture spécifiques ou entraîné sur l'ensemble des types. Evaluation sur les bases type-spécifiques de l'ensemble de validation de la base Maurdor.

Base d'évaluation	RNNs spécialisés	RNN générique
Manuscrits	4.20%	3.83%
Imprimés	1.50%	2.58%

TABLE 3.5 – Comparaison des proportions de paragraphes avec le nombre correct de lignes détectées entre des réseaux entraînés sur des scripts d'écriture spécifiques ou entraîné sur l'ensemble des scripts. Evaluation sur les bases script-spécifiques de l'ensemble de validation de la base Maurdor.

Base d'évaluation	RNNs spécialisés	RNN générique
Arabe manuscrit	3.16%	3.60%
Latin manuscrit	4.09%	3.92%
Arabe imprimé	1.21%	2.23%
Latin imprimé	2.58%	2.67%

documents, donc à la fois sur des documents manuscrits et imprimés dans plusieurs langues, il est possible d'entraîner des réseaux spécifiques au prix d'un ensemble d'entraînement plus restreint.

Les réseaux de cette section sont entraînés avec un label "blanc" et avec des séquences d'entraînement commençant et finissant par un label ligne.

Dans le tableau 3.4, nous comparons des réseaux entraînés avec un type d'écriture spécifique avec le réseau générique entraîné à la fois sur de l'imprimé et du manuscrit. La spécialisation au niveau du type permet d'améliorer d'environ 40% les performances pour les paragraphes imprimés mais causent une dégradation des performances de 9% pour les paragraphes manuscrits.

Dans le tableau 3.5, des réseaux sont spécialisés suivant le script de la langue. L'un est entraîné avec les paragraphes en langues latines (français et anglais ici) et l'autre est entraîné avec les paragraphes en arabe seulement. Ils sont, de nouveau, comparés au réseau générique. De manière générale, spécialiser les réseaux permet d'améliorer légèrement les performances.

Et dans le tableau 3.6, des réseaux entraînés sur des paragraphes venant de pages de documents de la même catégorie sont comparés avec le réseau générique. En général, ces spécialisations s'avèrent utiles.

La plupart des spécialisations apportent une légère amélioration des résultats par rapport au réseau générique. Cette amélioration peut être expliquée par le fait que la diversité à l'intérieur de la base est plus faible ce qui signifie que la tâche d'apprentissage est plus facile. C'est particulièrement vrai pour l'imprimé (ou les polices sont peu variées) et pour les textes en arabe qui

TABLE 3.6 – Comparaison des proportions de paragraphes avec le nombre correct de lignes détectées entre des réseaux entraînés sur des catégories de documents spécifiques ou entraîné sur l'ensemble des catégories. Evaluation sur les bases catégorie-spécifiques de l'ensemble de validation de la base Maurdor.

Base d'évaluation	RNNs spécialisés	RNN générique
Catégorie 1	0.95%	1.26%
Catégorie 2	3.69%	4.45%
Catégorie 3	8.76%	10.18%
Catégorie 4	2.93%	2.24%
Catégorie 5	2.70%	2.99%

sont minoritaires par rapport au latin.

Mais cela signifie aussi que l'ensemble d'entraînement est plus restreint, ce qui peut compliquer la généralisation, en particulier sur la tâche plus compliquée de segmentation de lignes manuscrites.

3.5.4 Comparaison vis à vis d'autres techniques de segmentation en paragraphes

Dans cette section nous comparons notre approche de segmentation de paragraphes en lignes de texte à base de réseaux de neurones à d'autres techniques de référence issues de la littérature. Un réseau générique est utilisé pour segmenter l'ensemble des paragraphes. Les images des lignes de texte prédites par le réseau (et son post-traitement décrit en section 3.4.3) sont envoyées à des reconnaisseurs de texte dont l'architecture et l'entraînement sont ceux décrits en sections 3.2 et 3.3 et détaillés dans [Moysset et al., 2014a].

Les autres méthodes de segmentation de paragraphes utilisées sont :

— Une adaptation de Shi et al. [Shi et al., 2009] qui floute l'image avec des filtres ellipsoïdaux orientés et assimile, après binarisation, les composantes connexes aux lignes.
— Une adaptation de Nicolaou et al. [Nicolaou and Gatos, 2009] qui suit les vallées sans texte en fonction de la densité des pixels s'y trouvant.
— Les meilleurs détecteurs de lignes pour chaque sous-ensemble utilisés dans le système A2iA lors de la compétition Maurdor [Moysset et al., 2014a].
— Une combinaison de détecteurs de lignes décrite dans [Moysset et al., 2014a].

Les évaluations sont effectuées sur l'ensemble de test de la base Maurdor et un taux d'erreur mot est utilisé comme métrique. Elles sont visibles dans les tableaux 3.7 et 3.8. On peut voir que notre approche à base de réseaux récurrents obtient des résultats meilleurs que chacun des détecteurs de lignes auxquels nous l'avons comparée, et ce pour chacune des langues (français, anglais et arabe) et pour les deux types d'écriture (manuscrit et imprimé).

Les résultats obtenus sont meilleurs que la combinaison de détecteurs utilisée dans [Moysset et al., 2014a] pour l'imprimé mais moins bon pour le manuscrit. Néanmoins, nous noterons que cette combinaison nécessite de faire la reconnaissance du texte plusieurs fois alors que notre méthode n'en nécessite qu'une ce qui permet un gain significatif au niveau du temps de calculs.

TABLE 3.7 – Comparaison des taux d'erreur mots obtenus avec différents algorithmes de segmentation de paragraphes en lignes. Évaluation sur les paragraphes manuscrits de l'ensemble de test de la base Maurdor.

Détecteur de ligne	Manuscrit		
	Français	Anglais	Arabe
Réseau récurrent présenté dans ce chapitre	25.20%	36.01%	31.04%
Adaptation de Nicolaou et al. [Nicolaou and Gatos, 2009]	27.04%	41.19%	34.45%
Adaptation de Shi et al. [Shi et al., 2009]	37.16%	44.88%	40.01%
Système de la compétition Maurdor[Moysset et al., 2014a] Meilleur détecteur de lignes	25.6%	42.7%	33.3%
Système de la compétition Maurdor[Moysset et al., 2014a] Avec alternatives de segmentations	**22.2%**	**35.2%**	**29.8%**

TABLE 3.8 – Comparaison des taux d'erreur mots obtenus avec différents algorithmes de segmentation de paragraphes en lignes. Évaluation sur les paragraphes imprimés de l'ensemble de test de la base Maurdor.

Détecteur de ligne	Imprimé		
	Français	Anglais	Arabe
Réseau récurrent présenté dans ce chapitre	**9.44%**	**8.81%**	**18.88%**
Adaptation de Nicolaou et al. [Nicolaou and Gatos, 2009]	11.49%	11.49%	28.66%
Adaptation de Shi et al. [Shi et al., 2009]	27.85%	39.59%	31.35%
Système de la compétition Maurdor[Moysset et al., 2014a] Meilleur détecteur de lignes	18.2%	15.9%	22.2%
Système de la compétition Maurdor[Moysset et al., 2014a] Avec alternatives de segmentations	11.3%	12.8%	22.8%

3.5.5 Illustration de résultats

Une illustration de résultats obtenus pour une image de lettre manuscrite est donnée figure 3.9. Les boites rouges correspondent aux boites englobant les paragraphes, considérées comme connues, elles correspondent aux images envoyées en entrée de notre système de segmentation des

paragraphes en lignes de texte. A la gauche de ces boites est donnée la séquence des labels prédits par le réseau avec un 'l' pour les lignes et un 'b' pour les interlignes ou les labels "blanc". Les boîtes bleues correspondent aux boites englobant les lignes prédites.

3.6 Conclusion

Dans ce chapitre, nous avons décrit le fonctionnement et l'entraînement d'un système de reconnaissance de lignes de texte. Cela nous a permis d'introduire une nouvelle méthode de segmentation de paragraphes de texte en lignes utilisant des réseaux récurrents à base de 2D-LSTMs et ne nécessitant pas l'annotation des positions des lignes de texte. Cette technique s'est montrée capable de fonctionner au niveau de l'état de l'art sur une large variété de documents, sans adaptation intensive.

Mais cette segmentation est mono-dimensionnelle puisqu'elle n'est faite que selon la direction verticale. Cela signifie que la méthode rencontre des problèmes, comme illustré en figure 3.9, lorsque les lignes n'ont pas toutes exactement la même orientation. Plus grave, cela signifie que la méthode ne pourra fonctionner que sur des paragraphes ne contenant qu'une colonne. Ce qui empêche cette méthode d'être appliquée directement à des pages complètes dans des bases hétérogènes complexes.

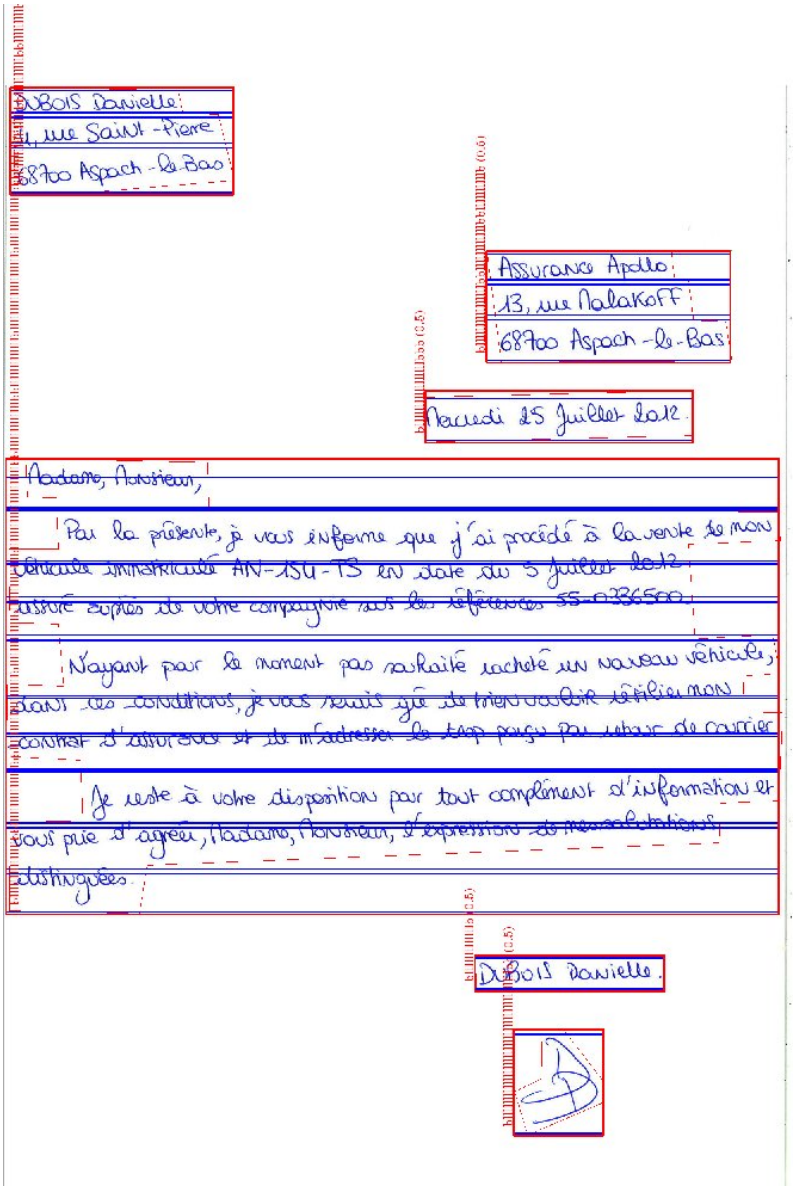


Fig. 3.9 – Illustration de résultats sur une page de lettre manuscrite. Les 'l' correspondent aux prédictions des labels ligne et les 'b' correspondent aux labels interligne. Les boites bleues correspondent aux lignes produites par le système.

Chapitre 4

Étude, description, analyse et comparaison de systèmes de détection d'objets par régression des coordonnées

Si la technique présentée dans le chapitre 3 permet de correctement segmenter les paragraphes de texte et présente l'avantage de ne pas nécessiter de connaître la position explicite des lignes de texte, le caractère mono-dimensionnel de cette segmentation complique la généralisation à des pages complètes dans lesquelles plusieurs paragraphes ou colonnes peuvent être présents.

Nous allons donc proposer une méthode basée sur la régression directe des coordonnées des boîtes englobant les lignes de texte.

La tâche de détection des lignes de texte dans des documents hétérogènes, si elle présente des spécificités qui nous conduirons à présenter les contributions des Sections 5 et 6, a aussi des points communs avec les tâches de détection d'objets dans des scènes naturelles. Notamment, ces tâches partagent la nécessité de prédire un nombre variable d'objets, la possibilité de localiser ces objets à partir des boîtes qui les englobent et le besoin d'être capable de trouver des objets de tailles et de formes variables. Cette similarité des tâches explique pourquoi les méthodes que nous avons développées s'apparentent fortement à des méthodes récentes de détection d'objets dans des images de scènes naturelles, tout en ajoutant des contributions liées aux enjeux de ce projet. Elles sont directement inspirées de MultiBox [Erhan et al., 2014] et s'apparentent sur certains points à YOLO [Redmon et al., 2016].

Dans ce chapitre, nous détaillons ces deux méthodes afin de poser les bases et les notations que nous utiliserons dans les parties suivantes. Cela nous permettra de souligner les contributions au modèle que nous apporterons dans les sections 5 et 6. Nous montrons également les spécificités

et les points communs de ces deux modèles et analysons leur fonctionnement.

La section 4.1 montre les similarités entre les deux méthodes au niveau des architectures des réseaux utilisés et au niveau des objets prédits et la section 4.2 les compare au niveau de la méthode d'entraînement et en particulier au niveau de la fonction de coût. Au contraire, la section 4.3 souligne la différence majeure entre ces méthodes qui se trouve au niveau de l'appariement entre les objets références et les objets hypothèses.

4.1 Prédiction des objets

MultiBox comme YOLO ont pour objectif principal de prédire la position des boîtes englobant les objets présents dans une image et ce, à partir des pixels de cette image. Pour ce faire, ces algorithmes ont en commun d'utiliser des réseaux de neurones convolutionnels $\mathcal{F}$ qui s'appliquent directement aux pixels de l'image d'entrée $\boldsymbol{Im}$ et ont pour sortie un nombre fixe de réels permettant de définir un nombre fixe B de boîtes englobantes d'objets candidats. Ces réseaux de neurones sont fonctions de paramètres libres $\boldsymbol{\Theta}$ qui seront appris. Chaque objet candidat $\boldsymbol{O_b}$ avec $b \in \{1, ..., B\}$ est défini par cinq réels. Les quatre premiers correspondent aux quatre coordonnées relatives à la position de la boîte (gauche, droite, haut, bas), on les notera $l_{i,b}$ avec $i \in \{0, ..., 3\}$. Le cinquième correspond à une confiance dans le fait que cet objet existe, on le notera c_b.

$$\boldsymbol{O_b} = \{l_{0,b}, l_{1,b}, l_{2,b}, l_{3,b}, c_b\} \tag{4.1}$$

De manière générale, YOLO et MultiBox visent tous deux à résoudre un problème de la forme :

$$\boldsymbol{O_1}, ..., \boldsymbol{O_B} = \mathcal{F}(\boldsymbol{Im}, \boldsymbol{\Theta}) \tag{4.2}$$

Ce problème est également illustré en Figure 4.1 où l'on voit la transformation, à travers un réseau convolutionnel, d'une image en une liste d'objets hypothèses.

4.1.1 Les réseaux et leurs architectures

Dans cette section nous décrivons l'architecture des réseaux utilisés par ces deux techniques qui partagent une approche convolutionnelle.

Les deux méthodes sont appliquées au même problème, à savoir la détection d'objets dans des scènes naturelles et en particulier aux tâches de localisation d'objets de Pascal VOC

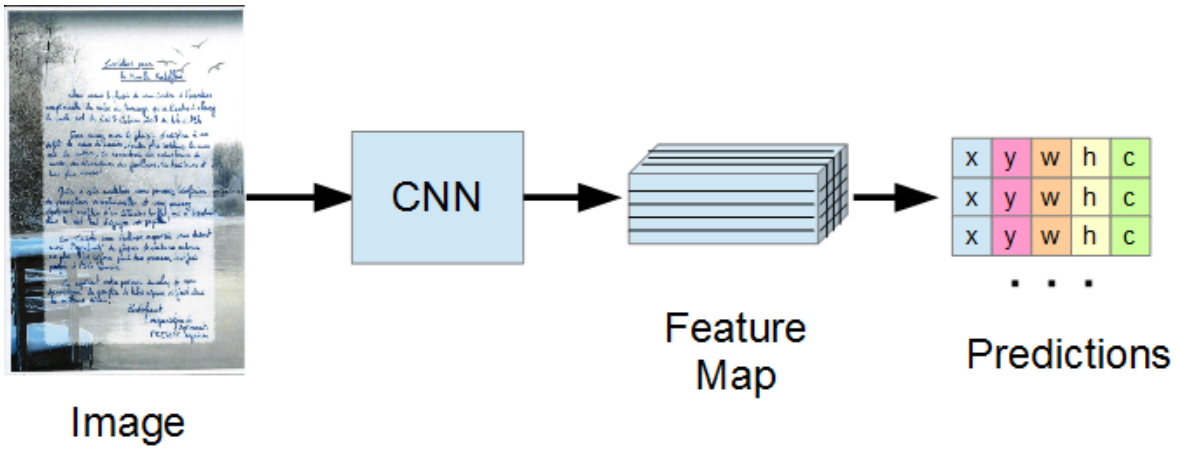


Fig. 4.1 – Illustration de l'utilisation d'un réseau de neurones convolutionnel pour la détection d'objets à base de régression directe des coordonnées.

[Everingham et al., 2010] et d'ILSVRC [Russakovsky et al., 2015]. Pour ces deux méthodes, l'entrée des réseaux est similaire. Il s'agit d'images en couleur, redimensionnées en taille fixe (224x224 pour MultiBox et 448x448 pour YOLO) et dont les pixels couleurs sont directement fournis comme caractéristiques en entrée du réseau.

Les deux méthodes utilisent des réseaux de neurones à base de couches convolutionnelles. Certaines de ces couches convolutionnelles sont suivies de couches de *max-pooling* pour réduire la dimensionnalité. A la fin du réseau, on place des couches complètement connectées qui prédisent de manière globale les valeurs de sortie que sont la position des boîtes hypothèses et la confiance en leur existence.

Pour MultiBox [Erhan et al., 2014], l'architecture est basée sur AlexNet [Krizhevsky et al., 2012]. Elle est donc composée de cinq couches convolutionnelles, trois couches de *max-pooling* et trois couches complètement connectées. Ce réseau comporte un total d'environ 61 millions de paramètres libres dont environ 58 millions au niveau des couches complètement connectées.

Pour YOLO [Redmon et al., 2016], l'architecture est inspirée de celle de GoogleNet [Szegedy et al., 2015a] pour la taille et le nombre des filtres mais sans la mécanique d'*inception*, utilisant donc une architecture similaire à celle décrite dans Lin et al [Lin et al., 2013]. Elle est composée de 24 couches convolutionnelles dont presque la moitié avec des filtres de taille 1×1 permettant de réduire le nombre de paramètres. Quatre couches de *max-pooling* sont disposées entre certaines de ces couches convolutionnelles et deux couches complètement connectées se trouvent à la fin du réseau. Ce réseau comporte environ 271 millions de paramètres libres dont 211 sur les couches complètement connectées. On notera qu'un réseau plus petit (9 couches de convolutions) est aussi proposé dans l'article YOLO [Redmon et al., 2016] avec environ 45 millions de paramètres libres (dont 19 millions sur les couches complètement connectées).

Les différences entre les deux réseaux au niveau des choix d'architectures s'expliquent principalement par l'évolution des techniques en *Deep Learning* au cours des deux ans qui séparent les publications de ces travaux. MultiBox a d'ailleurs été par la suite adapté avec un réseau basé sur GoogleNet [Szegedy et al., 2015a] [Szegedy et al., 2015b].

On notera cependant que des sorties supplémentaires sont ajoutées dans YOLO pour déterminer la classe de l'objet en même temps que l'on détermine son existence et sa position. Cette possibilité avait été mentionnée dans MultiBox [Erhan et al., 2014] mais pas expérimentée. S'il s'agit d'une des contributions principales de YOLO [Redmon et al., 2016], notre tâche n'ayant qu'une classe d'objets, nous n'approfondirons pas cette aspect.

4.1.2 Prédictions globales ou locales

La première différence importante entre les deux méthodes se situe au niveau de la nature des objets prédits. Là où MultiBox prédit dans un référentiel global, c'est à dire dans le référentiel de l'image, YOLO va prédire des objets dans un référentiel local, comme cela est décrit ci-dessous. Par contre, elles partagent, contrairement aux méthodes que nous introduirons en Section 5, le fait d'utiliser des caractéristiques globales pour prédire ces objets. Nous montrons dans cette section que seule une transformation linéaire différencie ces deux méthodes. Mais cela illustre une conception différente des objets qui se retrouvera dans la fonction de coût de la Section 4.2 et dans l'appariement de la Section 4.3.

Au niveau des réels prédits, pour MultiBox, on a bien, comme décrit en Section 4.1, un nombre fixe $B = 200$ de 5-uplets à prédire qui correspondent aux quatre coordonnées de la boîte englobant les objets et la confiance. Une fonction sigmoïde est appliquée aux sorties de confiance pour les borner entre 0 et 1. Les variables de position sont normalisées par rapport à la taille de l'image.

Les choses sont légèrement différentes pour YOLO. Premièrement, on ne cherche pas à prédire les coordonnées de la boîte englobante mais, à la place, on va prédire la position de son centre ainsi que les racines carrées de la largeur et de la hauteur de l'objet dans le but, d'après les auteurs, de donner plus d'influence aux petites boîtes. Deuxièmement, l'image est découpée en une grille de taille $S \times S$ avec $S = 7$. Les B boîtes prédites par le réseau sont associées arbitrairement et de manière fixe à l'une des positions dans l'image. Il y a $K = 2$ boîtes associées à chaque position. Donc $B = K \times S \times S = 98$. C'est dans ce référentiel relatif aux cases de la grille que les centre des boîtes sont définis, et normalisés, alors que la hauteur et la largeur sont normalisées par rapport à l'image complète.

Donc, en s'inspirant de l'équation 4.1, et en utilisant la notation prime ($'$) pour distinguer la formulation locale de YOLO de la formulation globale introduite dans les équations 4.2 et 4.1, on

peut définir les objets $\boldsymbol{O'_b}$ prédits par YOLO :

$$\boldsymbol{O'_b} = \{l'_{0,b}, l'_{1,b}, l'_{2,b}, l'_{3,b}, c_b\} \tag{4.3}$$

Avec :

$$l'_{0,b} = ((l_{0,b} + l_{1,b})/2 - \delta_x(b)) \times S \tag{4.4}$$

$$l'_{1,b} = ((l_{2,b} + l_{3,b})/2 - \delta_y(b)) \times S \tag{4.5}$$

$$l'_{2,b} = \sqrt{l_{1,b} - l_{0,b}} \tag{4.6}$$

$$l'_{3,b} = \sqrt{l_{3,b} - l_{2,b}} \tag{4.7}$$

où $\delta_x(b)$ et $\delta_y(b)$) correspondent aux décalages, respectivement verticaux et horizontaux, relatifs à la cellule de la grille associés à la boîte b. En particulier, si la boîte b est associée à la cellule i,j avec $i \in 0, ..., S-1$ et $j \in 0, ..., S-1$, on a :

$$\delta_x(b) = i/S \tag{4.8}$$

$$\delta_y(b) = j/S \tag{4.9}$$

En définissant $\mathcal{G}$ qui transforme du repère global au repère local, c'est à dire qui transforme $\boldsymbol{O}$ en $\boldsymbol{O'}$, le réseau de YOLO transforme donc $\boldsymbol{Im}$ en $\mathcal{F}'(\boldsymbol{Im},\boldsymbol{\Theta})$ et :

$$\mathcal{F}'(\boldsymbol{Im},\boldsymbol{\Theta}) = \mathcal{G}(\mathcal{F}(\boldsymbol{Im},\boldsymbol{\Theta})) \tag{4.10}$$

Comme $\mathcal{G}$ est linéaire, on a $\mathcal{F}'$ et $\mathcal{F}$ de complexité similaire et les réseaux peuvent être considérés comme similaires comme illustré en Figure 4.2.

4.1.3 Décodage

Pour les deux architectures, la phase d'inférence est similaire. Les boîtes hypothèses sont sélectionnées par un seuillage sur le terme de confiance. La position est, si besoin, corrigée pour s'assurer que la boîte est entièrement dans l'image de départ. Enfin, une suppression des non maxima est effectuée pour filtrer d'éventuels objets détectés plusieurs fois.

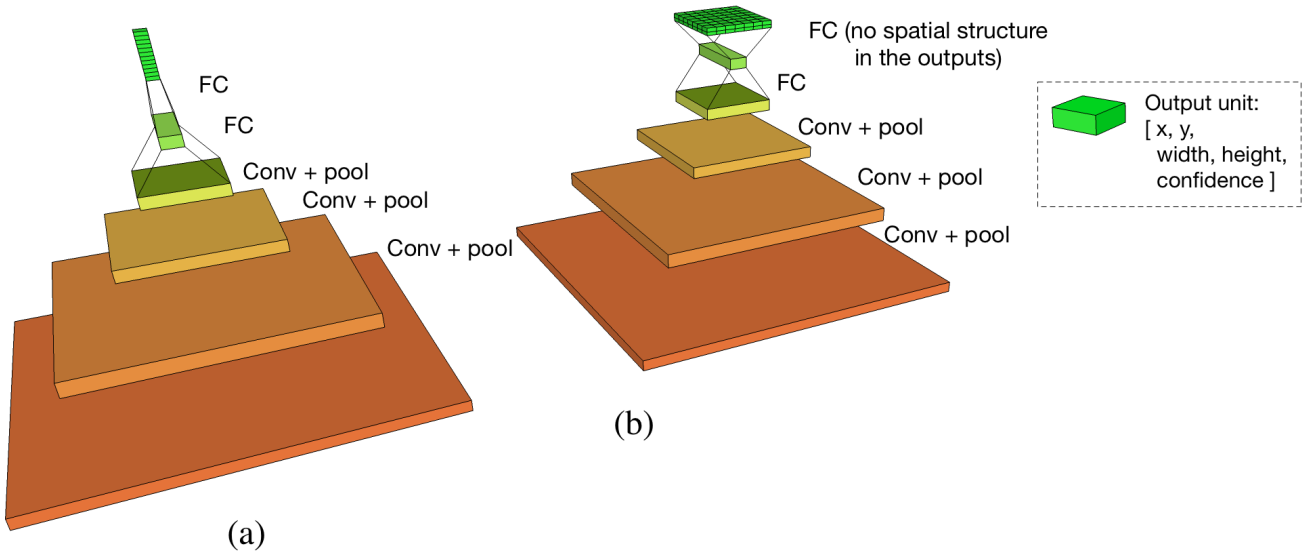


Fig. 4.2 – Schématisation des architectures de : a) MultiBox et b) YOLO. On distingue que les deux architectures sont constituées de couches convolutionnelles suivie de couches complètement connectées transformant la dernière carte de caractéristiques en prédictions ce qui souligne leurs points communs.

4.2 Entraînement des réseaux de localisation d'objets

Dans cette section, nous analysons la manière dont les réseaux des deux méthodes sont entraînés. Nous étudions les fonctions de coût utilisées pour la rétro-propagation des gradients et montrons leur similarité. Cette fonction de coût dépend d'un appariement entre objets références et objets hypothèses qui, lui, diffère fortement entre les deux méthodes et sera détaillé dans la Section 4.3 suivante.

L'entraînement est, dans les deux cas, effectué avec AdaGrad [Duchi et al., 2011] pour fonction d'optimisation. Des *minibatchs* de taille 64 sont utilisés. La technique du *Dropout* [Hinton et al., 2012] est utilisée sur l'avant dernière couche afin de limiter les co-adaptations et donc le sur-apprentissage. Enfin, un *weight decay* est également utilisé dans cette même optique de limiter le sur-apprentissage.

Les fonctions de coût C, qui sont dérivées pour obtenir les gradients de sortie à rétropropager lors de l'apprentissage, différent légèrement mais restent relativement similaires et de la forme :

$$C = \alpha C_{pos} + C_{conf} \tag{4.11}$$

Avec C_{pos} le coût relatif aux prédictions des positions et C_{conf} le coût relatif aux prédictions des confiances. L'hyperparamètre α permet de pondérer ces deux coûts. En particulier, on va reprendre la définition des B boîtes hypothèses $\boldsymbol{O_b}$ de l'équation 4.1 avec $b \in \{1, ..., B\}$ et on va

similairement définir dans l'équation 4.12 les N boîtes références $\boldsymbol{R_n}$ avec $n \in \{1, ..., N\}$. Il va de soit que ce nombre N de boîtes références représentant le nombre d'objets à détecter va varier à chaque page de l'ensemble d'entraînement.

$$\boldsymbol{R_n} = \{r_{0,n}, r_{1,n}, r_{2,n}, r_{3,n}\} \tag{4.12}$$

Ici, pour tout $i \in 0, ..., 3$, les $r_{i,n}$ sont les valeurs définissant la position des boîtes références. Elles sont homogènes aux valeurs $l_{i,b}$ définissant les positions des boîtes hypothèses (cf. équation 4.1). Ce qui veut dire que pour MultiBox, les $r_{i,n}$ correspondent aux quatre coordonnées de la boîte englobante de l'objet $\boldsymbol{R_n}$ alors que pour YOLO, ils correspondent à la position du centre de l'objet et aux racines carrées des tailles de l'objet.

Le coût de position C_{pos} est similaire pour MultiBox et YOLO et est basé sur une distance Euclidienne :

$$C_{pos} = \frac{1}{2} \sum_{b,n} X_{b,n} ||\boldsymbol{r_n} - \boldsymbol{l_b}||_2^2 = \frac{1}{2} \sum_{b,n} X_{b,n} \sum_{i=0}^{3} (r_{i,n} - l_{i,b})^2 \tag{4.13}$$

Où $\boldsymbol{X}$ est la matrice d'association entre les boîtes références et les boîtes hypothèses. On a $X_{b,n} = 1$ lorsque la boîte hypothèse b est associée à la boîte référence n et $X_{b,n} = 0$ lorsque ce n'est pas le cas. L'obtention de cette matrice d'association $\boldsymbol{X}$ sera détaillée en Section 4.3.

Le coût de confiance C_{conf} diffère selon la méthode. Pour MultiBox, on utilise un coût de confiance logarithmique :

$$C_{conf} = -\sum_{b,n} X_{b,n} log(c_b) - \sum_{b} \left(1 - \sum_{n} X_{b,n}\right) log(1 - c_b) \tag{4.14}$$

Le premier terme de cette équation vise à pénaliser les boîtes hypothèses qui ont été associées à des boîtes références et dont la confiance est faible alors que le deuxième terme vise à pénaliser les boîtes hypothèses qui n'ont pas été associées à des boîtes références alors que leur confiance est élevée.

Cela marche similairement pour YOLO sauf que l'on utilise un coût quadratique à la place du coût logarithmique et on ajoute un paramètre λ pour donner une influence différente au fait d'associer une boîte avec une faible confiance et au fait de ne pas associer une boîte avec une forte confiance :

$$C_{conf} = \sum_{b,n} X_{b,n} (1 - c_b)^2 + \lambda \sum_{b,n} (1 - X_{b,n}) c_b^2 \tag{4.15}$$

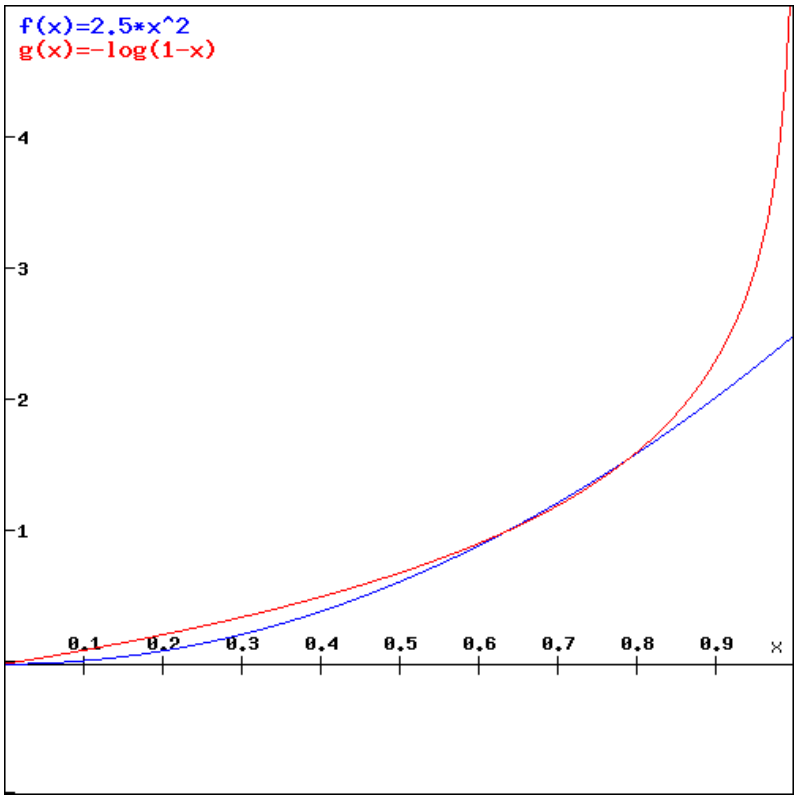


Fig. 4.3 – Comparaison des fonctions x^2 et $-log(1-x)$ sur l'intervalle [0,1].

On remarque une forte similarité entre ces deux coûts de confiance. YOLO utilise une fonction en x^2 (respectivement $1-x^2$) pour être proche de 0 lorsque x est proche de 0 (respectivement de 1) et positif lorsque x se rapproche de 1 (respectivement de 0) la où MultiBox utilise une fonction en $-log(1-x)$ (respectivement $-log(x)$) pour être proche de 0 lorsque x est proche de 0 (respectivement de 1) et positif lorsque x se rapproche de 1 (respectivement de 0). Comme illustré en Figure 4.3, ces fonctions sont assez similaires à un facteur positif près lorsque la prédiction de confiance est proche d'être correcte.

4.3 Appariements entre objets références et objets hypothèses

Si, comme mentionné dans les parties précédentes, des différences mineurs existent entre les modèles YOLO et MultiBox et les manières dont ils sont entraînés, et même si les détails comptent beaucoup en apprentissage automatique [Chatfield et al., 2014], on remarque surtout le nombre important de leurs similarités.

La différence majeure entre ces modèles réside au niveau de l'appariement entre les boîtes

hypothèses et les boîtes références. C'est à dire dans le calcul de la matrice d'association $\boldsymbol{X}$ présente dans les équations du coût de position 4.13 et du coût de confiance 4.14 ou 4.15. On retrouve le fait, mentionné en Section 4.1.2, que YOLO utilise une approche locale pour concevoir les objets et les apparier alors que MultiBox raisonne de manière globale.

4.3.1 Appariement utilisé par YOLO

Pour YOLO [Redmon et al., 2016], l'appariement est effectué uniquement sur des critères de position. Chaque boîte référence est associée à la cellule de l'image dans laquelle se trouve son centre. Seule la première boîte référence qui est associée à une cellule est prise en compte et elle est associée à la boîte hypothèse correspondant à cette cellule qui à la valeur d'IoU (intersection sur union) maximale.

Cette technique a pour avantage d'être très efficace d'un point algorithmique et est très bien adaptée au cas pour lequel elle a été développée, à savoir la détection d'objets naturels, où le nombre d'objets à détecter est très réduit et où, par conséquent, il est très rare que plusieurs objets références soient associés à la même cellule.

Par contre, si le nombre d'objets augmente, le système doit apprendre à détecter seulement l'un des objets présent dans chaque cellule et cet objet change en fonction de l'ordre dans lequel sont présentés les objets références. La Figure 4.4 illustre ces problèmes d'appariement avec une part importante des objets (en bleu) dont le centre se situe dans une cellule (indiquées en pointillés) dans laquelle une autre boîte référence a déjà été placée (en rouge). Cela perturbe fortement les entraînements et empêche certains objets d'être détectés.

4.3.2 Appariement utilisé par MultiBox

L'appariement décrit dans MultiBox [Erhan et al., 2014] permet d'éviter ce problème en utilisant un algorithme d'appariement biparti. Il s'agit d'un problème général et courant où l'on doit déterminer un appariement optimal entre deux ensembles différents de manière à ce que chaque élément d'un ensemble soit affecté à au maximum un élément de l'autre ensemble. Chaque affectation possible d'un élément à un autre a un coût et c'est la somme de ces coûts qui va être minimisée, comme on peut le voir dans la Figure 4.5 qui illustre cette association des tâches aux agents.

L'algorithme Hongrois [Kuhn, 1955] [Munkres, 1957] peut être utilisé pour résoudre ce problème. Il permet de s'assurer que chaque boîte référence est associée à une et une seule boîte hypothèse (sous la contrainte, vérifiée en pratique, qu'il y ait plus de boîtes hypothèses que de

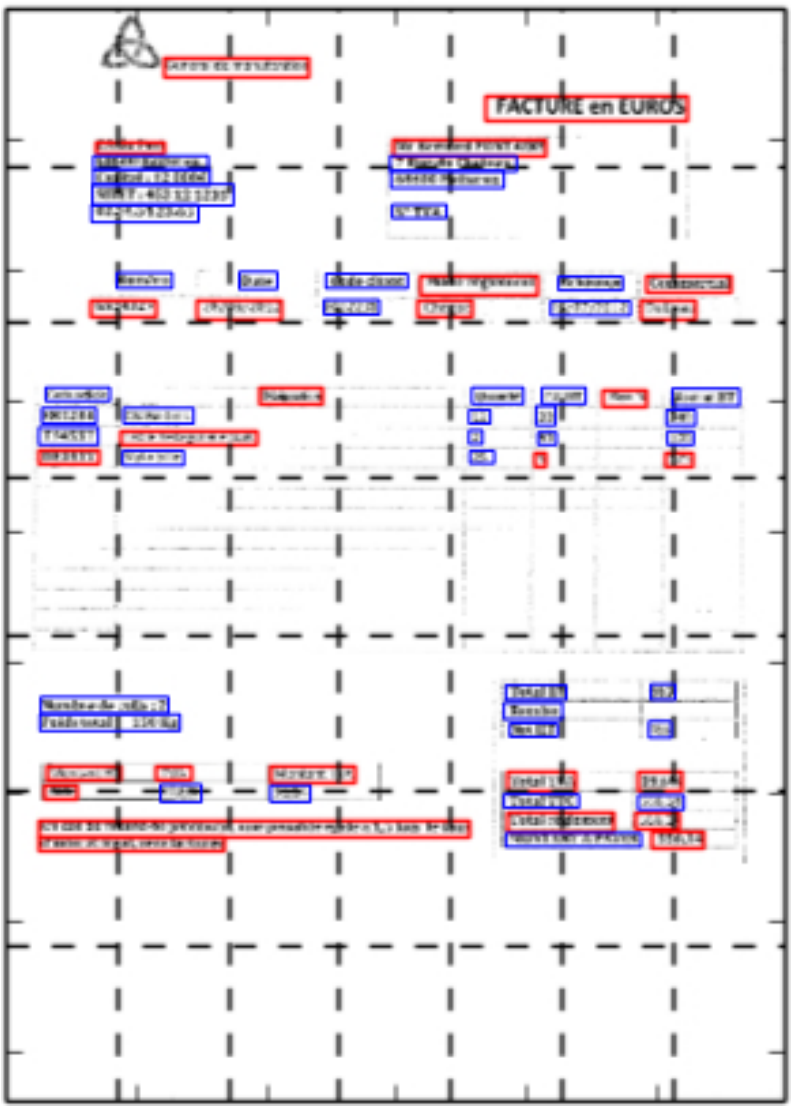


Fig. 4.4 – Problème rencontré avec YOLO. Seuls les objets encadrés en rouge sont associés à des sorties du réseau.

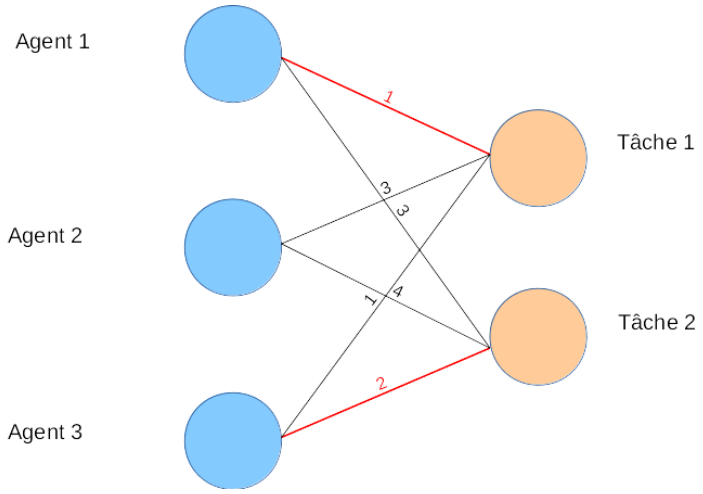


Fig. 4.5 – Exemple de problème d'appariement biparti où les tâches sont associées aux agents en minimisant le coût global.

boîtes références). Il nous assure aussi qu'une boîte hypothèse n'est pas associée à plusieurs boîtes références. En cela, il permet de trouver la matrice binaire $\hat{\boldsymbol{X}}$ qui minimise l'équation 4.11.

$$\hat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X}} C(\boldsymbol{X}) \qquad \text{t.q.} \quad \forall b, \sum_{n=1}^{N} X_{b,n} \in \{0,1\} \quad \text{ET} \quad \forall n, \sum_{b=1}^{B} X_{b,n} = 1 \tag{4.16}$$

$$\hat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X}} \sum_{b,n} X_{b,n} \left[\frac{1}{2} \alpha_{match} \sum_{i=0}^{3} (r_{i,n} - l_{i,b})^2 - log(c_b + log(1 - c_b) \right] - \sum_{b} log(1 - c_b) \tag{4.17}$$

Comme le deuxième terme est constant pour chaque boîte référence (et que l'on suppose $N < B$), cela revient à :

$$\hat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X}} \sum_{b,n} X_{b,n} \left[\frac{1}{2} \alpha_{match} \sum_{i=0}^{3} (r_{i,n} - l_{i,b})^2 - log(c_b + log(1 - c_b) \right] \tag{4.18}$$

On va donc définir pour chaque couple de boîte hypothèse $\boldsymbol{O_b}$ et référence $\boldsymbol{R_n}$, un coût local $L_{b,n}$ qui va servir d'entrée à l'algorithme Hongrois.

$$L_{b,n} = \frac{1}{2} \alpha_{match} \sum_{i=0}^{3} (r_{i,n} - l_{i,b})^2 - log(c_b + log(1 - c_b) \tag{4.19}$$

L'algorithme Hongrois [Kuhn, 1955] [Munkres, 1957] permet de résoudre exactement ce problème d'appariement. Notons néanmoins la principale faiblesse de cet algorithme d'appariement qui réside dans le fait que l'algorithme Hongrois a une complexité maximale en $O(b^3)$

[Munkres, 1957]. Cependant, le nombre d'objets référence étant réduit, les calculs restent relativement rapides puisque la matrice d'appariement est très lacunaire. On note néanmoins que les articles plus récents se basant sur MultiBox [Szegedy et al., 2015b] [Liu et al., 2016] n'utilisent plus cet appariement. C'est probablement à cause d'un nombre plus important d'objets à détecter et surtout de l'augmentation du nombre de sorties du réseau qui induisent une augmentation importante du temps de calcul nécessaire à l'appariement.

Association d'ancres aux objets hypothèses

MultiBox [Erhan et al., 2014] introduit aussi la notion d'ancres vis à vis des objets détectés. L'idée générale est de structurer l'espace des sorties et donc d'associer les sorties du réseau à des boîtes définies a priori et d'associer les boîtes références à ces boîtes a priori plutôt qu'aux boîtes hypothèses du réseau.

Pour cela, un partitionnement en k-moyennes [Steinhaus, 1956], avec un nombre de groupes égal au nombre de boîtes prédites par le réseau (c'est à dire avec $k = B$), est utilisé sur les coordonnées des boîtes références de l'ensemble d'entraînement. Chacun des centroïdes des B groupes obtenus est associé arbitrairement et définitivement à l'une des sorties du réseau. Au moment de l'appariement entre sorties du réseau et boîtes références, on utilisera désormais ces centroïdes à la place des boîtes hypothèses dans les équations 4.17 , 4.18 et 4.19.

Cela n'est utilisé que pour le calcul de la matrice d'appariement $\boldsymbol{X}$. La fonction de coût décrite dans l'équation 4.13 et les gradients associés restent inchangés. On les calcule toujours à partir des boîtes hypothèses (et pas des ancres).

D'après MultiBox [Erhan et al., 2014], utiliser ces ancres au lieu des boîtes hypothèses permet de forcer les prédictions à se diversifier, un nombre plus important de sorties du réseau vont être utilisées. En pratique, cela se traduit par un gain en performance et en temps de convergence.

On note que dans les articles adaptés de MultiBox [Szegedy et al., 2015b] [Liu et al., 2016], les ancres ont été conservées mais définies de manière arbitraire au lieu d'utiliser un partitionnement en k-moyennes.

4.4 Conclusions

Dans cette section, MultiBox [Erhan et al., 2014] et YOLO [Redmon et al., 2016] ont été décrits en détails. Ces méthodes, ou des adaptations de ces méthodes, rencontrent les meilleurs résultats actuels sur les tâches de détection d'objets dans des scènes naturelles.

Nous avons souligné le nombre important de similarités entre ces deux techniques, tant au niveau de leur architecture qu'au niveau de la manière dont les réseaux sont appris.

Nous avons également exposé leur différence majeure qui se situe au niveau de la manière dont les boîtes références sont associées aux sorties du réseau et avons expliqué les limitations de l'approche YOLO lorsque le nombre d'objets présents dans la page est important. Pour cette raison, les techniques que nous avons proposées, et dont les détails sont développés en section 5, sont basées principalement sur l'approche MultiBox.

Chapitre 5

Modèle récurrent et local pour la localisation des lignes de texte.

La tâche de détection des lignes de texte, appliquée à la reconnaissance pleine page de documents hétérogènes, à laquelle nous nous intéressons diffère de la tâche de détection d'objets dans des scènes naturelles sur plusieurs points que nous soulignons en Section 5.1. En particulier, le nombre d'objets à détecter est plus grand et le nombre de données disponibles pour l'entraînement est nettement plus réduit.

Pour cette raison, nous proposons un modèle inspiré mais différent des techniques YOLO [Redmon et al., 2016] et MultiBox [Erhan et al., 2014] que nous avons analysées et dont nous avons décrit le fonctionnement dans le chapitre 4 et qui ont été développées dans le but de détecter des objets dans des scènes naturelles. La Section 5.2 détaille notre utilisation d'un modèle plus petit. La Section 5.4 introduit le fait de prédire les objets de manière locale en partageant les paramètres alors que la Section 5.3 détaille l'utilisation de couches récurrentes pour propager des informations contextuelles.

Dans ce chapitre, nous décrivons un modèle neuronal spécifiquement conçu pour l'analyse d'images de documents en ciblant des situations où la quantité des échantillons est faible. Il s'agit d'un méta-modèle qui servira de base pour un ensemble de modèles décrits dans le chapitre suivant.

TABLE 5.1 – Comparaison des bases PascalVOC [Everingham et al., 2010], ILSRVC [Russakovsky et al., 2015] et Maurdor [Brunessaux et al., 2014]

	Pascal-VOC 2012	ILSRVC 2014	Maurdor
Nombre d'images	17125	544546	3995
Nombre d'objets	40138	615299	135834
Nombre moyen d'objets par page	2.34	1.13	34.00
Nombre maximal d'objets par page	56	17	567
Surface moyenne des objets	0.1529	0.3833	0.0050
Taille moyenne horizontale des objets	0.3406	0.6033	0.2327
Taille moyenne verticale des objets	0.449	0.6353	0.0216
Nombre de classes d'objets	20	200	1

5.1 Définition du problème

Nous commençons par constater que le problème abordé dans le cadre de cette thèse se distingue du cadre classique de la détection d'objets dans la vision par ordinateur. Il y a des différences entre d'un côté les bases utilisées par les méthodes à l'état de l'art dans la tâche de détection d'objets dans des scènes naturelles que sont Pascal-VOC 2012 [Everingham et al., 2010] et ILSRVC 2014 [Russakovsky et al., 2015] et de l'autre, la base Maurdor [Brunessaux et al., 2014] qui est composée d'images de documents hétérogènes et que nous utiliserons dans cette thèse. Le tableau 5.1 nous montre certaines de ces différences.

Une des principales différences entre les deux tâches est que dans le cas de la détection de lignes dans des documents, le nombre d'images disponibles pour l'entraînement est beaucoup plus limité. C'est d'autant plus vrai lorsque l'on se compare à la base ILSRVC puisque les systèmes ayant de bonnes performances sur Pascal-VOC sont souvent pré-entraînés sur ImageNet [Redmon et al., 2016] [Szegedy et al., 2015b]. De plus, le nombre d'objets à détecter est nettement plus élevé ce qui implique des contraintes sur le nombre d'objets maximal que doit pouvoir prédire le système. Et ces objets sont plus petits, surtout dans leur direction verticale, ce qui rend critique la précision de la prédiction des coordonnées des boites.

Par contre, le nombre de classes d'objets est beaucoup plus faible. On peut donc penser que le problème est plus simple. Il n'y a qu'une classe d'objet, des lignes de texte (qui peuvent cependant être manuscrites ou imprimées, et dans des langues différentes) alors que, pour la détection d'objets dans des scènes naturelles, des dizaines ou des centaines de classes différentes sont présentes. De plus, le contexte est plus important dans les images de documents puisque la position des objets les uns par rapport aux autres est plus fortement contrainte que dans les images de scènes naturelles. Si ce contexte est exploité, il pourrait potentiellement permettre de

simplifier la tâche.

Pour prendre en compte ces spécificités, trois changements majeurs ont été effectués par rapport aux modèles classiques de la littérature. Ces contributions seront décrites dans les sections suivantes. En particulier, nous proposons une architecture simplifiée du réseau de neurones (cf. Section 5.2), nous proposons de rendre locale la dernière couche du réseau (cf. Section 5.3), et nous ajoutons des couches récurrentes pour prendre en compte le contexte (cf. Section 5.4).

5.2 Simplification de l'architecture du réseau

Dans cette section nous détaillons l'architecture des réseaux de neurones utilisés. Cette architecture a été conçue afin de tenir compte des spécificités de la tâche sur laquelle nous travaillons.

Notons d'abord, que nous avons choisi d'utiliser comme entrée de nos réseaux des images en niveaux de gris. L'usage d'images couleur n'a pas, expérimentalement, apporté d'amélioration. Dans le but de pouvoir détecter des objets plus petits, nous avons utilisé comme taille d'entrée pour nos images 598×838. Les images sont redimensionnées de manière isotropique, en gardant le ratio des dimensions horizontales et verticales, afin de garder l'apparence des lignes. Un remplissage blanc est ajouté autour de l'image si nécessaire.

Le premier choix de conception majeur que nous avons effectué a été d'opter pour une architecture simplifiée du réseau de neurones. Comme la quantité d'images disponibles pour l'entraînement est nettement plus petite que pour la tâche de détection d'images dans des scènes naturelles, nous avons choisi de fortement réduire la taille de nos réseaux. Nous sommes passés à un réseau ne comportant que 5 couches de convolution (comme pour MultiBox) contre 24 dans le réseau utilisé dans YOLO. Surtout, nous avons fortement limité le nombre de neurones présents sur chaque couche. Si dans AlexNet (et donc dans MultiBox), le nombre de neurones sur les couches convolutionnelles passait de 96 à 384 à travers le réseau, nous avons choisi, comme illustré dans le Tableau 5.2 qui indique les paramètres utilisés pour chaque couche, un nombre de neurones sur les couches convolutionnelles nettement plus petit allant de 12 à 36. Cela permet de passer d'un nombre de paramètres libres au niveau des couches convolutionnelles d'environ 2.3 millions dans MultiBox à un peu moins de 25 000 avec notre réseau modifié. Ce qui est rendu possible en raison du fait que les objets soient moins variés et qu'une seule catégorie d'objets soit présente dans notre jeu de données.

Le deuxième choix de conception que nous avons réalisé au niveau de l'architecture de notre réseau se situe au niveau de la taille des filtres convolutionnels. Comme on le voit dans le Tableau 5.1, les objets à détecter dans la tâche Maurdor sont beaucoup plus larges que hauts, d'environ un facteur 10, alors qu'ils sont environ aussi hauts que larges pour les tâches de détection Pascal-

TABLE 5.2 – Taille des filtres, nombre de neurones et nombre de paramètres sur les 5 couches convolutionnelles de notre réseau. Comparaison avec le réseau AlexNet/MultiBox.

	Notre modèle				AlexNet/MultiBox			
Couche	Taille des filtres	*Stride*	Nombre de neurones	Nombre de paramètres	Taille des filtres	*Stride*	Nombre de neurones	Nombre de paramètres
C1	4×4	3×3	12	204	11×11	4×4	96	34944
C2	4×3	3×2	16	2320	5×5	1×1	256	307456
C3	6×3	4×2	24	6936	3×3	1×1	384	885120
C4	4×3	3×2	30	8670	3×3	1×1	384	663936
C5	3×2	2×1	36	6516	3×3	1×1	256	442624

Voc et ILSRVC. On remarque également que les objets sont plus petits. Pour cette raison, là où MultiBox et YOLO utilisent uniquement des filtres convolutionnels et des *pooling* de forme carrés, nous avons utilisé des filtres rectangulaires. Cela permet, comme illustré Figure 5.1 et détaillé en Figure 5.2, d'avoir des champs réceptifs horizontaux qui correspondent mieux à nos objets lignes ou à leurs différents composants (mots, lettres).

Enfin, pour réduire la dimensionnalité de nos cartes de caractéristiques, nous avons préféré utiliser des *strides* plus grands dans nos couches convolutionnelles plutôt que d'utiliser du *maxpooling*. En effet, le *maxpooling* est motivé par le fait de pouvoir identifier des caractéristiques locales de manière invariante par rapport à la position. Dans une tâche de localisation, cette invariance n'est pas souhaitable.

Les choix mentionnés ci-dessus ont été justifiés expérimentalement, le détail de ces expériences sera donné en Section 7.4.

5.3 Convolution 1x1 – Couche complètement convolutionnelle.

Si réduire le nombre de neurones dans les couches convolutionnelles a permis de baisser le nombre de paramètres libres de notre réseau, on remarque que la majeure partie de ces paramètres sont présents sur les couches complètement connectées. C'est le cas de 58 millions sur 61 pour le réseau Alexnet-MultiBox. Ce nombre important de paramètres à ce niveau du réseau est causé, d'une part, par le fait qu'il soit nécessaire d'avoir, comme entrée, des caractéristiques permettant de retranscrire l'image entière et, notamment, la position et la nature de tous les objets s'y trouvant. On utilise donc toutes les caractéristiques obtenues en sortie de la dernière couche

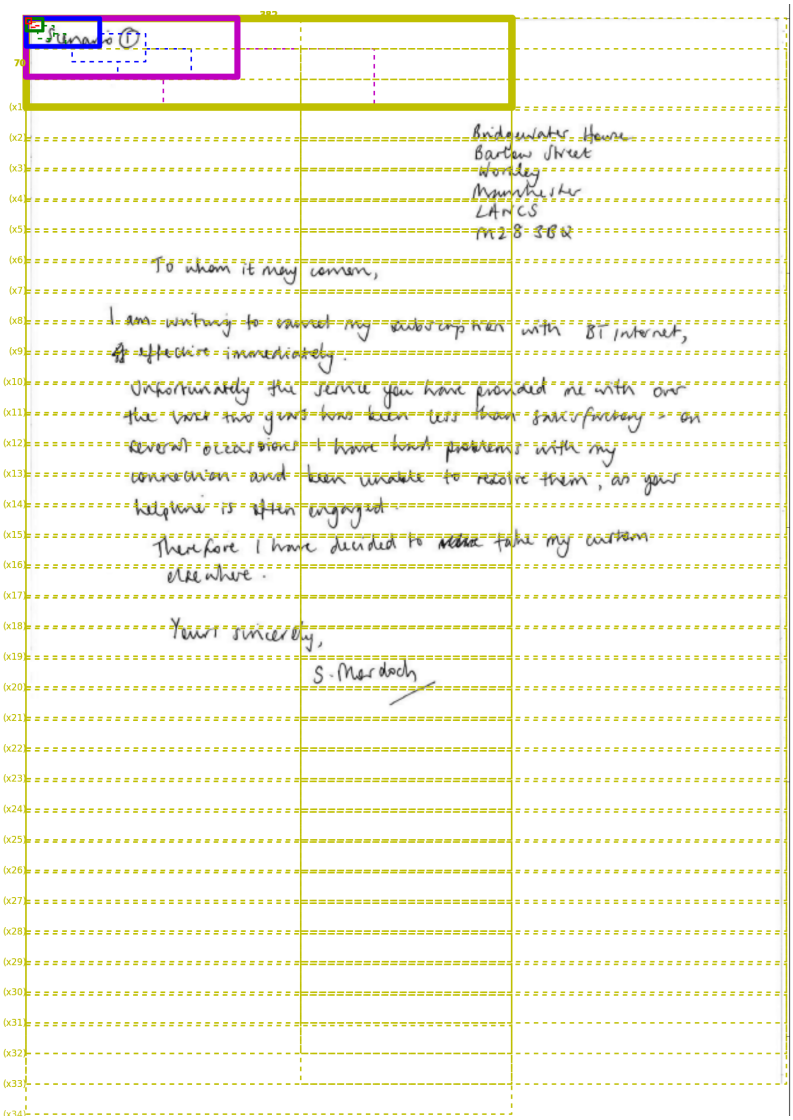


Fig. 5.1 – Illustration des champs réceptifs relatifs aux différentes couches de notre réseau. Les différentes positions des champs réceptifs, séparées par les *strides*, sont illustrées en pointillés.

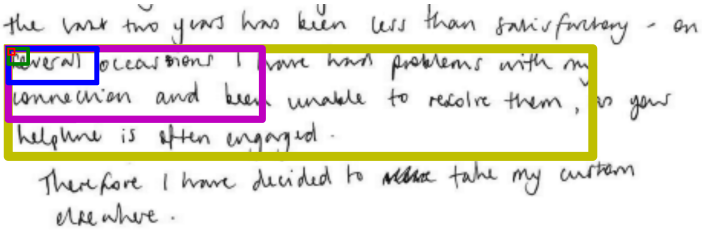


Fig. 5.2 – Détail de champs réceptifs correspondant aux neurones des différentes couches convolutionnelles (1ère : rouge, 2ème : vert, 3ème : bleu, 4ème : magenta et 5ème jaune). On observe que certains objets textuels sont plus larges que les champs réceptifs convolutionnels de nos neurones de sortie.

convolutionnelle. Ce nombre important de paramètres est également causé par le fait qu'on ait comme sortie de cette (ou de ces) couche complètement connectée l'ensemble des objets hypothèses pour cette image. Ce nombre d'objets hypothèses doit être supérieur au nombre maximal d'objets pouvant être présents dans la même image. MultiBox [Erhan et al., 2014] utilise 100 ou 200 objets hypothèses pour des tâches avec un nombre maximal d'objets à détecter de 56 (cf. Tableau 5.1). Notre tâche a la particularité d'avoir un nombre moyen d'objets par page beaucoup plus élevé ce qui se traduit par un nombre maximal d'objets pouvant être présents dans une même page de 567. Augmenter le nombre de sorties du réseau en conséquence entraîne une augmentation mécanique du nombre de paramètres libres sur la dernière couche complètement connectée ; ce qui est en contradiction avec notre contrainte de travailler avec de petits jeux de données lors de l'apprentissage.

Dans le but de résoudre ce problème, l'une de nos principales contributions a été de rendre les prédictions de notre réseau locales et de remplacer cette (ou ces) couche complètement connectée par une couche localement connectée. Seules les caractéristiques relatives à une localisation de l'image, dont les champs réceptifs sont visibles en jaune, en Figure 5.1, vont être utilisées pour prédire la présence d'objets.

Notons que cette couche localement connectée est, au final, assimilable à une nouvelle couche convolutionnelle ayant pour taille de filtres 1×1.

La prédiction devient locale, et pour chaque position $i \in \{0, ..., I\}$, $j \in \{0, ..., J\}$ de notre carte de caractéristiques en sortie de la dernière couche convolutionnelle, on va prédire K objets (définis chacun par $P = 5$ valeurs). Donc le nombre total B d'objets hypothèses prédits par le système pour chaque page est égal à $I \times J \times K$. I, J et K sont définis par la taille de l'image d'entrée (598×838) et par les tailles des filtres et *strides* du réseau détaillés dans le tableau 5.2. Dans la pratique, $I = 2$, $J = 33$ et $K = 20$ donc le réseau prédira $B = 1320$ objets hypothèses par image ce qui est nettement supérieur au nombre d'objets pouvant être présents dans une page.

Une conséquence majeure de ce choix est que les objets peuvent être plus grands que les champs réceptifs associés à nos sorties locales, comme illustré en Figure 5.2. Cela rend nécessaires quelques modifications par rapport aux prédictions directes des coordonnées décrites dans le chapitre précédent et utilisées par MultiBox.

On va prédire les coordonnées du coin en bas à gauche de nos objets dans le référentiel des champs réceptifs. La largeur et la hauteur de nos objets seront quant à eux prédits en étant normalisés par rapport à la largeur de la page.

En se rappelant des B objets $\boldsymbol{O_b}$ prédits par MultiBox pour une image $\boldsymbol{Im}$ et définis dans l'équation 4.1 :

$$\boldsymbol{O_b} = \{l_{0,b}, l_{1,b}, l_{2,b}, l_{3,b}, c_b\} \tag{5.1}$$

$$\{\boldsymbol{O_1}, ..., \boldsymbol{O_B}\} = \mathcal{F}(\boldsymbol{Im}) \tag{5.2}$$

Et en définissant W la largeur et H la hauteur de la page, et w et h les largeur et hauteur des champs réceptifs finaux, on va prédire les objets $\boldsymbol{O''_{i,j,k}}$ correspondants avec i et j pour la position dans la carte de caractéristiques et $k \in \{0, ..., K\}$ représentant l'indice de l'objet à cette position.

$$\boldsymbol{O''_{i,j,k}} = \{l''_{0,i,j,k}, l''_{1,i,j,k}, l''_{2,i,j,k}, l''_{3,i,j,k}, c_{i,j,ks}\} \tag{5.3}$$

Avec :

$$l''_{0,i,j,k} = (l_{0,i,j,k} - \delta_x(i)) \times W/w \tag{5.4}$$

$$l''_{1,i,j,k} = (l_{2,i,j,k} - \delta_y(j)) \times H/h \tag{5.5}$$

$$l''_{2,i,j,k} = l_{1,i,j,k} - l_{0,i,j,k} \tag{5.6}$$

$$l''_{3,i,j,k} = l_{3,i,j,k} - l_{2,i,j,k} \tag{5.7}$$

Où les $\delta_x(i))$ et $\delta_y(j)$ correspondent à la position du champs réceptif correspondant aux sorties i,j. Soit S_x et Sy les décalages verticaux et horizontaux entre deux champs réceptifs successifs (eux mêmes définis par les *strides* des couches convolutionnelles).

On a :

$$\delta_x(i) = i \times S_x/W \tag{5.8}$$

$$\delta_y(j) = j \times S_y/H \tag{5.9}$$

Comme pour YOLO [Redmon et al., 2016] dans l'équation 4.10, on a aussi la particularité de pouvoir revenir aux prédictions directes des coordonnées des objets avec une transformation linéaire $\mathcal{G}'$. La différence étant que les prédictions sont désormais locales.

$$\mathcal{F}''(\boldsymbol{Im}) = \mathcal{G}'(\mathcal{F}(\boldsymbol{Im})) \tag{5.10}$$

La fonction de coût C, quant à elle, reste similaire à celle décrite pour MultiBox en Section 4.2 et dans les équations 4.11, 4.13 et 4.14 :

$$C = \alpha \left(\frac{1}{2} \sum_{b,n} X_{b,n} ||r_n - l_b||_2^2 \right) - \sum_{b,n} X_{b,n} log(c_b) - \sum_b \left(1 - \sum_n X_{b,n} \right) log(1 - c_b) \tag{5.11}$$

De même pour l'appariement où, comme pour MultiBox en Section 4.3.2, on associe les boites références aux boites hypothèses de manière globale.

L'algorithme Hongrois [Munkres, 1957] est utilisé pour minimiser la fonction de coût vis à vis de la matrice d'appariement $\boldsymbol{X}$:

$$\boldsymbol{X} = \arg\min_x \sum_{b,n} X_{b,n} \left[\frac{1}{2} \alpha_{match} \sum_{i=0}^{3} (r_{i,n} - l_{i,b})^2 - log(c_b + log(1 - c_b) \right] \tag{5.12}$$

En s'assurant que (comme le nombre d'objets références N est plus petit que le nombre d'objets hypothèses B) :

$$\forall n, \sum_b X_{b,n} = 1 \quad \text{ET} \quad \forall b, \sum_n X_{b,n} \leq 1 \tag{5.13}$$

Contrairement à ce qui est présenté dans MultiBox [Erhan et al., 2014], nous n'avons expérimentalement pas obtenu d'amélioration de la vitesse de convergence ou des performances en utilisant les ancres détaillées en Section 4.3.2. Nous avons donc choisi d'effectuer cet association entre objets hypothèses et objets références en utilisant directement la position des objets hypothèses dans l'équation 5.12.

Par contre, nous avons observé qu'utiliser un paramètre α_{match}, pondérant le coût des erreurs de position vis à vis des erreurs de confiance, plus grand lors de l'association des boites références aux boites hypothèses que le paramètre α utilisé dans notre fonction de coût, et donc dans notre optimisation à l'aide de rétro-propagation des gradients, permettait de faciliter l'entraînement et d'obtenir de meilleures performances. Cela est dû au fait que cela aide, en particulier au début de l'entraînement, notre système à associer des boites hypothèses plus diverses. Utiliser un α_{match} plus grand a donc un résultat similaire à celui d'utiliser des ancres dans MultiBox [Erhan et al., 2014].

Dernier changement au niveau de l'optimisation, nous avons utilisé RmsProp [Tieleman and Hinton, 2012] et une taille de *minibatches* de 8. Avoir des *minibatches* plus petits (que les 64 utilisés par MultiBox) est en accord avec la taille de notre base de données plus restreinte.

5.4 Prise en compte du contexte avec 2D-LSTMs

Dans la section précédente, nous avons introduit une approche visant à prédire les objets de manière locale, ce qui permet de réduire fortement le nombre de paramètres présents dans le réseau et donc de faciliter l'entraînement en présence d'un nombre réduit d'images dans l'ensemble d'entraînement.

Cette manière d'utiliser des filtres dont les champs réceptifs sont locaux, comme illustré Figure 5.1, signifie une perte d'information. En particulier, cela signifie que la prise en compte du contexte n'est plus possible. Autre problème, certains objets particulièrement grands (ou longs) peuvent sortir du cadre de ces champs réceptifs. Dit autrement, et comme on peut le voir dans la Figure 5.2 qui illustre ces champs réceptifs, il est alors parfois demandé au réseau de prédire la taille de l'objet sans voir l'intégralité de cet objet, ce qui n'est pas possible précisément. De plus, cela signifie une perte du contexte lié au reste du document. La connaissance de la mise en page globale du document peut, par exemple, permettre d'apprendre à aligner les lignes de texte appartenant à un même paragraphe ou à séparer les colonnes d'un tableau de manière plus efficace.

Dans l'optique de profiter à la fois des avantages de la détection locale détaillée dans la partie 5.3 et de récupérer des informations liées au contexte, nous nous sommes inspirés des techniques de reconnaissance de l'écriture [Graves and Schmidhuber, 2009] et avons ajouté des couches récurrentes entre nos couches convolutionnelles.

Les problèmes d'évanouissement des gradients [Hochreiter, 1998] sont particulièrement présents dans les réseaux récurrents. En effet, les gradients sont propagés à la fois sur le nombre de couches convolutionnelles mais aussi tout au long des dimensions horizontales et verticales des cartes de caractéristiques sur lesquelles les couches récurrentes sont appliquées.

Afin de résoudre ces problèmes nous avons utilisé des cellules LSTMs (pour Long Short-Term Memory) [Hochreiter, 1991] [Hochreiter and Schmidhuber, 1997] qui sont une adaptation des modèles récurrents avec des portes. Ces cellules LSTMs sont décrites en détails dans la sous-section 5.4.1.

En particulier, nous utilisons des LSTMs en deux dimensions (2D-LSTMs) introduites par Graves et al. [Graves and Schmidhuber, 2009] pour la reconnaissance de l'écriture manuscrite. Ces travaux ont pour particularité de s'attaquer à un problème en deux dimensions, une image, au lieu des traditionnels problèmes avec une seule dimension, temporelle, comme la traduction ou la reconnaissance de la parole. Cela signifie que pour chaque position dans la carte de caractéristiques qui entre dans la couche 2D-LSTM, on aura comme entrée de la LSTM, en plus des caractéristiques de la couche sous-jacente, les sorties de cette même couche LSTM aux positions précédentes horizontales et verticales.

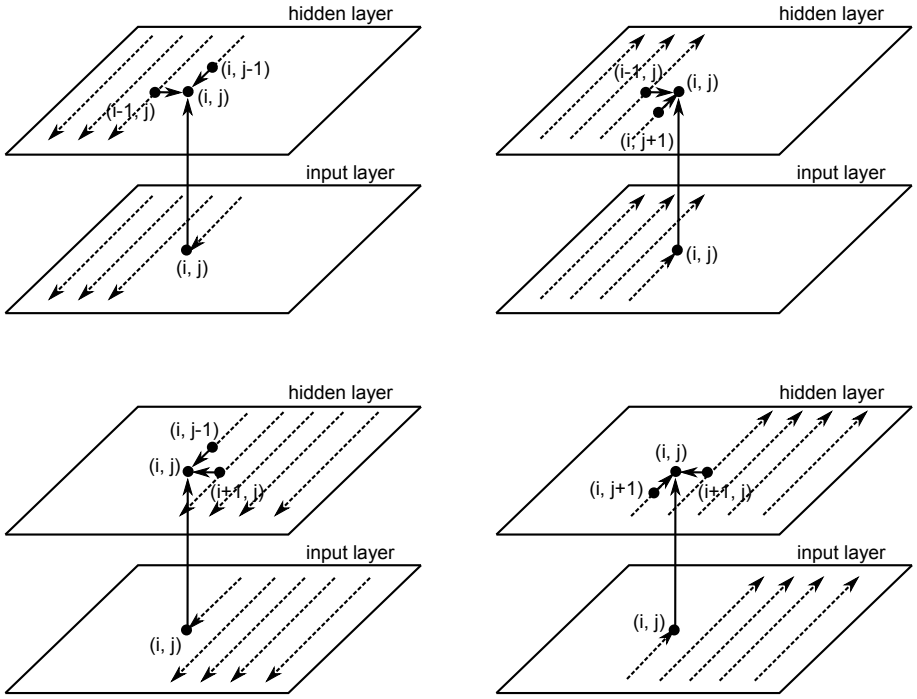


Fig. 5.3 – Illustration du parcours des images dans les quatre directions par les couches récurrentes.

Similairement à ce qui est fait dans certains problèmes mono-dimensionnels [Schuster and Paliwal, 1997], cela est fait séparément pour les quatre directions possibles (Haut-Gauche, Haut-Droite, Bas-Gauche et Bas-Droite) comme illustré dans la Figure 5.3 et les sorties sont combinées.

Les 2D-LSTMs ont été utilisées avec succès pour la reconnaissance de l'écriture [Graves and Schmidhuber, 2009] [Moysset et al., 2014a] mais aussi, concomitamment à ces travaux, pour des tâches aussi variées que la binarisation de documents [Afzal et al., 2015], la segmentation sémantique d'images [Stollenga et al., 2015] [Byeon et al., 2015], la génération de description d'images [Li et al., 2016] ou la génération d'images [Theis and Bethge, 2015] [Oord et al., 2016].

Dans nos travaux, nous proposons d'utiliser les 2D-LSTMs pour localiser les objets. Leur fonctionnement est décrit en détails dans la partie suivante.

5.4.1 Description du fonctionnement des 2D-LSTMs

La LSTM en deux dimensions est illustrée par le schéma présenté dans la Figure 5.4. Comme mentionné précédemment, et illustré en Figure 5.3, les cartes de caractéristiques sont parcourues selon leurs quatre directions diagonales (Haut-Gauche, Haut-Droite, Bas-Gauche et Bas-Droite). Il y a donc quatre cellules 2D-LSTMs qui fonctionnent en parallèle.

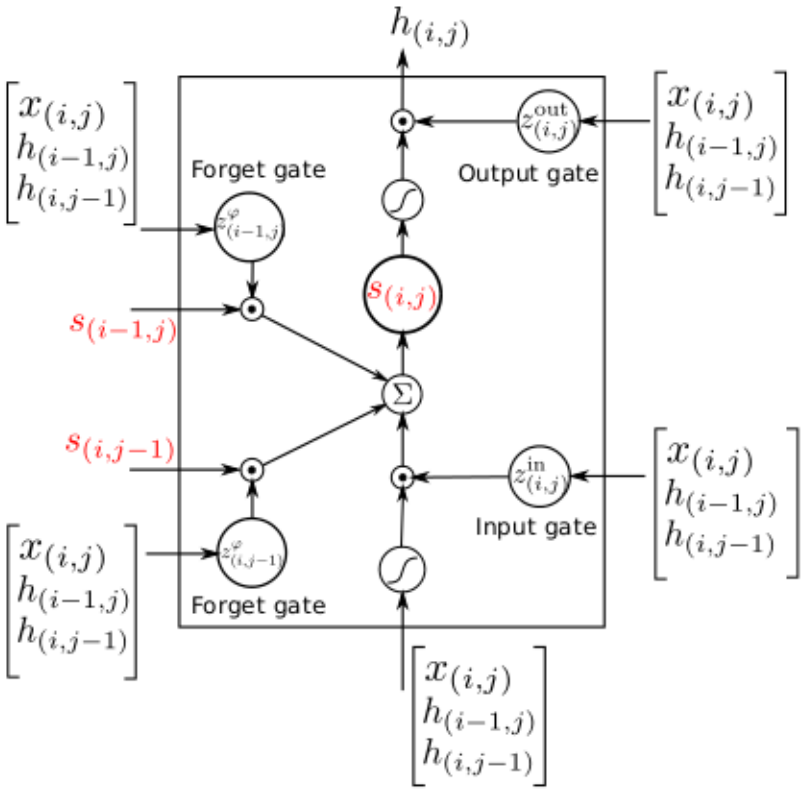


Fig. 5.4 – Illustration d'une cellule 2D-LSTM

La cellule LSTM multi-dimensionnelle prend comme entrées et sorties les valeurs suivantes :

— $\boldsymbol{x_{(i,j)}}$ le vecteur d'entrée à la position i, j, c'est à dire la sortie de la couche précédente à la position i, j.
— $\boldsymbol{h_{(i,j)}}$ la sortie de la cellule à la position i, j.
— $\boldsymbol{s_{(i,j)}}$ l'état interne de la cellule.

On définit aussi les paramètres du réseau qui vont être appris au cours de l'entraînement :

— $\boldsymbol{W_c}$, $\boldsymbol{Ux_c}$, $\boldsymbol{Uy_c}$ et $\boldsymbol{b_c}$ les paramètres du réseau pour le calcul de l'état interne. Il s'agit, respectivement, des paramètres relatifs au vecteur d'entrée et aux sorties des positions précédentes horizontales et verticales ainsi que du biais.
— $\boldsymbol{W_{in}}$, $\boldsymbol{Ux_{in}}$, $\boldsymbol{Uy_{in}}$ et $\boldsymbol{b_{in}}$ les paramètres du réseau pour la porte d'entrée.
— $\boldsymbol{W_{fx}}$, $\boldsymbol{Ux_{fx}}$, $\boldsymbol{Uy_{fx}}$ et $\boldsymbol{b_{fx}}$ les paramètres du réseau pour la porte d'oubli horizontale.
— $\boldsymbol{W_{fy}}$, $\boldsymbol{Ux_{fx}}$, $\boldsymbol{Uy_{fy}}$ et $\boldsymbol{b_{fy}}$ les paramètres du réseau pour la porte d'oubli verticale.
— $\boldsymbol{W_o}$, $\boldsymbol{Ux_o}$, $\boldsymbol{Uy_o}$ et $\boldsymbol{b_o}$ les paramètres du réseau pour la porte de sortie.

Deux types de non-linéarités sont présentes dans la cellule LSTM :

— σ la fonction sigmoïde $\sigma(x) = 1/(1 + e^{-x})$.
— $tanh$ la fonction tangente hyperbolique

Enfin, pour les besoins de l'explication, nous définirons les valeurs internes suivantes :

— $\boldsymbol{c}$ la sortie de la fonction d'entrée.
— $\boldsymbol{in}$ le vecteur relatif à la porte d'entrée.
— $\boldsymbol{fx}$ le vecteur relatif à la porte d'oubli horizontale.
— $\boldsymbol{fy}$ le vecteur relatif à la porte d'oubli verticale.
— $\boldsymbol{o}$ le vecteur relatif à la porte de sortie.

Le produit d'Hadamard (produit terme à terme) sera noté $\circ$.

Les calculs effectués dans la cellule LSTM, et illustrés graphiquement dans la figure 5.4 peuvent

se formaliser de la façon suivante :

$$\boldsymbol{c} = tanh(\boldsymbol{W_c x_{(i,j)}} + \boldsymbol{U x_c h_{(i-1,j)}} + \boldsymbol{U y_c h_{(i,j-1)}} + \boldsymbol{b_c}) \tag{5.14}$$

$$\boldsymbol{in} = \sigma(\boldsymbol{W_{in} x_{(i,j)}} + \boldsymbol{U x_{in} h_{(i-1,j)}} + \boldsymbol{U y_{in} h_{(i,j-1)}} + \boldsymbol{b_{in}}) \tag{5.15}$$

$$\boldsymbol{fx} = \sigma(\boldsymbol{W_{fx} x_{(i,j)}} + \boldsymbol{U x_{fx} h_{(i-1,j)}} + \boldsymbol{U y_{fx} h_{(i,j-1)}} + \boldsymbol{b_{fx}}) \tag{5.16}$$

$$\boldsymbol{fy} = \sigma(\boldsymbol{W_{fy} x_{(i,j)}} + \boldsymbol{U x_{fy} h_{(i-1,j)}} + \boldsymbol{U y_{fy} h_{(i,j-1)}} + \boldsymbol{b_{fy}}) \tag{5.17}$$

$$\boldsymbol{o} = \sigma(\boldsymbol{W_o x_{(i,j)}} + \boldsymbol{U x_o h_{(i-1,j)}} + \boldsymbol{U y_o h_{(i,j-1)}} + \boldsymbol{b_o}) \tag{5.18}$$

$$\tag{5.19}$$

$$\boldsymbol{s_{(i,j)}} = \boldsymbol{s_{(i-1,j)}} \circ \boldsymbol{fx} + \boldsymbol{s_{(i,j-1)}} \circ \boldsymbol{fy} + \boldsymbol{c} \circ \boldsymbol{in} \tag{5.20}$$

$$\boldsymbol{h_{(i,j)}} = tanh(\boldsymbol{s_{(i-1,j)}}) \circ \boldsymbol{o} \tag{5.21}$$

$$\tag{5.22}$$

5.4.2 Analyse de la LSTM

Par rapport aux modèles récurrents simples, avoir un état interne permet de garantir que le flux d'erreur reste constant à travers l'espace des cartes de caractéristiques sur lesquelles sont appliquées nos récurrences. Le fait que ce flux soit constant permet d'éviter à la fois que le gradient n'explose, et donc que l'entraînement devienne instable, mais aussi qu'il ne s'évanouisse et donc qu'il soit incapable d'apprendre des relations de long terme [Hochreiter, 1991].

Les portes d'entrée, multiplicatives [Hochreiter and Schmidhuber, 1997], servent à éviter que la mémoire stockée dans les états internes ne soit perturbée par des entrées non pertinentes. Similairement, la porte de sortie permet d'éviter que la mémoire stockée n'impacte les couches suivantes lorsqu'elle n'est pas pertinente pour elles. Dit autrement, le réseau va apprendre à décider, respectivement grâce à ses portes d'entrée et de sortie, quand il est utile de modifier l'information en mémoire et quand il est utile d'y accéder. Enfin, les portes d'oubli, introduites par Gers et al. [Gers et al., 1999] vont permettre d'apprendre à reconnaître le moment où il est utile de remettre à zéro la mémoire stockée dans l'état interne. Cette capacité à oublier des informations désormais non pertinentes a été prouvée très importante sur un ensemble varié de tâches [Greff et al., 2016].

Nous avons ajouté quatre couches de LSTMs spatiales, multi-directionnelles, entre nos couches convolutionnelles. Le détail de leurs positions, ainsi que les nombres de paramètres libres associés sont donnés dans le tableau 5.3. Les quatre couches récurrentes représentent un total d'environ 115 000 paramètres libres mais permettent de prendre en compte le contexte en utilisant une couche finale convolutionnelle au lieu d'une couche complètement connectée. On notera qu'utiliser une couche complètement connectée à cet endroit aurait nécessité 237 000 paramètres libres.

TABLE 5.3 – Taille des filtres, nombre de neurones et nombre de paramètres sur l'ensemble des couches de notre réseau.

Couche	Taille des filtres	*Stride*	Nombre de neurones	Taille de la carte de caractéristiques	Nombre de paramètres
Input	/	/	1	598×838	
C1	4×4	3×3	12	199×279	204
LSTM 1	/	/	" "	199×279	8880
C2	4×3	3×2	16	66×139	2320
LSTM 2	/	/	" "	66×139	15680
C3	6×3	4×2	24	16×69	6936
LSTM 3	/	/	" "	16×69	35040
C4	4×3	3×2	30	5×34	8670
LSTM 4	/	/	" "	5×34	54600
C5	3×2	2×1	36	2×33	6516
Output	1×1	1×1	100	2×33	3700

Le nombre total de paramètres libres de notre réseau est d'environ 142 000, bien en deçà des 61 millions utilisés par MultiBox ou des 271 millions utilisés par YOLO.

5.5 Conclusions

Nous avons conçu un système adapté à une tâche dans laquelle beaucoup d'objets sont présents, et pour laquelle un nombre restreint d'images annotées sont disponibles. En particulier, trois contributions majeurs ont été apportées.

Nous avons choisi d'utiliser un nombre réduit de neurones sur les couches cachées de notre réseau et avons adapté la taille des filtres convolutionnels à la forme de nos objets.

Nous avons rendu la dernière couche locale en choisissant de mettre, au lieu d'une couche complètement connectée, une couche convolutionnelle avec des filtres de taille 1×1 ce qui a permis de réduire drastiquement le nombre de paramètres et de partager l'apprentissage entre les différentes positions de l'image.

Nous avons, pour récupérer des informations de contexte perdues à cause du caractère local des prédictions, intercalé des couches de 2D-LSTMs entre nos couches convolutionnelles.

Ces trois apports ont permis de rendre notre système particulièrement adapté à la tâche de

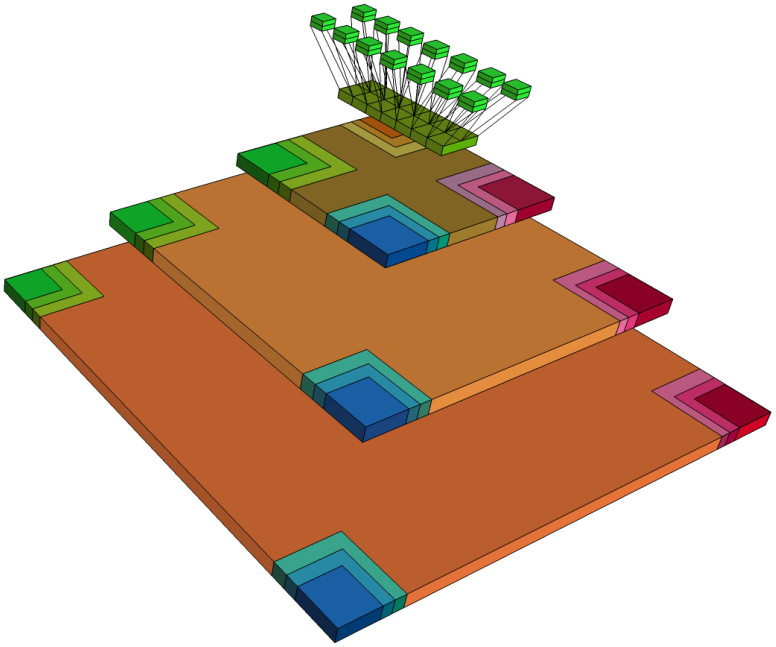


Fig. 5.5 – Schéma de notre réseau mêlant couches convolutionnelles et couches récurrentes qui prédit localement la position des objets. En comparaison de MultiBox, illustré en Figure 4.2, des couches récurrentes multi-directionnelles sont ajoutées entre les couches convolutionnelles et les prédictions sont faites de manière locale pour chaque position. Toutes les couches et les sorties ne sont pas représentées.

localisation des lignes dans des documents hétérogènes et à ses contraintes. Ils seront expérimentalement justifiés dans le chapitre 7.

Chapitre 6

Stratégies de reconnaissance pleine page

Dans le chapitre 5, nous avons décrit un méta-modèle adapté à l'analyse de documents. Ce méta-modèle est, dans ce chapitre, décliné en trois variantes correspondant à des stratégies de prédiction de la position des objets différentes.

La détection de lignes dans des images de documents s'inscrit dans l'optique de reconnaître le texte présent dans celles-ci. Comme nous l'illustrons en Section 6.1, cette reconnaissance de texte est sensible à la finesse des prédictions effectuées. Dans le but de prendre en compte ce besoin de précision, nous proposons en Section 6.2 trois alternatives de détection et de reconnaissance pleine page. La première est de prédire directement les boîtes englobant les lignes et est détaillée en Section 6.2.1). La deuxième, détaillée en Section 6.2.2 est de prédire séparément les coins inférieurs gauches et les coins supérieurs droits de ces boîtes englobantes et de les apparier à l'aide d'un classifieur. Et la troisième, en Section 6.2.3, est de prédire uniquement les côtés gauches des boîtes englobantes et de charger le reconnaisseur d'identifier la position de la fin de la ligne.

Les performances de ces systèmes seront montrées et analysées dans le chapitre suivant, en Section 7.3.

6.1 Importance de la précision des prédictions

Comme illustré précédemment dans le tableau 5.1, les objets que nous souhaitons obtenir dans notre tâche de détection de lignes sont plus petits que ceux des tâches de détection d'objets dans des scènes naturelles. Ceci est particulièrement vrai dans la direction verticale. La Section 6.1.1 montre l'impact des erreurs de localisation des lignes sur le taux de reconnaissance du texte compris dans ces lignes alors que la Section 6.1.2 étudie l'impact du nombre de coordonnées détectées par le réseau sur la précision de ces localisations.

6.1.1 Impact de la précision sur le taux de reconnaissance

Dans cette section, nous travaillons sur les lignes en anglais de la base Maurdor, à la fois manuscrites et imprimées. Nous visons à évaluer l'impact de la précision des positions des lignes sur la reconnaissance du texte. Pour cela, les positions références des lignes de texte sont utilisées et un décalage artificiel leur est appliqué pour simuler différents types d'erreurs de localisation. Le contenu de ces lignes est ensuite reconnu a l'aide d'un système de reconnaissance à base de 2D-LSTMs similaire à celui présenté par Pham et al. [Pham et al., 2014] et décrit précédemment en section 3.2. Ce système est entraîné à reconnaître à la fois les textes manuscrits et imprimés, de la manière décrite et expérimentalement justifiée dans Moysset et al. [Moysset et al., 2014b]. Les taux d'erreurs au niveau mot et caractère, qui seront détaillés en section 7.2.2, sont calculés et nous servent de métriques.

La première expérience étudie l'impact du décalage de la boîte par rapport à sa position référence. Cela signifie que la hauteur et la largeur de la boîte restent inchangées mais que la position du coin inférieur gauche de cette boîte est décalé par rapport à sa position initiale. Les résultats sont montrés dans la Figure 6.1. On observe que les erreurs au niveau de la position verticale sont plus graves que celles sur la position horizontale. Cela s'explique en raison du fait que nos objets sont plus larges que hauts et qu'une erreur peut facilement faire sortir l'objet de la boîte prédite, mais aussi par le fait que nos mots et caractères sont ordonnés de manière horizontale. Une erreur horizontale ne causera la perte que de quelques lettres alors qu'une erreur verticale impliquera la perte du bas ou du haut de toutes les lettres de la ligne, rendant celles-ci difficilement reconnaissables. On observe qu'une erreur verticale d'une dizaine de pixels dans des images ayant une résolution de 200 dpi (et donc ayant typiquement une taille en pixels de 1654 x 2339 en format A4) cause une réduction très forte des performances avec un taux d'erreur mot de plus de 80% contre 15% sans décalage. On notera que le taux d'erreur mot peut être supérieur à 100% à cause des insertions.

La deuxième expérience vise à étudier l'impact des erreurs sur les prédictions des hauteurs et des largeurs des boîtes. Pour les mêmes images, toujours en 200 dpi, un nombre donné de pixels est ajouté ou retiré aux largeurs ou aux hauteurs des boîtes références. Ces pixels peuvent être ajoutés (respectivement retirés) d'un côté ou de l'autre de la boîte ou bien partagés entre les deux. Les résultats sont donnés en Figure 6.2. On remarque, comme dans l'expérience précédente, que l'on est plus sensible aux erreurs verticales qu'aux erreurs horizontales. On observe également que l'on est beaucoup plus sensible au fait de réduire la taille de la boîte qu'au fait de l'agrandir. Cela s'explique par le fait que réduire la taille de la boîte implique de rendre non reconnaissables certains caractères alors que augmenter la taille de celle-ci laisse l'information mais ajoute un bruit sous la forme des objets voisins ; bruit qui est acceptable dans une certaine mesure.

Ces deux expériences nous permettent de prouver que la précision des prédictions est essen-

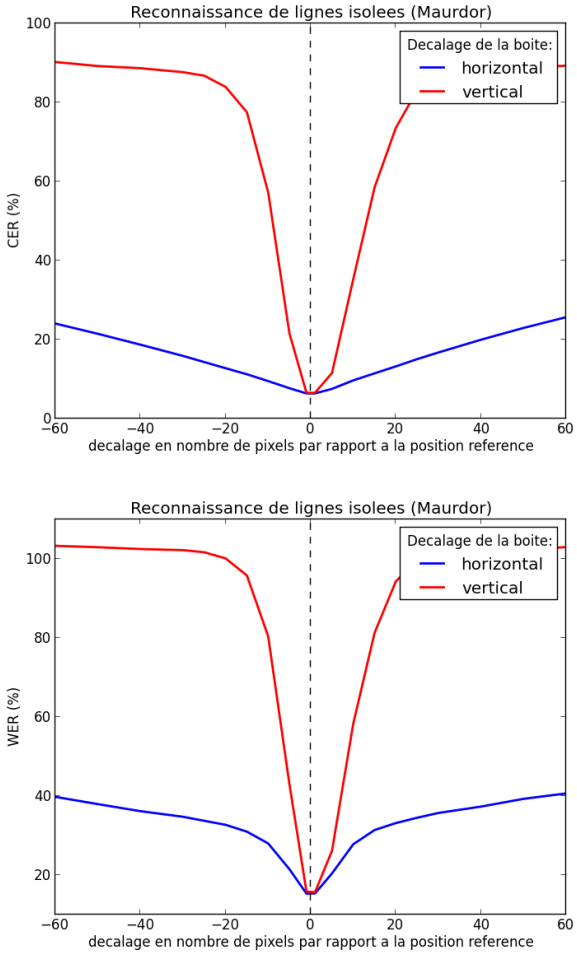


Fig. 6.1 – Taux de reconnaissance caractère (CER) et taux de reconnaissance mot (WER) en fonction du décalage en pixels de la boîte pour une image en 200 dpi.

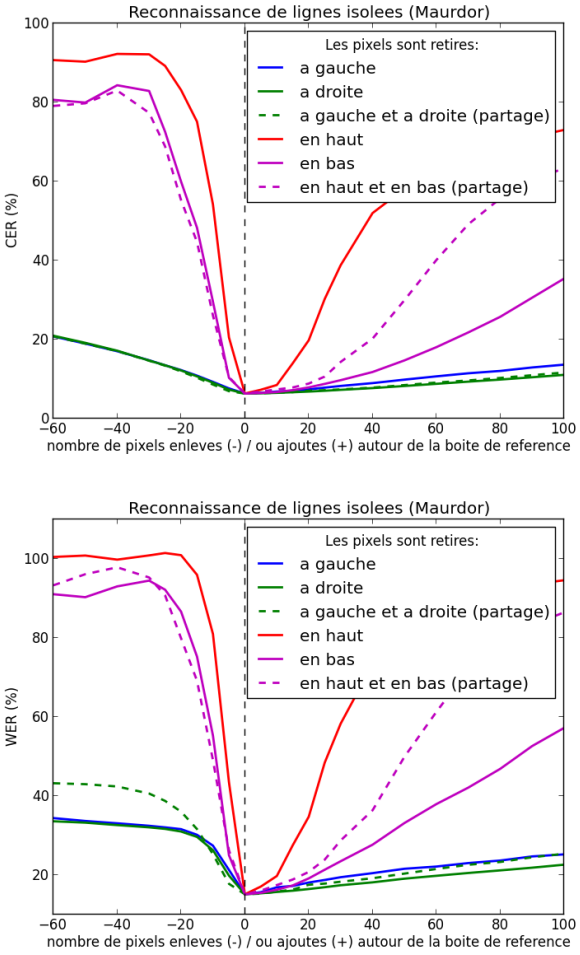


Fig. 6.2 – Taux de reconnaissance caractère (CER) et taux de reconnaissance mot (WER) en fonction de l'agrandissement ou du rétrécissement de la boîte.

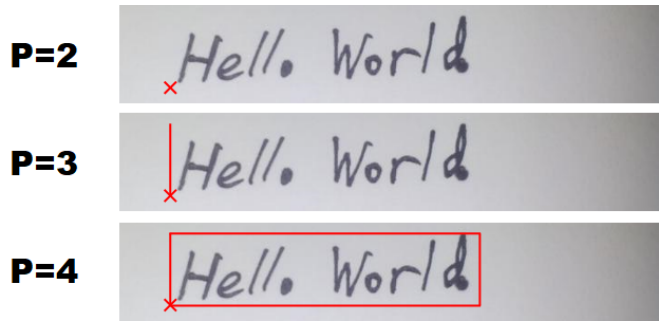


Fig. 6.3 – Nature des objets prédits en fonction du nombre de paramètres P prédits.

tielle.

6.1.2 Finesse des prédictions en fonction du nombre de coordonnées prédites

Comme la précision des prédictions est importante, nous avons étudié la performance de nos systèmes en fonction de celle-ci. Pour cela, nous avons utilisé une métrique à base de F-Mesure où un objet hypothèse défini par P coordonnées $\boldsymbol{h} = \{h_1, ..., h_P\}$ est considéré comme correct si il est l'objet hypothèse le plus proche d'un objet référence $\boldsymbol{r} = \{r_1, ..., r_P\}$ et si il est situé dans l'hypercube de côté D autour de cet objet référence. C'est à dire si :

$$\forall i \in \{1,...,P\}, |h_i - r_i| \leq D/2 \tag{6.1}$$

Cet hypercube peut être un tesseract si on s'intéresse à la position des 4 coordonnées définissant les boîtes englobant les objets mais cela peut également être un carré si on s'intéresse à un coin de ces boîtes englobantes ou un cube si on s'intéresse à un côté.

A l'aide de cette métrique, nous avons analysé l'impact sur la précision des prédictions du nombre de coordonnées P prédites par le réseau. C'est à dire que nous avons entraîné, de la manière décrite en Section 5 trois réseaux différents. Le premier est entraîné à détecter les boîtes englobant les lignes de texte ($\boldsymbol{O} = \{gauche, bas, hauteur, largeur, confiance\}$ et $P = 4$), le second est entraîné à détecter les coins inférieurs gauches de ces boîtes ($\boldsymbol{O} = \{gauche, bas, confiance\}$ et $P = 2$), comme dans [Moysset et al., 2015a], et le troisième est entraîné à en détecter les côtés gauches ($\boldsymbol{O} = \{gauche, bas, hauteur, confiance\}$ et $P = 3$). La nature des objets prédits, en fonction de P est illustrée en Figure 6.3.

L'objectif de cette expérience est de mesurer l'impact de la prédiction des valeurs de largeur et de hauteur de la boîte sur les autres prédictions. En effet, comme nos prédictions sont locales,

TABLE 6.1 – Comparaison des scores de F-Mesure sur la détection des coins inférieurs gauches en fonction de la taille de la zone d'acceptation pour des réseaux entraînés à détecter un nombre de coordonnées P par objet variable ; à savoir des réseaux entraînés à détecter respectivement les coins inférieurs gauches, les bords gauches ou les boîtes. Les tailles D des zones d'acceptation sont donnés en proportion de la largeur de la page.

Taille D de la zone d'acceptation	0.003	0.01	0.03	0.1
Coins inférieurs gauches, P=2	10.7%	57.4%	85.7%	91.7%
Côtés gauches, P=3	11.2%	58.4%	87.0%	92.6%
Boîtes, P=4	6.8%	45.0%	82.8%	89.9%

TABLE 6.2 – Comparaison des scores de F-Mesure sur la détection des trois coordonnées définissant les bords gauches des boîtes englobantes en fonction de la taille de la zone d'acceptation pour des réseaux entraînés à détecter un nombre de coordonnées P par objet variable ; à savoir des réseaux entraînés à détecter respectivement les bords gauches ou les boîtes. Les tailles D des zones d'acceptation sont donnés en proportion de la largeur de la page.

Taille D de la zone d'acceptation	0.003	0.01	0.03	0.1
Côtés gauches, P=3	4.2%	47.2%	84.8%	92.4%
Boîtes, P=4	3.4%	24.6%	71.4%	89.7%

certains objets peuvent ne pas être entièrement présents dans les champs réceptifs convolutionnels et cela peut être visualisé dans la Figure 5.1 qui montre lesdits champs réceptifs. Les couches de récurrence sous forme de cellules LSTMs permettent de transmettre le contexte lié à la page et donc de prédire ces valeurs. Mais on peut se demander quel impact cette charge de travail supplémentaire demandée aux LSTMs a sur la prédiction des autres coordonnées.

Le tableau 6.1 montre la précision de la détection des coins inférieurs gauches en fonction du nombre de coordonnés prédites par objet hypothèse alors que le tableau 6.2 évalue la précision de la détection des 3 coordonnées déterminant les bords gauches des boîtes englobant les objets.

Pour rappel, le tableau 5.1 qui donne des statistiques sur les objets de la base Maurdor nous indique que la hauteur moyenne des lignes de texte par rapport à la largeur de la page est de 0.0216. La largeur moyenne est, quant à elle, de 0.2327 mais sa prédiction est moins critique, comme illustré en section 6.1.

On observe que, si les performances entre les trois réseaux sont sensiblement similaires pour des tailles de zones d'acceptation grandes, les performances du système prédisant des boîtes sont moins bonnes pour des tailles de zones d'acceptation valant 0.01 ou 0.03 fois la largeur de la page. C'est à dire pour des tailles de zone d'acceptation proches de la hauteur moyenne des lignes.

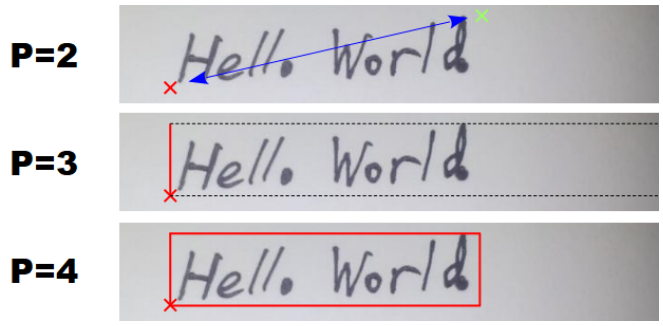


Fig. 6.4 – Illustration des trois stratégies de détection des lignes de texte. Les couleurs différentes utilisées signifient que des réseaux différents sont utilisés. Les pointillés noirs indiquent une information déduite.

On remarque, par contre, que prédire les côtés gauches, et donc les hauteurs en plus des coins inférieurs gauches ne nuit pas aux performances et aurait même tendance à les améliorer. Ces résultats sont confirmés par l'étude de la prédiction des côtés gauches où ajouter la détection de la largeur dégrade nettement les performances pour des tailles de zones d'acceptation intermédiaires.

Cette observation du fait que la détection de la largeur puisse handicaper la fiabilité de la précision des détections de boîtes est importante lorsque elle est mise en relation avec les observations de la section 6.1 qui montrent l'importance de ladite précision. Elle a servi de motivation pour proposer trois stratégies différentes de localisation de lignes pour la reconnaissance pleine page qui sont détaillées dans la section suivante.

6.2 Description de trois stratégies de reconnaissance pleine page

L'importance de fiabilité et de précision des prédictions des positions lors de la localisation des boîtes englobant les lignes de texte et la description de l'impact du nombre de coordonnées P prédites par le réseau sur cette précision ont été décrits en Section 6.1. Dans cette partie, trois stratégies de reconnaissance de texte dans des pages complètes, qui utilisent un nombre de coordonnées par objet hypothèse variable, sont détaillées. Elles seront évaluées en Section 7.3. La première, décrite en Section 6.2.1, utilise une prédiction directe de boîtes ($P = 4$). La seconde, décrite en Section 6.2.2, détecte séparément les coins supérieurs-gauches et les coins inférieurs-droits des boîtes englobantes ($P = 2$) puis apparie ces coins à l'aide d'un troisième réseau de neurones pour former les boîtes. Enfin la troisième, décrite en Section 6.2.3 utilise une prédiction des bords gauches des boîtes englobantes ($P = 3$) et charge le reconnaisseur de texte de prédire la position de la droite du texte.

6.2.1 Détection directe des boîtes englobant les lignes

La première stratégie est la stratégie, immédiate, employée par les modèles profonds génériques de détection et de reconnaissance d'objet dans les images de scènes naturelles : régression de toutes les coordonnées des boîtes englobant les objets (coins inférieurs gauches et supérieurs droits ; ou, alternativement, un coin accompagné de la largeur et de la hauteur de la boîte).

Dans ce cas, on va prédire les boîtes englobant nos objets. Le réseau va directement prédire les objets $\boldsymbol{O}$ décrits dans l'équation 5.3, soit la position du point en bas à gauche de l'objet, sa largeur, sa hauteur et la confiance que l'objet existe. Soit :

$$\boldsymbol{O} = \{gauche, bas, hauteur, largeur, confiance\} \tag{6.2}$$

Au décodage, seuls les objets dont la confiance est supérieure à un seuil $T = 0.5$ sont conservés.

On a constaté, et l'illustration en est donnée en Figure 6.2, que rogner une ligne avait une influence plus négative que de l'élargir. Cela s'explique simplement par le fait que les parties retirées à l'image de la ligne comportent en règle général du texte dont les mots ne pourront plus être correctement reconnus, induisant une baisse de performance. Alors que le fait d'ajouter de l'image autour du texte à reconnaître complique la tâche du reconnaisseur mais ne la rend pas impossible, d'autant plus dans les cas ou seul du fond est ajouté. Pour tenir compte de cela, nous ajoutons une marge fixe de 20 pixels autour des boîtes prédites.

Cette technique de détection de boîtes, et donc les améliorations décrites dans le chapitre 5, ont permis une amélioration importante des performances par rapport à la technique MultiBox et par rapport aux méthodes à base de traitements d'image auxquelles nous nous comparerons dans les tableaux 7.6, 7.7 et 7.8.

Néanmoins, comme montré dans la section précédente, cette stratégie n'est pas optimale dans notre cas où le modèle est local et dans lequel la régression est partagée entre les différentes positions de l'image. En particulier, la fin de la ligne de texte est souvent en dehors du champs réceptif convolutionnel relatif à la sortie et prédire précisément cette position est une tâche difficile dont la charge est entièrement supportée par les couches récurrentes. Ces couches de contexte peuvent amoindrir le problème mais ne peuvent pas le résoudre complètement.

6.2.2 Détection de coins des boîtes englobant les lignes et appariement

La deuxième stratégie [Moysset et al., 2016] utilise les résultats de la section 6.1.2 et le fait que la précision des prédictions est meilleure lorsque l'on détecte uniquement des points. Deux réseaux différents sont chargés de, respectivement, prédire les coins inférieurs gauches et les coins supérieurs droits de la boîte englobant les objets. Ces réseaux utilisent la même architecture mais leurs paramètres entraînés sont différents. Puisque les objets détectés sont des points, ils tombent complètement dans le champs réceptif relatif à des sorties et la charge sur les couches récurrentes est donc allégée. Les coins inférieurs gauches et supérieurs droits qui sont détectés (et donc qui ont une valeur de confiance supérieure au seuil de détection T) sont appariés pour former les prédictions de boîtes englobantes désirées. Cet appariement est effectué en minimisant une fonction d'énergie sous la contrainte que chaque coin inférieur gauche n'est apparié qu'à un seul coin supérieur droit et vice versa. La fonction d'énergie utilisée, détaillée dans l'équation 6.3 est minimisée par rapport à la variable discrète $\boldsymbol{z}=\{z_{ij}\}$ qui vaut 1 lorsque le coin inférieur gauche $l_{ll}(i)$ est apparié avec le coin supérieur droit $l_{ur}(j)$.

$$\begin{aligned} \hat{\boldsymbol{z}} &= \arg\min_{\boldsymbol{z}} \sum_i \sum_j z_{ij} D(l_{ll}(i), l_{ur}(j)) \\ \text{s.t.} \quad & \forall j \textstyle\sum_i z_{ij} \in \{0,1\}, \quad \forall i \sum_j z_{ij} \in \{0,1\} \end{aligned} \tag{6.3}$$

Ce problème de satisfaction de contraintes peut être résolu à l'aide de l'algorithme Hongrois [Munkres, 1957].

La fonction $D(.,.)$ est une distance responsable d'évaluer si un coin inférieur gauche et un coin supérieur droit doivent être appariés, ou en d'autres mots, de prédire à quel point la boîte formée par ces deux coins est susceptible d'être une boîte englobant une ligne de texte. Cette distance est mise en place à l'aide d'un troisième réseau de neurones, convolutionnel, entraîné sur un large nombre d'images définies par des paires de coins inférieurs gauches et de coins supérieurs droits et augmentés d'une bordure de 10 pixels pour en voir l'environnement immédiat. Certains de ces couples appartiennent au même objet référence, d'autres non. Le réseau, composé de 6 couches convolutionnelles, suivi d'une somme des éléments des cartes de caractéristiques de la dernière couche, prend pour entrée l'image incluse dans la boîte formée par le couple de points candidats et donne une classification de cette image étant, ou non, une ligne de texte. La probabilité a posteriori de cette décision, c'est à dire la soft-max de cette sortie correspond à la distance D utilisée pour l'appariement.

Afin de ne conserver que des couples de coins plausibles, seuls les couples pour lesquels le coin supérieur droit est, à la fois, à droite, au dessus, et verticalement à moins de 600 pixels par

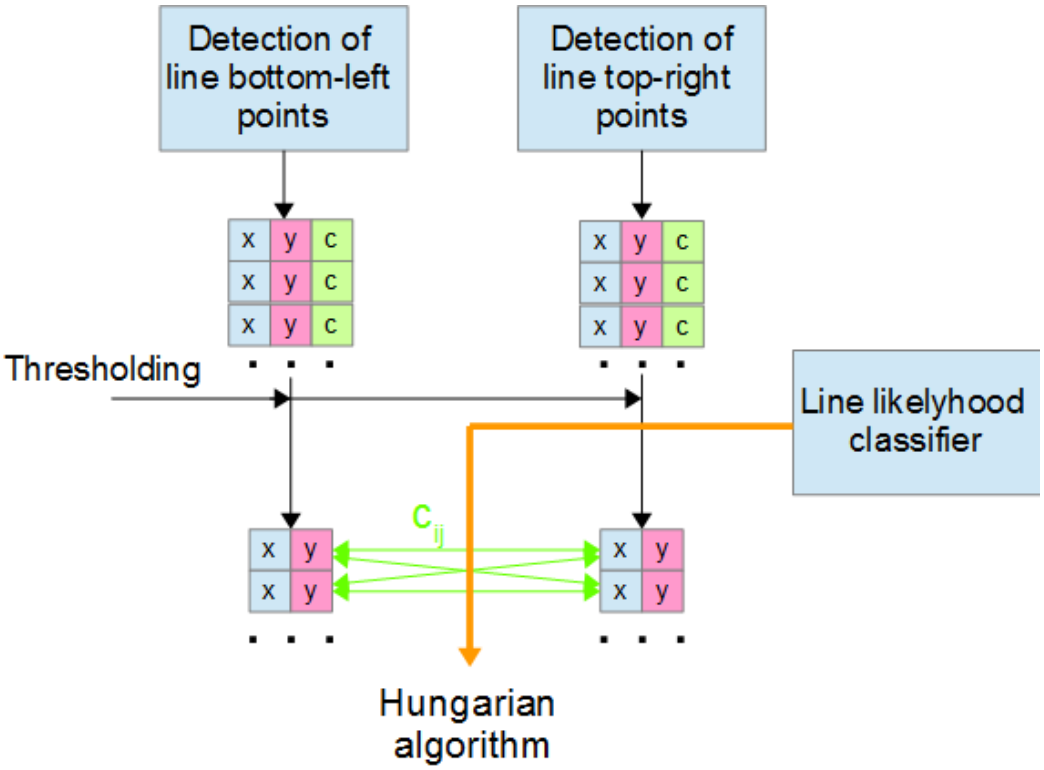


Fig. 6.5 – Illustration de la méthode d'appariement des coins. Trois réseaux de neurones différents sont utilisés pour, respectivement, la détection des coins inférieurs gauches, la détection des coins supérieurs droits des boîtes englobant les lignes de texte, et pour le calcul des probabilités d'appariement.

rapport au coin inférieur gauche sont considérés. Cela à la fois lors de l'entraînement et lors du décodage. Ces règles ont également pour avantage de permettre d'équilibrer les classes lors de l'entraînement et de diminuer le nombre de couples et donc le temps de calcul lors du décodage.

Cette stratégie de détection de boîtes et l'utilisation des trois réseaux de neurones est illustrée en Figure 6.5.

Utiliser cette stratégie de détection et d'appariement des coins peut permettre d'améliorer la précision des résultats. Mais le système global est rendu plus complexe puisque trois réseaux différents doivent être entraînés simplement pour l'étape de détection des lignes. De plus, le temps de calcul est plus important puisque la détection est effectuée deux fois (pour les coins inférieurs gauches et pour les coins supérieurs droits) et en raison du fait que le réseau donnant la distance pour l'appariement doit être utilisé pour chaque couple de coins candidats.

On observe de plus que le mécanisme d'appariement peut être responsable de certaines erreurs et que les erreurs de détection des coins gauches et droits se cumulent.

6.2.3 Détection des bords gauches des boîtes englobant les lignes

Les expériences de la Section 6.1.2 ont montré que, si la détection de la largeur des lignes de texte perturbe la précision des prédictions, cela n'est pas le cas pour la prédiction des hauteurs des lignes. Cela est probablement dû au fait que les hauteurs des lignes sont plus petites que leurs largeurs et, par conséquent, restent incluses dans les champs réceptifs relatifs aux objets.

Pour cette raison, nous proposons une nouvelle stratégie [Moysset et al., 2017] dans le but d'à la fois être plus précis que l'approche prédisant directement les boîtes englobantes décrite en Section 6.2.1 et d'éviter le besoin d'optimisation combinatoire et d'appariement exprimé par l'approche à base de détection de coins décrite en Section 6.2.2. Pour cela, nous utilisons un système entraîné à détecter uniquement les coordonnées définissant les bords gauches des boîtes englobant les lignes de texte, et donc la position de leurs coins inférieurs gauches ainsi que leurs hauteurs, mais pas leurs largeurs. Au lieu de prédire cette largeur, elle est déterminée par l'étape de *reconnaissance de texte* qui succède à la détection. Le reconnaisseur va à la fois prédire le contenu du texte et quand le texte est terminé.

Il est donné comme entrée à ce reconnaisseur de texte une image dont le côté gauche est la position prédite par le réseau de localisation et dont le côté droit est le bord droit de l'image de la page correspondant. Cela signifie que d'autres objets textuels peuvent être présents dans cette partie droite ajoutée. Le reconnaisseur de texte est entraîné pour apprendre à les ignorer. Cette stratégie est illustrée en Figure 6.6.

Pour valider le fait que le reconnaisseur de texte soit capable de réaliser cette tâche et donc de déterminer la position de la droite de la ligne de texte, nous comparons dans le Tableau 6.3 les taux de reconnaissance de texte obtenus pour des reconnaisseurs de texte entraînés et évalués sur les boîtes références ou sur les boîtes définies uniquement par les bords gauches de ces boîtes références et étendues jusqu'à la limite droite de la page, incluant potentiellement d'autres objets textuels.

On observe que le taux d'erreur du système entraîné et évalué sur les images définies uniquement par les bords gauches des lignes de texte vaut 9.8% et est proche du taux de reconnaissance de 9.0% du réseau entraîné et évalué à reconnaître le contenu des boîtes références. Ce faible écart justifie la pertinence de cette stratégie puisque cela montre que le réseau a appris à faire abstraction des objets présents à droite de la ligne sans trop perturber la qualité de reconnaissance du texte.

On note qu'un réapprentissage du reconnaisseur de texte est nécessaire puisque le système entraîné sur les boîtes références est très mauvais lorsqu'il est évalué sur les images définies par les côtés gauches uniquement. Ce qui s'explique aisément par la présence d'autres éléments textuels à droite du texte à reconnaître ; éléments qui seront considérés comme des insertions.

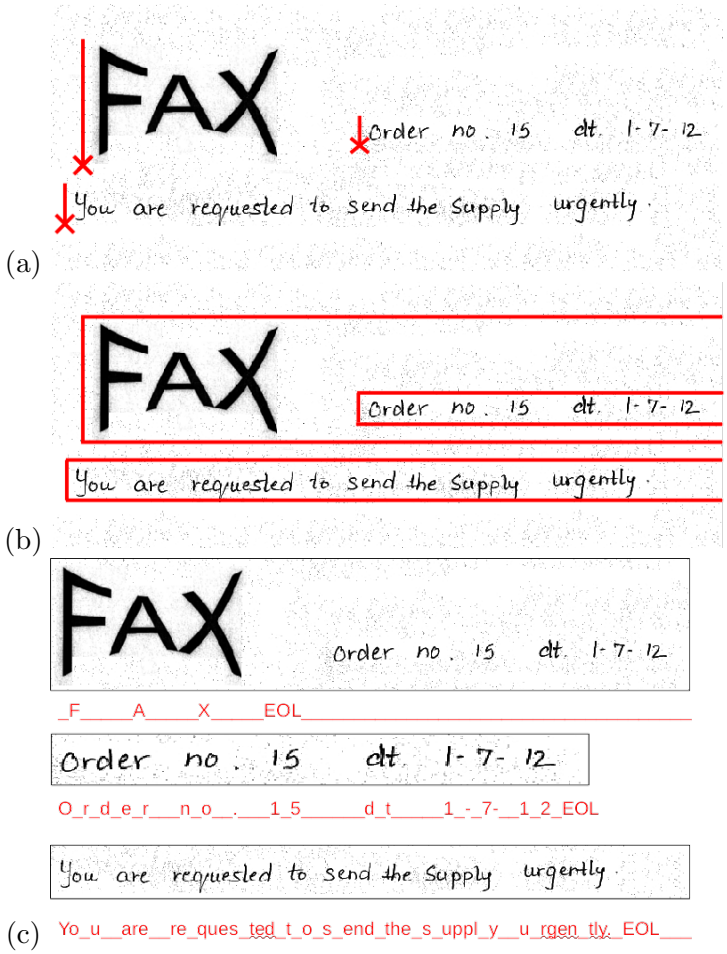


Fig. 6.6 – Description de la stratégie de localisation à base de détection des bords gauches des boîtes englobantes. (a) Détection des bords gauches des lignes de texte. (b) Extraction des images lignes correspondantes. (c) Reconnaissance du contenu des lignes de texte à l'aide d'un reconnaisseur ayant appris à ignorer le texte appartenant à d'autres lignes. Le caractère "fin de ligne" ajouté (EOL) permet de faciliter cet entraînement.

TABLE 6.3 – Taux d'erreur mot (WER) pour des réseaux de reconnaissance de texte entraînés et évalués sur des boîtes références ou des boîtes définies seulement par les côtés gauches et étendues jusqu'au bord droit de l'image de la page. Pour les lignes en anglais de l'ensemble de validation de la base Maurdor.

Entraîné sur	Évalué sur les boîtes références	Évalué sur les boîtes définies par les côtés gauches
Les boîtes références	9.0%	46.7%
Les boîtes définies par les côtés gauches	10.6%	9.8%

Chapitre 7

Expériences

Après avoir défini, dans le chapitre 5, les principales contributions apportées à notre modèle de localisation de lignes et après avoir proposé trois stratégies de reconnaissance pleine page dans le chapitre 6, nous allons dans ce chapitre analyser les résultats de notre modèle et de ses différentes variantes. Nous commençons par, en Section 7.1, décrire le cadre expérimental dans lequel sont effectuées nos expériences et, en particulier, quelles bases de données et quelles métriques nous utilisons. Nous allons ensuite, en Section 7.3, comparer les résultats de nos différentes stratégies de détection de lignes de texte aussi bien entre elles que par rapport à des techniques de référence. Nous analysons ensuite, en Section 7.4 l'influence de certains paramètres importants de notre modèle. En particulier, nous nous intéressons à la transmission de l'information contextuelle à l'aide de couches de 2D-LSTMs en Section 7.4.1, à la nature de la couche de sortie en Section 7.4.2 et à la fonction de coût utilisée en Section 7.4.3. Enfin, nous étudions la répartition des temps de calculs en Section 7.5.

7.1 Protocole Expérimental

La Section 7.1.1 décrit les bases de données utilisées dans ce travail et justifie la création de différents sous ensembles alors que la Section 7.2 décrit les métriques utilisées. A la fois les bases de données et les métriques sont utilisées dans les résultats de reconnaissance pleine page de la Section 7.3 et dans les études ablatives de la Section 7.4 qui justifient les choix de conception du modèle décrit dans le chapitre 5.

7.1.1 Bases de données

La base que nous avons majoritairement utilisée dans nos travaux est la base Maurdor [Brunessaux et al., 2014]. Cette base a pour avantage de réunir un certain nombre de caractéristiques :

— Elle est publique. [1]
— Elle contient un nombre de données (8774 pages) bien plus grand que les autres bases de documents.
— Les documents qui la compose sont fortement différents les uns des autres.

Description de la base Maurdor

L'hétérogénéité de la base Maurdor est soulignée par la présence de trois langues différentes (français, anglais, arabe) et d'à la fois de l'écriture manuscrite et imprimée. Mais aussi par la présence de documents venant de sources très différentes comme des lettres manuscrites, imprimées, des formulaires ou des articles de journaux. Cette hétérogénéité, illustrée par des exemples d'images de pages en figure 7.1, fait de la base Maurdor un véritable challenge technique et permet de se rapprocher de tâches réelles pour obtenir un système le plus générique possible.

La base officielle contient 192536 zones de texte dans 8774 pages. La répartition entre les différentes langues et les différents scripts, ainsi que les tailles des ensembles officiels d'entraînement, de validation et de test sont détaillées dans le tableau 7.1.

Les documents qui composent la base Maurdor sont séparés en cinq catégories de documents qui en soulignent la diversité. Ces catégories sont expliquées dans le tableau 7.2. Les documents appartenant à ces catégories sont séparés de manière homogène entre les ensembles d'entraînement, de validation et de test.

Sous-ensembles de la base Maurdor pour l'évaluation

Si l'hétérogénéité de la base est une chance, elle pose certaines contraintes. En particulier, comme l'on ne sait pas la langue présente dans les lignes de texte localisées, et comme les reconnaisseurs de texte sont spécifiques à chaque langue, il est nécessaire d'ajouter une étape intermédiaire de classification de la langue. Cette classification n'est pas évidente pour les langues partageant le même script (comme par exemple le français et l'anglais dans notre cas) et les erreurs à ce stade peuvent réduire l'interprétabilité des résultats de la localisation des lignes de texte.

1. http://catalog.elra.info/product_info.php?products_id=1242

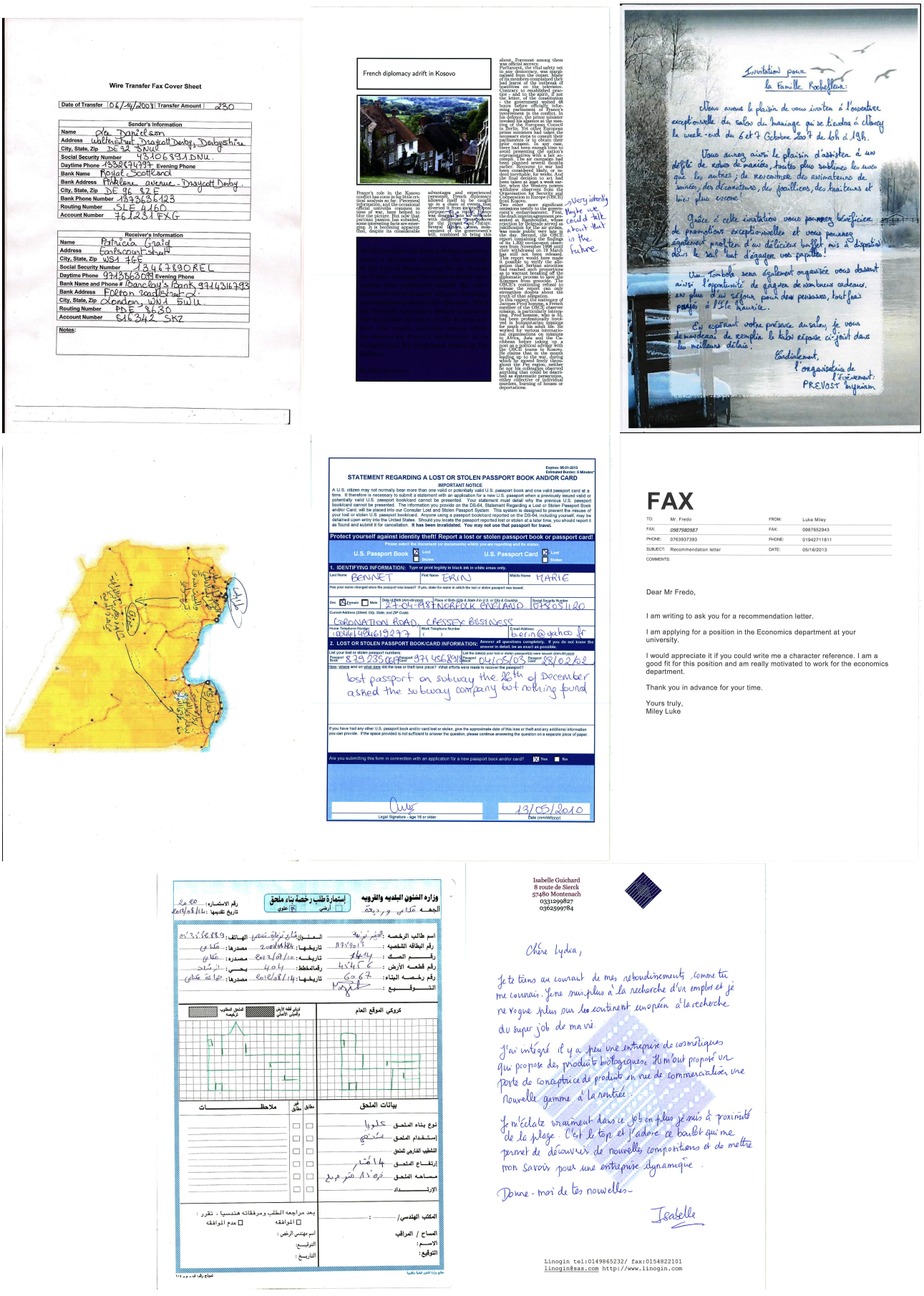


Fig. 7.1 – Exemples d'images de pages provenant de la base Maurdor.

TABLE 7.1 – Base Maurdor : ensembles officiels d'entraînement, de validation et de test avec le nombre de zones de texte pour chaque langue et type d'écriture.

Ensemble	Pages	Zones					
		Zones imprimées			Zones manuscrites		
		Français	Anglais	Arabe	Français	Anglais	Arabe
Entraînement	6 592	141 683					
		105 002				36 681	
		57 821	25 773	21 263	18 417	8 530	9 729
Validation	1 110	25 663					
		19 205				6 458	
		9 908	5 124	4 122	2 857	1 765	1 835
Test	1 072	25 180					
		18 907				6 273	
		11 519	4 131	3 210	3 241	1 450	1 582
Total	8 774	192 526					
		143 114				49 412	
		79 248	35 028	28 595	24 515	11 745	13 146

Pour cette raison, nous avons sélectionné trois sous-ensembles de la base Maurdor dont les pages ne contiennent qu'une et une seule langue. Ces bases restent mixtes manuscrit-imprimé. Ces bases sont utilisées pour l'évaluation uniquement (les entraînements sont, sauf mention contraire, faits sur l'ensemble complet multi-langues) et les tailles des ensembles obtenus sont illustrées dans le tableau 7.3.

Sous-ensembles de la base Maurdor pour l'entraînement des reconnaisseurs de texte

La base Maurdor est une base très riche. Mais elle a un inconvénient majeur vis à vis des tâches que nous réalisons. L'annotation des champs textuels est réalisée au niveau paragraphe et pas au niveau ligne. On notera la présence dans l'annotation des paragraphes de caractères 'sauts de ligne', ce qui nous donne l'information du nombre de lignes et du contenu de chacune de ces lignes. Nous avons réalisé une annotation automatique afin d'obtenir les positions des lignes de texte associées à ces contenus. Obtenir les positions des lignes (et donc les images correspondantes à ces lignes) et les contenus associés est nécessaire pour apprendre les reconnaisseurs de texte. Obtenir la position des lignes est aussi nécessaire pour apprendre nos détecteurs de lignes. Ces travaux ont été décrits [Bluche et al., 2014] et utilisés pour entraîner nos systèmes de reconnaissance pour la compétition Maurdor avec succès [Moysset et al., 2014a]. Cette technique est décomposée en

TABLE 7.2 – Description des différentes catégories des documents de la base Maurdor

Catégorie	Description	Exemples	Nombre
C1	Formulaires imprimés remplis en manuscrit	Formulaires administratifs, formulaires d'inscription ou d'abonnement, formulaires d'enquête	1068
C2	Documents commerciaux, privés ou professionnels, imprimés ou photocopiés	Bons de commande, pages de catalogues, tracts commerciaux, articles de presse, contrats , notes de frais, devis	3575
C3	Correspondances privées manuscrites	courriers manuscrits, notes personnelles manuscrites, cartes de félicitations	1807
C4	Correspondances privées ou professionnelles dactylographiées	courriers dactylographiés, notes de service, impressions de courriels	1377
C5	Autres	plans, schémas, dessins, tableaux de chiffres	300

TABLE 7.3 – Taille des sous-ensembles mono-langues de la base Maurdor pour l'évaluation pleine page.

Ensemble	Français	Anglais	Arabe
Validation	533 pages	267 pages	111 pages
Test	507 pages	265 pages	141 pages

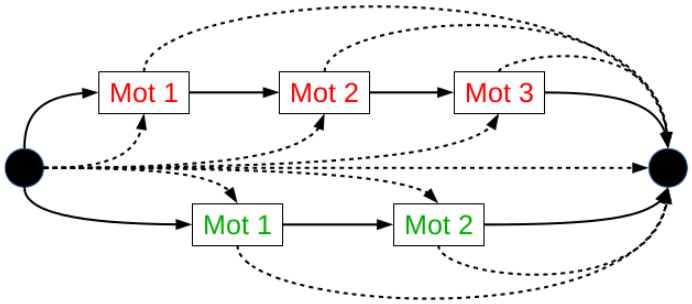


Fig. 7.2 – Illustration du graphe de décodage utilisé pour contraindre la reconnaissance lors de l'étape d'annotation automatique des données. Ici un paragraphe de deux lignes (la première composée de trois mots rouges, la seconde de deux mots verts). Seuls les transcriptions ayant choisi des chemins ne suivant que des flèches pleines, et donc passant par tous les mots d'une ligne, seront conservés.

trois étapes.

Premièrement, les paragraphes ne comportant qu'une ligne de texte sont mis de côté puisque leur contenu et la position de la ligne sont déjà connus. Les paragraphes contenant plusieurs lignes sont segmentés en lignes de texte à l'aide de trois algorithmes de détection de lignes pré-existants, nommément des adaptations de Shi et al [Shi et al., 2009], Nicolaou et al. [Nicolaou and Gatos, 2009] et un algorithme qui agrège les morceaux de composantes connexes pour former les lignes [Moysset et al., 2014a]. Notons que le fait que cette segmentation soit réalisée au niveau des paragraphes et pas au niveau des pages complètes simplifie ce problème de segmentation.

Deuxièmement, les contenus de ces hypothèses de segmentation sont transcrits à l'aide d'un reconnaisseur de texte. Ce reconnaisseur pourra être initialement entraîné uniquement sur les lignes provenant de paragraphes mono-lignes, puis sera itérativement ré-entraîné en ajoutant les lignes nouvellement trouvées. Cette reconnaissance est contrainte par un graphe de décodage qui ne permet de reconnaître que le texte présent dans l'annotation du paragraphe, comme illustré en figure 7.2. Pour tenir compte du fait que les segmentations en lignes peuvent manquer certaines parties de la ligne, il est donné comme possibilité à ce graphe de commencer au début de chaque mot et de finir après chaque mot de la ligne. Il est aussi possible de ne rien reconnaître. L'idée de cette reconnaissance contrainte réside dans le fait que le reconnaisseur utilisé n'est pas suffisamment bon pour reconnaître le contenu exact d'une ligne mais pourra en revanche choisir si une ligne est présente et discriminer entre les lignes du paragraphe.

Troisièmement, seules les lignes complètement reconnues sont conservées, les autres sont écartées. Si plusieurs hypothèses de segmentation reconnaissent correctement la même ligne, celle avec le plus fort taux de confiance est conservée.

TABLE 7.4 – Nombre de lignes trouvées grâce à l'annotation automatique et proportion du total de lignes.

Ensemble	Français	Anglais	Arabe
Entraînement	124599 (91.0%)	60044 (84.4%)	43320 (86.0%)
Validation	21170 (91.2%)	11903 (89.6%)	7187 (90.8%)

TABLE 7.5 – Tailles des sous-ensembles de la base Maurdor pour l'entraînement de la localisation des lignes. Seules les pages où toutes les lignes sont automatiquement annotées sont conservées.

Ensemble	Nombre de pages conservées	Nombre de pages total	Proportion
Entraînement	3995 pages	6592 pages	60.6%
Validation	697 pages	1110 pages	62.8%
Test	616 pages	1071 pages	57.5%

Le système se stabilise après deux itérations du cycle création de données annotées automatique - entraînement des réseaux de reconnaissance sur ces données, le nombre de lignes obtenues est indiqué dans le tableau 7.4.

Sous-ensembles de la base Maurdor pour l'entraînement de détecteurs de lignes

Pour entraîner les réseaux de localisation des lignes, seules les positions des lignes sont nécessaires. Mais il est souhaitable d'avoir toutes les lignes appartenant à une page pour éviter la présence de lignes non annotées qui constitueraient un bruit. Pour les mêmes raisons, il est souhaitable que les lignes soient correctement localisées. Nous utilisons donc une technique similaire à celle utilisée précédemment et nous conservons uniquement les pages pour lesquelles toutes les lignes références ont été associées à une hypothèse de segmentation. C'est à dire que nous conservons uniquement les pages dans lesquelles nous pensons avoir une annotation correcte de toutes les lignes.

Nous constatons dans le tableau 7.5 que une proportion d'environ 60% des pages de l'ensemble d'entraînement a pu être conservée.

7.2 Métriques

Plusieurs métriques ont été utilisées dans nos travaux. Certaines mesurent si la position des objets est bien trouvée sur des critères géométriques. D'autres ont recours à une reconnaissance du texte pour évaluer la détection en combinaison avec la tâche finale. Ces méthodes ont chacune des avantages et des inconvénients. Elles sont décrites dans cette section.

7.2.1 Les métriques géométriques

Tout d'abord, nous détaillons les métriques basées sur la géométrie des objets détectés. Elles ont pour avantage de se faire uniquement sur la position des objets et d'être peu intensives d'un point de vu du temps de calcul. De plus, la tâche finale de reconnaissance du texte n'est pas réalisée, ce qui évite de devoir sélectionner la langue présente dans chaque ligne et de cumuler les erreurs liés à la détection et à la reconnaissance.

Intersection sur Union

Une première métrique que nous avons utilisée, basée sur la géométrie des objets, est l'intersection sur union (IoU). Comme son nom l'indique, le critère de sélection est basé sur les aires de l'intersection et de l'union des objets références et hypothèses.

$$IoU(A,B) = \frac{Aire(A \cap B)}{Aire(A \cup B)} \tag{7.1}$$

Avec A un rectangle hypothèse et B un rectangle référence. On seuille ensuite les couples selon un seuil T, on considère un couple (A,B) comme correct si $IoU(A,B) > T$. Si une boîte hypothèse est associée à plusieurs boîtes références, seule l'association avec la boîte référence ayant la valeur d'IoU maximale est conservée. La boîte référence est alors retirée de la liste. On calcule ensuite précision, rappel et F-Mesure. Pour rappel :

$$Precision = \frac{Nombre\ corrects}{Nombre\ hypotheses} \tag{7.2}$$

$$Rappel = \frac{Nombre\ corrects}{Nombre\ references} \tag{7.3}$$

$$F - Mesure = 2\,\frac{Precision \times Rappel}{Precision + Rappel} \tag{7.4}$$

detEval

L'algorithme detEval [Wolf and Jolion, 2006], utilisé fréquemment dans les compétitions d'objets textuels [Karatzas et al., 2015], permet de prendre en charge les cas où une boîte référence se superpose avec plusieurs boîtes hypothèses et vice versa.

En notant $\sigma_{i,j}$ le rappel au niveau des pixels entre une boîte référence G_i et une boîte hypothèse D_j, en notant similairement $\tau_{i,j}$ la précision pixelique, et en définissant des seuils t_r et t_p sur ces rappels et précisions, on associe la boîte G_i à la boîte G_j si $\sigma_{i,j} \geq t_r$ et $\tau_{i,j} \geq t_p$. On peut ainsi définir un rappel R_{OB} et une précision P_{OB} au niveau objet entre les ensembles de boîtes références $\boldsymbol{G}$ et hypothèses $\boldsymbol{D}$.

$$R_{OB}(\boldsymbol{G},\boldsymbol{D},t_r,t_p) = \sum_i \frac{Match_G(G_i,\boldsymbol{D},t_r,t_p)}{|\boldsymbol{G}|} \tag{7.5}$$

Avec :

$$Match_G(G_i,\boldsymbol{D},t_r,t_p) = \begin{cases} 1 \textit{ si } G_i \textit{ est associé à un seul élément de } D \\ 0 \textit{ si } G_i \textit{ n'est associé à aucun élément de } D \\ f_{sc}(k) \textit{ si } G_i \textit{ est associé à } k \textit{ éléments de } D \end{cases} \tag{7.6}$$

Et,

$$P_{OB}(\boldsymbol{G},\boldsymbol{D},t_r,t_p) = \sum_j \frac{Match_D(D_j,\boldsymbol{G},t_r,t_p)}{|\boldsymbol{D}|} \tag{7.7}$$

Avec :

$$Match_D(D_j,\boldsymbol{G},t_r,t_p) = \begin{cases} 1 \textit{ si } D_j \textit{ est associé à un seul élément de } G \\ 0 \textit{ si } D_j \textit{ n'est associé à aucun élément de } G \\ f_{sc}(k) \textit{ si } D_j \textit{ est associé à } k \textit{ éléments de } G \end{cases} \tag{7.8}$$

Le terme $f_{sc}(k)$ permet d'apporter une pénalité sur le fait d'avoir sur-segmenté ou sous segmenté notre objet. Il est fixé à une valeur de 0.8 . Nous pouvons ensuite moyenner ces valeurs de

précision et de rappel pour différentes valeurs de seuils t_r et t_p. Nous prenons un nombre $T = 20$ de ces valeurs sur l'intervalle [0,1]. Lorsque l'on fait varier t_p, une valeur par défaut $t_r = 0.8$ est utilisée. Respectivement, lorsque l'on fait varier t_r, on utilise $t_p = 0.4$. Utiliser des valeurs moyennées de rappel R_{OV} et de précision P_{OV} permet une comparaison plus juste et robuste des algorithmes. On a :

$$R_{OV} = \frac{1}{2T} \sum_{i=1}^{T} \left(R_{OB}(\boldsymbol{G},\boldsymbol{D},\frac{i}{T},t_p) + R_{OB}(\boldsymbol{G},\boldsymbol{D},t_r,\frac{i}{T}) \right) \tag{7.9}$$

Et :

$$P_{OV} = \frac{1}{2T} \sum_{i=1}^{T} \left(P_{OB}(\boldsymbol{G},\boldsymbol{D},\frac{i}{T},t_p) + P_{OB}(\boldsymbol{G},\boldsymbol{D},t_r,\frac{i}{T}) \right) \tag{7.10}$$

Enfin, on peut calculer un score final sous forme de F-Mesure de ces précision et rappel moyennés.

$$detEvalScore = 2\,\frac{P_{OV} \times R_{OV}}{P_{OV} + R_{OV}} \tag{7.11}$$

ZoneMap

La métrique ZoneMap a été développée par le Laboratoire National de métrologie et d'Essais (LNE) pour la compétition Maurdor [Galibert et al., 2014] [Brunessaux et al., 2014]. Son but premier est d'évaluer la détection et la localisation des paragraphes présents dans des images de documents mais cela peut également s'appliquer à l'évaluation de la segmentation des lignes.

Cette métrique est décomposée en deux étapes. Tout d'abord, les objets références et hypothèses sont réunis par groupes. Puis un score est établi pour chaque groupe en fonction de la taille des zones de recouvrement.

La première étape va s'effectuer en calculant la force du lien entre chaque boîte référence R et chaque boîte hypothèse H. Cette force f est définie par l'équation suivante :

$$f(R,H) = \left(\frac{Surface(H \cap R)}{Surface(H)} \right)^2 + \left(\frac{Surface(H \cap R)}{Surface(R)} \right)^2 \tag{7.12}$$

Cela signifie que plus une zone référence et une zone hypothèse sont superposées, plus la force

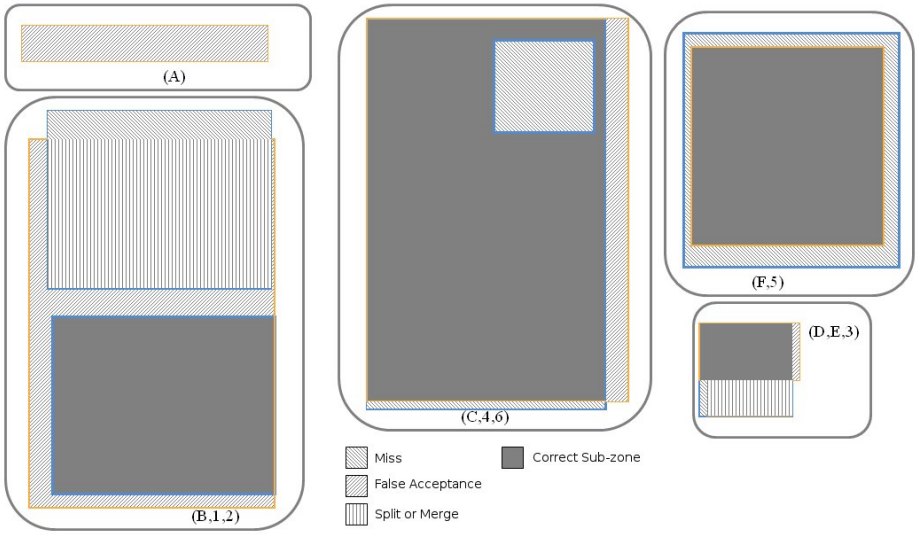


Fig. 7.3 – Illustration des différentes sortes de groupement réalisés par la métrique ZoneMap.

les liant sera forte. Ensuite, les zones sont groupées par force de lien décroissante tant qu'il n'y a pas à la fois plusieurs boîtes hypothèses et plusieurs boîtes références dans le même groupe.

Les groupes appartiennent donc tous à l'une des catégories illustrées en figure 7.3, c'est à dire un appariement, une fusion, une séparation, un oubli ou une fausse acceptation.

Dans le cas d'un appariement, l'erreur est la surface présente dans l'intersection des boîtes mais pas dans leur union. Pour une fusion ou une séparation, l'erreur est la surface présente dans l'intersection de toutes les boîtes du groupe mais pas dans l'union des boîtes du couple référence-hypothèse ayant le lien le plus fort. Enfin, pour un oubli ou une fausse acceptation, il s'agit de la surface de la boîte.

L'erreur totale, pour un document, est la somme des erreurs de tous les groupes normalisée par la surface totale des boîtes références.

Cette métrique, en prenant en compte les différents types d'erreurs possibles, permet d'obtenir des résultats relativement bien corrélés avec la tâche de reconnaissance du texte sur des tâches de segmentation de paragraphes en lignes [Moysset and Kermorvant, 2013]. On notera de plus que cette méthode présente, par rapport à une méthode plus simple comme l'IoU, l'avantage de nous donner des informations sur le type d'erreurs commises (c'est à dire la proportion de fusions, de séparations, d'oublis ou de fausses acceptations).

Position des points

On peut également considérer un objet hypothèse (de vecteur de position $\hat{\boldsymbol{x}}$), en particulier un point, comme correct si il est dans le voisinage immédiat d'un point référence (de vecteur de position $\boldsymbol{x}$). C'est à dire si :

$$\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_1 < S/2 \tag{7.13}$$

Où S est la largeur de la zone d'acceptation qui est carrée. On s'arrange en outre pour que seul un point hypothèse puisse être associé à chaque boîte référence. Et on calcule précision, rappel et F-Mesure en fonction de ce critère d'acceptation.

7.2.2 Les métriques de reconnaissance

Pour se rapprocher de la tâche finale, nous pouvons utiliser des métriques basées sur la bonne reconnaissance du texte compris dans les images de lignes. Si ces méthodes sont plus compliquées à mettre en place et nécessitent de savoir la langue à utiliser, elles permettent de comparer de manière plus juste des techniques ne faisant pas le même type d'erreurs.

Toutes ces méthodes utilisent des reconnaisseurs à base de réseaux récurrent [Moysset et al., 2014a] similaires à celui détaillé en partie 3.2 et utilisant des 2D-LSTMs dont le fonctionnement est décrit en section 5.4. Trois réseaux différents sont utilisés pour, respectivement, le français, l'anglais et l'arabe. Ces réseaux sont capables de reconnaître à la fois de l'imprimé et du manuscrit [Moysset et al., 2014b].

Le taux d'erreur mot

Le taux d'erreur mot (ou WER pour *Word Error Rate*) est une métrique qui compare deux séquences de texte, la séquence référence et la séquence hypothèse. Il est basé sur une distance de Levenshtein [Levenshtein, 1966]. On le calcule en prenant le ratio du nombre d'erreurs, qui est la somme du nombre de mots insérés I, du nombre de mots substitués S et du nombre de mots omis O, sur le nombre de mots références N.

$$WER = \frac{I + S + O}{N} \tag{7.14}$$

L'alignement des séquences est fait de manière optimale à l'aide d'un algorithme de program-

mation dynamique. On notera que plus l'erreur est basse, meilleur est le système et que l'erreur peut être supérieure à 1 à cause des insertions.

Le taux d'erreur caractère

Le taux d'erreur caractère (ou CER pour *Character Error Rate*) est identique au taux d'erreur mot sauf que l'unité de base est la caractère au lieu d'être le mot. Il permet de pondérer différemment des mots reconnus à quelques lettres près de mots complètement erronés.

Les sacs de mots

Les métriques WER et CER présentent pour point limitant de s'appliquer aux séquences uniquement. Dans une approche de reconnaissance pleine page, les objets hypothèses peuvent être sous-segmentés ou sur-segmentés par rapport aux objets références. De plus, les mots (et similairement les lignes et les paragraphes) peuvent être placés librement sur la page, c'est à dire sans ordre particulier. Pour éviter une étape délicate d'alignement entre les séquences, nous utilisons la métrique dite de sacs de mots (ou BoW pour *Bag of Words*). Cette métrique est particulièrement adaptée à l'évaluation de tâches de reconnaissance pleine page [Pletschacher et al., 2015].

La métrique BoW va considérer l'ensemble des mots références et l'ensemble des mots hypothèses présents dans une page, sans se soucier de l'ordre. On va s'intéresser au nombre C de mots présents dans les deux ensembles, et considérés comme correctement reconnus, au nombre de mots O présents uniquement dans l'ensemble de référence et considérés comme des omissions ainsi qu'au nombre de mots I présents uniquement dans l'ensemble hypothèse et considérés comme des insertions. Les mots présents plusieurs fois dans la page sont placés plusieurs fois dans les ensembles, sans remise. On notera que une substitution (un mot mal reconnu) sera considéré à la fois comme une insertion et comme une omission. On calcule ensuite la précision $C/(C+O)$, le rappel $C/(C+I)$ et la métrique BoW correspond à la F-Mesure qui leur est associée.

7.3 Résultats des différentes stratégies de reconnaissance pleine page.

Dans le chapitre précédent 6, trois techniques de détection d'objets ont été présentées. Dans cette section, nous analysons les performances de ces techniques et nous comparons ces performances à celles de systèmes issus, soit de la détection d'objets dans des scènes naturelles à l'aide

de réseaux de neurones, soit de la détection de lignes de texte dans des documents à partir de techniques de traitement d'images.

La Section 7.3.1 compare les résultats des systèmes sur des critères géométriques. La Section 7.3.2 compare les résultats des systèmes du point de vue de la tâche cible, la reconnaissance de texte. Enfin, la Section 7.3.3 compare visuellement les résultats obtenus par les systèmes proposés.

7.3.1 Evaluation géométrique.

Les résultats de localisation sont donnés sur l'ensemble de test de la base Maurdor restreint aux pages pour lesquelles on est confiant dans le fait d'avoir trouvé et correctement placé l'intégralité des lignes. Nous les donnons pour nos techniques à base de détection de boîtes et à base de détection puis d'appariement de coins mais aussi, pour comparaison, pour des techniques de référence issues de la communauté de l'analyse de document et pour la technique MultiBox développée pour la détection d'objets dans des images naturelles. Ils sont calculés à l'aide de deux métriques se basant sur la géométrie des positions des objets références et hypothèses.

Le Tableau 7.6 utilise une métrique à base de F-Mesure où un objet est considéré comme correctement détecté si sa valeur d'IoU (intersection sur union) est supérieure à un seuil T et le Tableau 7.7 utilise la métrique DetEval [Wolf and Jolion, 2006]. La méthode à base de détection des côtés gauches des lignes de texte, pour laquelle le reconnaisseur de texte apprend à trouver la droite de la ligne, n'est pas incluse dans ces tableaux puisque elle ne donne pas de manière explicite la position des boîtes englobantes.

Pour la métrique à base d'IoU, les résultats sont montrés dans le Tableau 7.6, et différentes valeurs de seuils sont indiquées afin de voir différent niveaux de qualité de localisation. On voit que les systèmes à base de traitements d'images inspirés de Shi et al. [Shi et al., 2009] et de Nicolaou et al. [Nicolaou and Gatos, 2009] ne fonctionnent pas très bien lorsque le seuil T est faible et que l'on s'intéresse donc à la capacité des systèmes à détecter l'ensemble des objets sans se focaliser sur leurs positions exactes. Néanmoins, leurs scores de F-Mesure diminue moins que les techniques à base d'apprentissage automatique lorsque le seuil sur l'IoU augmente et donc lorsque les besoins de précision sont accrus. Cela s'explique en raison de la nature de ces algorithmes qui fonctionnent à partir de binarisations des images d'entrée. Lorsque ces algorithmes donnent une segmentation correcte, la précision des prédictions est bonne puisque les positions sont données au niveau des pixels. Par contre, quand ils sont utilisés en dehors des cas pour lesquels ils ont été perfectionnés, les performances chutent.

A l'inverse, les méthodes à base d'apprentissage automatique et de régression directe des coordonnées comme MultiBox [Erhan et al., 2014] et les méthodes que nous proposons sont plus

TABLE 7.6 – Performance de la détection des boîtes : F-Mesure avec des seuils T sur la valeur d'IoU variables. Résultats donnés sur une restriction de l'ensemble de test de la base Maurdor (616 pages).

Méthode	F-Mesure		
	T=0.3	T=0.5	T=0.7
Shi et al. [Shi et al., 2009]	40.7%	31.1%	21.1%
Nicolaou et al. [Nicolaou and Gatos, 2009]	36.1%	26.3%	15.9%
Multibox [Erhan et al., 2014]	11.3%	2.1%	0.2%
Multibox [Erhan et al., 2014] (optimisée)	48.7%	23.0%	5.2%
Détection de boîtes (P=4), pas de LSTMs	49.9%	23.7%	5.3%
Détection de boîtes (P=4)	73.8%	43.6%	14.1%
Détection de points et appariement (P=2)	69.1%	45.1%	18.2%

robustes et ont un meilleur rappel, au prix d'une plus faible précision. Les méthodes que nous proposons donnent les meilleurs résultats pour des valeurs réalistes de seuils. On remarque également que la méthode à base de détection et d'appariement des coins des boîtes englobantes est meilleure que la technique qui prédit directement les boîtes lorsqu'une valeur élevée est utilisée pour le seuil. Cela confirme les observations de la section 6.1.2, et que la précision des prédictions de coins est meilleure. Par contre, on remarque que cette méthode est moins bonne pour de petites valeurs d'IoU, par exemple 0.3. Cela est dû au fait que les lignes non détectés par les deux réseaux s'ajoutent et par le fait qu'il puisse y avoir des erreurs d'appariement.

La métrique DetEval utilisée dans le Tableau 7.7 confirme les résultats de la métrique à base d'intersection sur union. Les résultats, pour ces deux métriques, sont indiqués avec et sans couches de récurrences sous forme de 2D-LSTMs pour le système qui détecte les boîtes. L'amélioration très significative des résultats lorsque les couches 2D-LSTMs sont présentes souligne l'importance de celles-ci afin de transmettre l'information de contexte.

Multibox [Erhan et al., 2014] est fortement moins performante que nos méthodes, même après optimisation des hyper-paramètres du réseau sur l'ensemble de validation. Cela est probablement lié au fait que la couche de sortie ne soit pas partagée et que le modèle doive donc apprendre des choses similaires aux différents endroits de l'image. Nous ne sommes pas parvenus, malgré d'intensifs efforts, à appliquer YOLO [Redmon et al., 2016] à ce problème, même après avoir optimisé l'architecture sur notre tâche [Moysset et al., 2018]. Si ce système donne de bons résultats sur la tâche de détection d'objets dans des scènes naturelles pour laquelle il a été développé, cela ne marche pas lorsqu'un nombre important d'objets sont présents dans les images, cadre dans lequel notre tâche se situe, pour les raisons indiquées en section 4.3.1 et illustrés dans l'image 4.4.

TABLE 7.7 – Performance de la détection des boîtes avec la métrique DetEval [Wolf and Jolion, 2006]. Résultats donnés sur une restriction de l'ensemble de test de la base Maurdor (616 pages).

Méthode	Rappel	Précision	F-Mesure
Shi et al. [Shi et al., 2009]	35.1%	38.4%	36.7%
Nicolaou et al. [Nicolaou and Gatos, 2009]	46.7%	39.6%	42.9%
Multibox [Erhan et al., 2014]	4.2%	10.0%	6.0%
Multibox [Erhan et al., 2014] (optimisée)	28.8%	52.3%	31.1%
Détection de boîtes (P=4), pas de LSTMs	28.6%	52.4%	31.1%
Détection de boîtes (P=4)	51.2%	61.4%	55.9%
Détection de points et appariement (P=2)	54.2%	58.6%	56.3%

7.3.2 Taux de reconnaissance pleine page.

Les résultats des reconnaissances pleine page du texte à partir des prédictions issues de nos différents systèmes sont indiquées dans le Tableau 7.8. La métrique *Bag of Words* (BoW), qui est une F-Mesure ou un mot hypothèse est considéré comme juste si il est présent parmi l'ensemble des mots de la page référence, est utilisée. Les résultats sont donnés respectivement pour l'ensemble des pages complètement en français et pour les pages complètement en anglais de l'ensemble de test de la base Maurdor afin d'éviter de devoir ajouter une étape de détection de la langue. Comme précédemment, les résultats sont comparés avec des systèmes de référence.

On voit que les méthodes proposées ont de bons résultats, à la fois sur les ensembles en français et en anglais. Les stratégies basées sur la détection de boîtes (P=4) et sur la détection et l'appariement des coins (P=2) ont des scores de F-Mesure similaires de plus de 70%. Ces performances similaires confirment que les erreurs dues aux mauvais positionnements des boîtes sont compensées par les erreurs liées à l'addition des erreurs entre les détecteurs de coins et leur appariement. La technique qui détecte les côtés gauches, quant à elle, obtient des scores de F-Mesure valant jusque presque 80%, obtenant de manière nette les meilleurs performances. Les résultats, pour nos trois stratégies, sont meilleurs que les systèmes de référence. Cela s'explique par le fait que le rappel général de notre méthode soit bon.

7.3.3 Exemples de résultats.

Les figures 7.4 à 7.10 montrent, visuellement, les résultats obtenus par nos trois méthodes proposées sur un jeu d'images de l'ensemble de validation.

TABLE 7.8 – Performance conjointe de la détection et de la reconnaissance : F-Mesure sur la reconnaissance de mots (BOW) sur l'ensemble des pages entièrement en français et entièrement en anglais de l'ensemble de test de la base Maurdor.

Méthode	Français (507 pages)	Anglais (265 pages)
Shi et al. [Shi et al., 2009]	48.6%	30.4%
Nicolaou et al. [Nicolaou and Gatos, 2009]	65.3 %	50.0%
Multibox [Erhan et al., 2014]	27.2%	14.8%
Multibox [Erhan et al., 2014] (optimisée)	32.4%	36.2%
Détection de boîtes (P=4), pas de LSTMs	57.8%	56.9%
Détection de boîtes (P=4)	71.2%	71.1%
Détection de points et appariement (P=2)	71.7%	72.3%
Détection de côtés gauches, puis reconnaissance (P=3)	79.9%	79.1%

La puissance des couches 2D-LSTMs peut être visualisée dans ces images. En particulier, les Figures 7.4 et 7.8 montrent que le système détectant les boîtes est capable de détecter des objets plus larges que les champs réceptifs des prédicteurs de boîtes. Cela est rendu possible grâce aux informations de contexte propagées par les couches LSTMs.

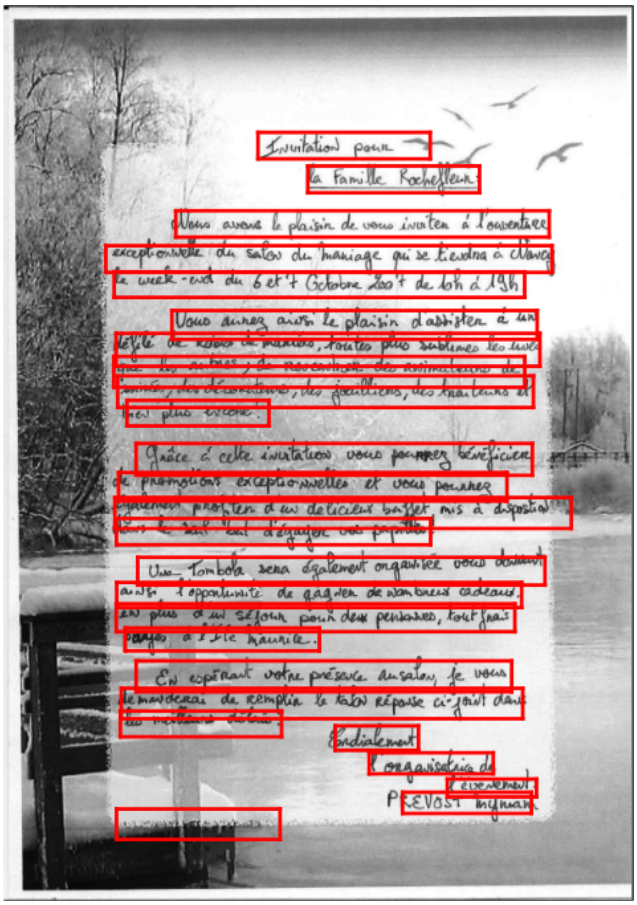
Les Figures 7.5 et 7.9 montrent que les systèmes sont capables de détecter et de localiser un nombre important de petits objets.

Les Figures montrent les résultats des systèmes prédisant les côtés gauches des lignes, des boîtes détectées directement, et des boîtes détectées à partir d'appariement des coins gauches et droits. On observe que, et particulièrement pour les images dans lesquelles de petits objets doivent être détectés, comme dans les Figures 7.6 ou 7.9, la précision des positions prédites est meilleure avec les systèmes détectant les coins ou les côtés gauches. On voit également que le système détectant les coins souffre de quelques erreurs d'appariement. De bons résultats sont obtenus sur un ensemble hétérogène de documents incluant des formulaires, des lettres, des tracts, des cartes ou des articles de journaux ; à la fois pour des documents en français, en arabe et en anglais, et à la fois pour des textes imprimés et manuscrits.

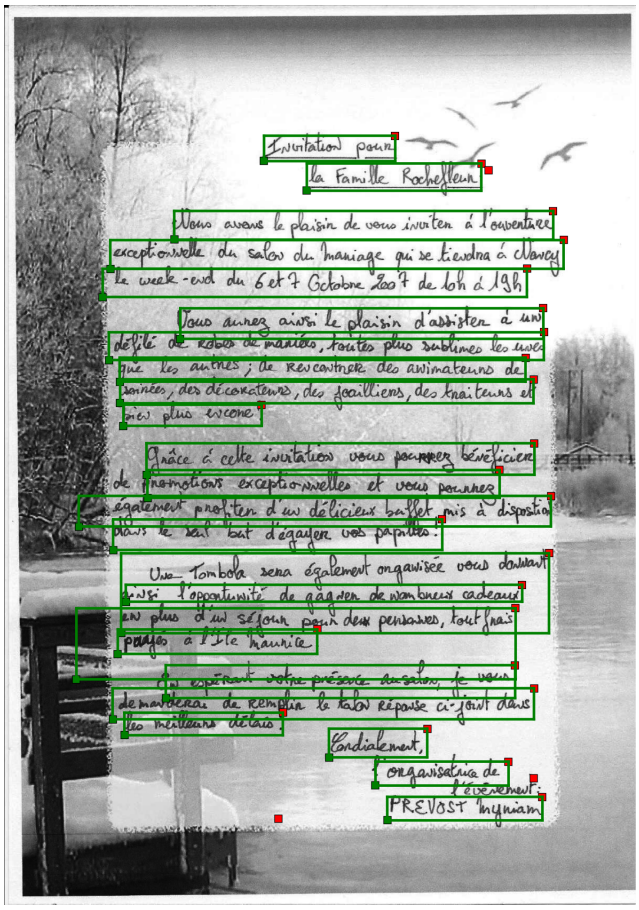
La Figure 7.11 montre des exemples de résultats de reconnaissance pleine page obtenus après une détection de côtés gauches des lignes de texte. On observe que la reconnaissance est correctement effectuée, même lorsque d'autres lignes sont présentes à droite du texte.

Fig. 7.4 – Illustration d'exemples de résultats de détections.

(a) Détection de boîtes



(b) Détection de coins et appariement



(c) Détection de côtés gauches

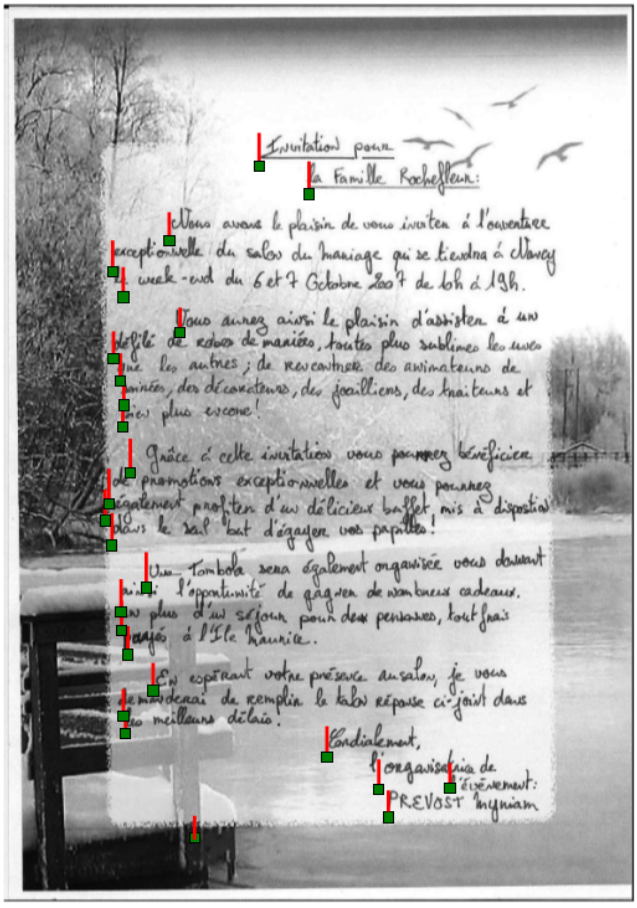
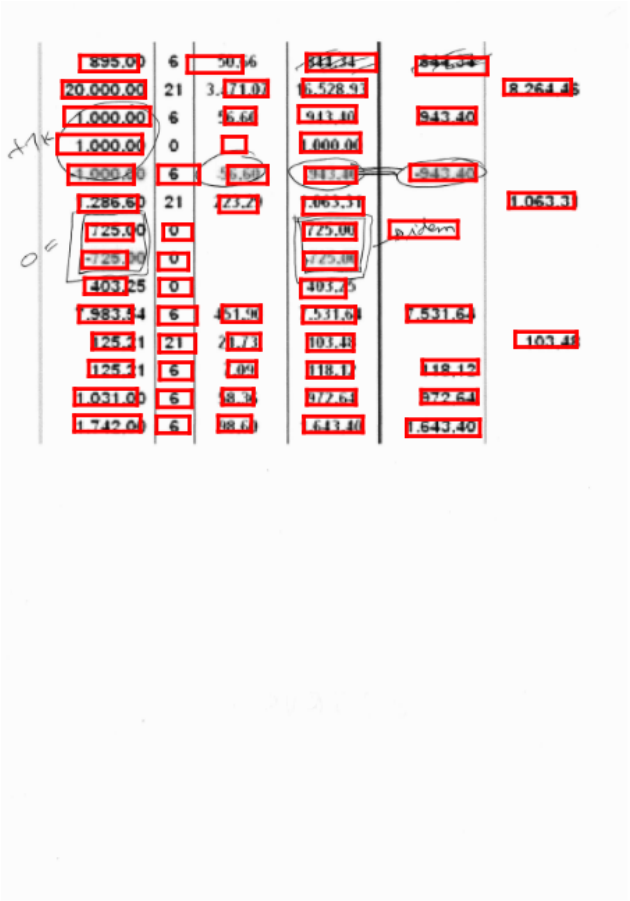
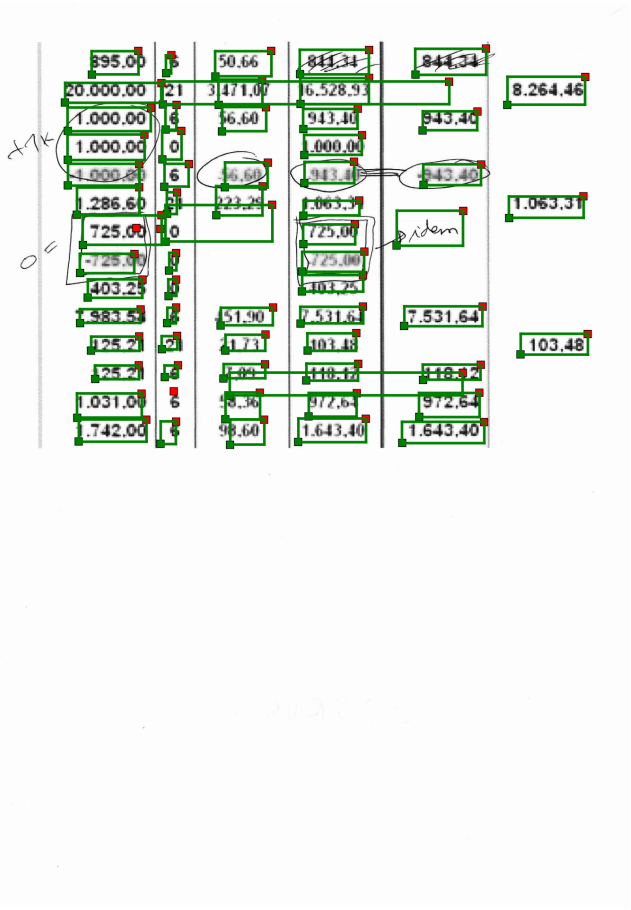


Fig. 7.5 – Illustration d'exemples de résultats de détections.

(a) Détection de boîtes



(b) Détection de coins et appariement



(c) Détection de côtés gauches

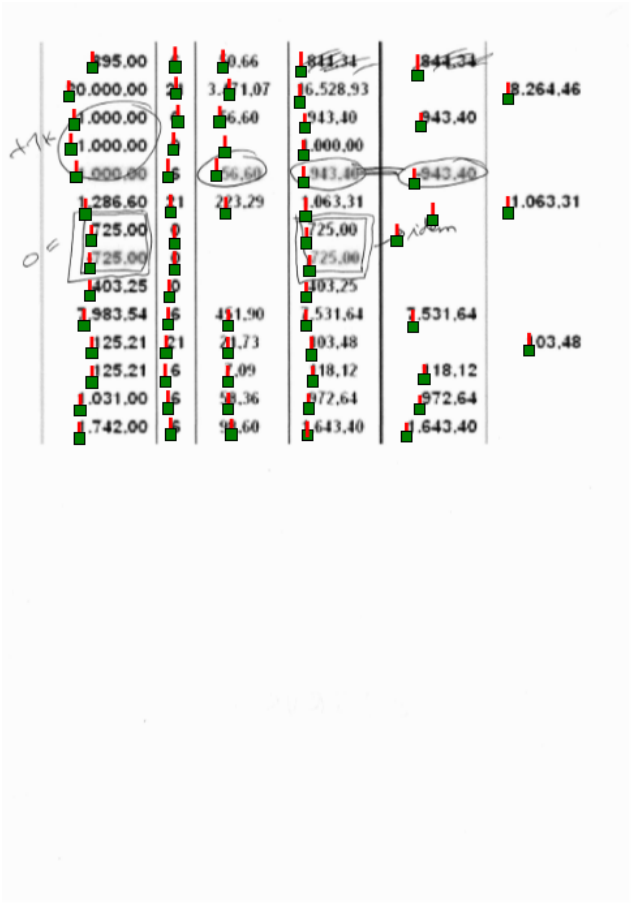
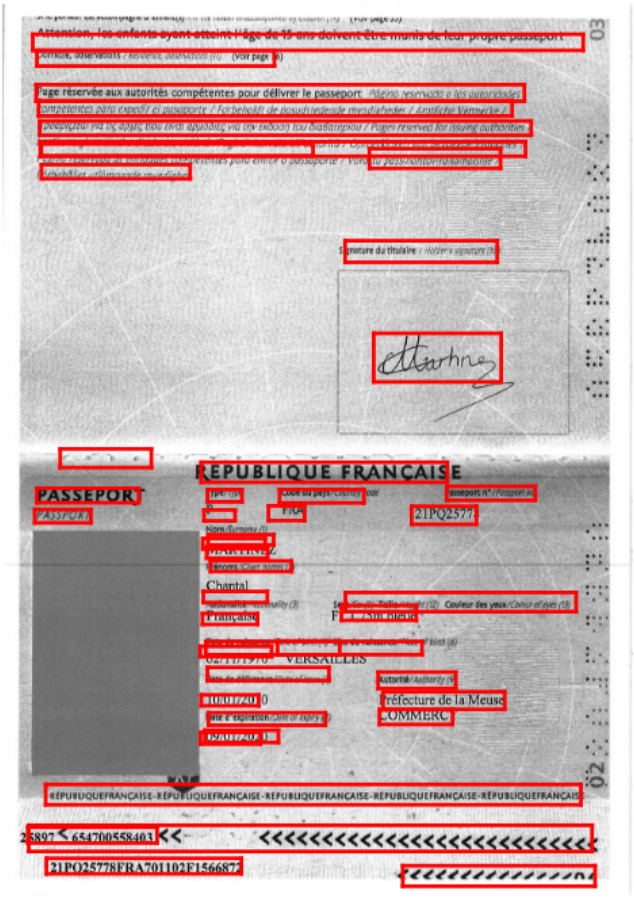
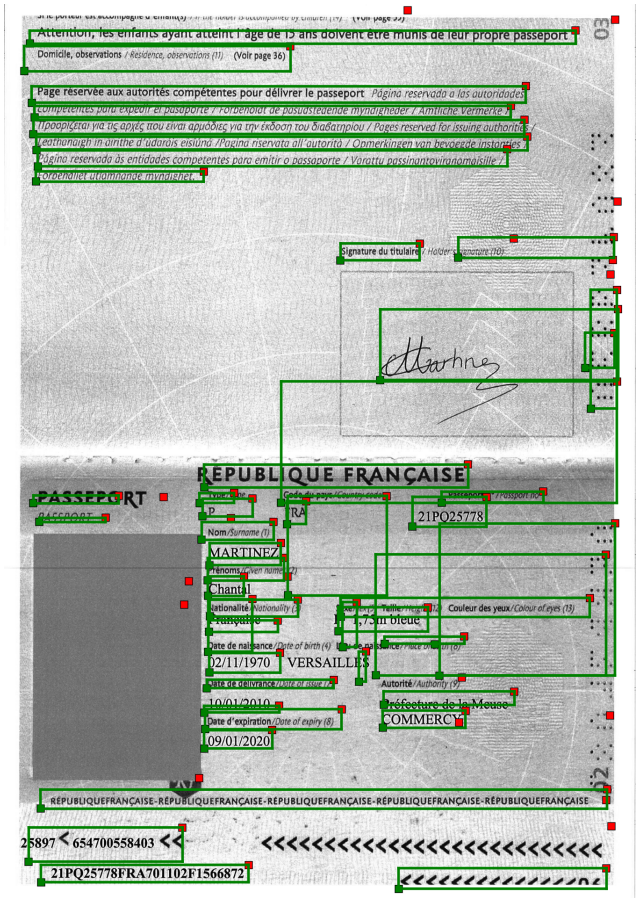


Fig. 7.6 – Illustration d'exemples de résultats de détections.

(a) Détection de boîtes



(b) Détection de coins et appariement



(c) Détection de côtés gauches

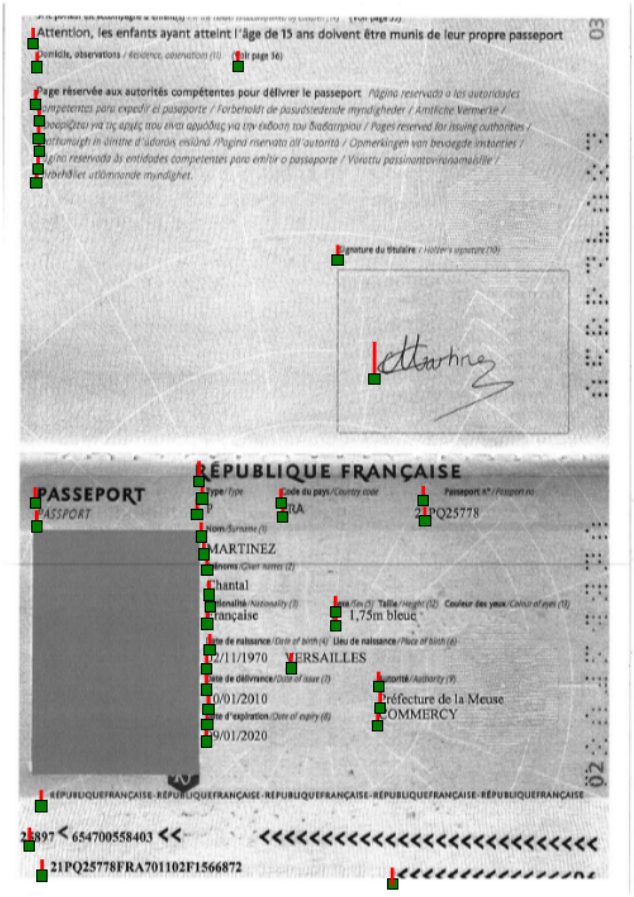


Fig. 7.7 – Illustration d'exemples de résultats de détections.

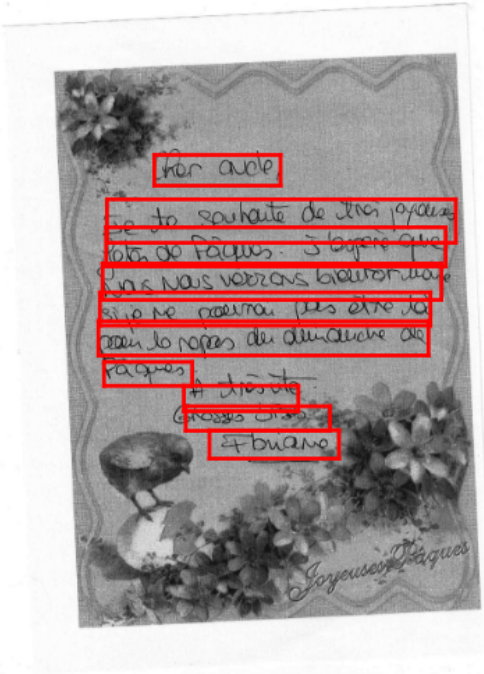
(a) Détection de boîtes

(b) Détection de coins et appariement

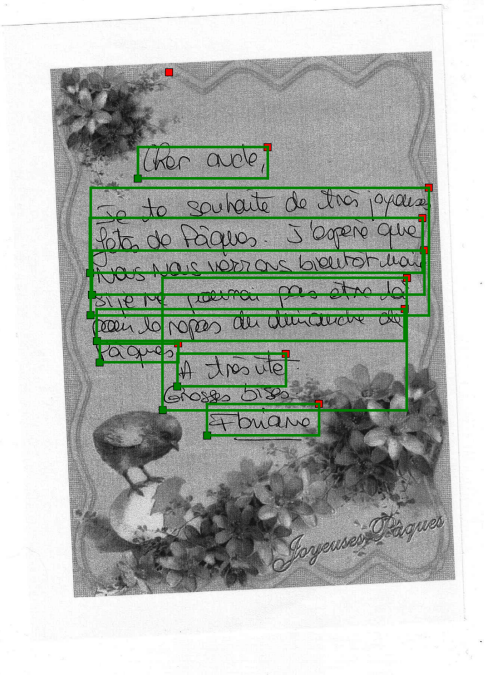
(c) Détection de côtés gauches

Fig. 7.8 – Illustration d'exemples de résultats de détections.

(a) Détection de boîtes



(b) Détection de coins et appariement



(c) Détection de côtés gauches

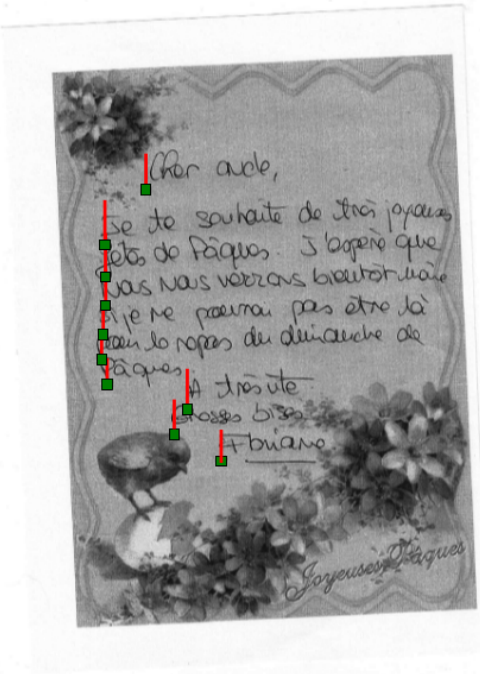
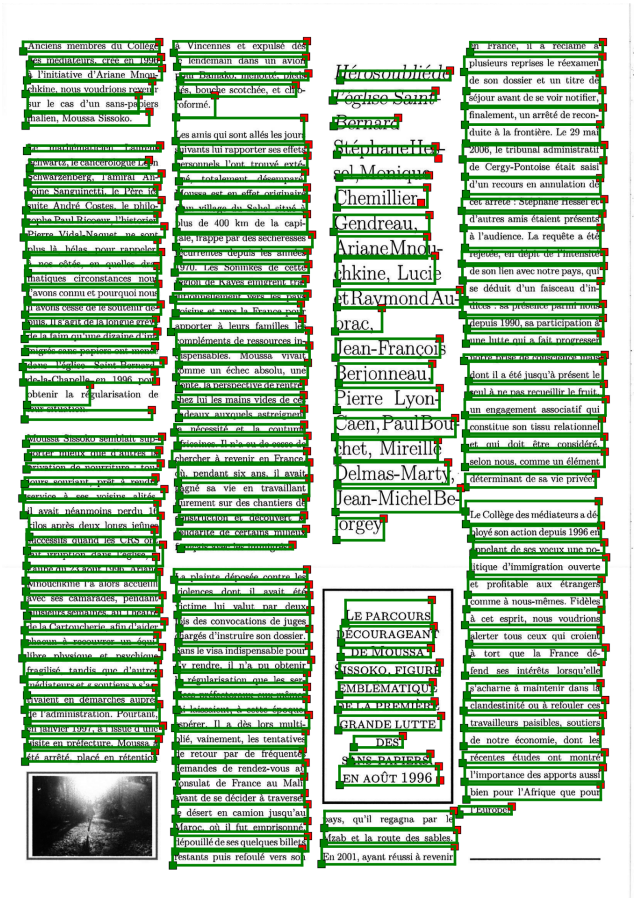


Fig. 7.9 – Illustration d'exemples de résultats de détections.

(a) Détection de boîtes

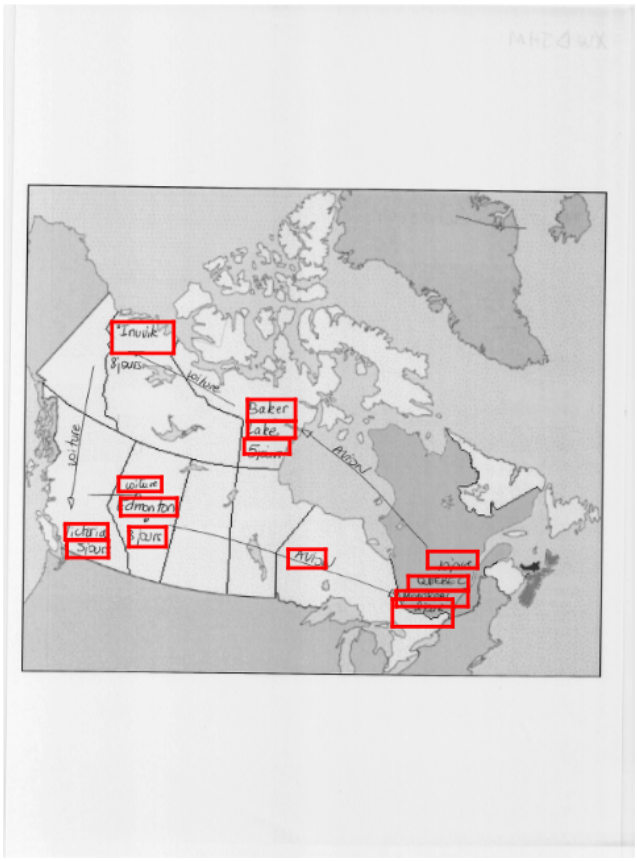
(b) Détection de coins et appariement



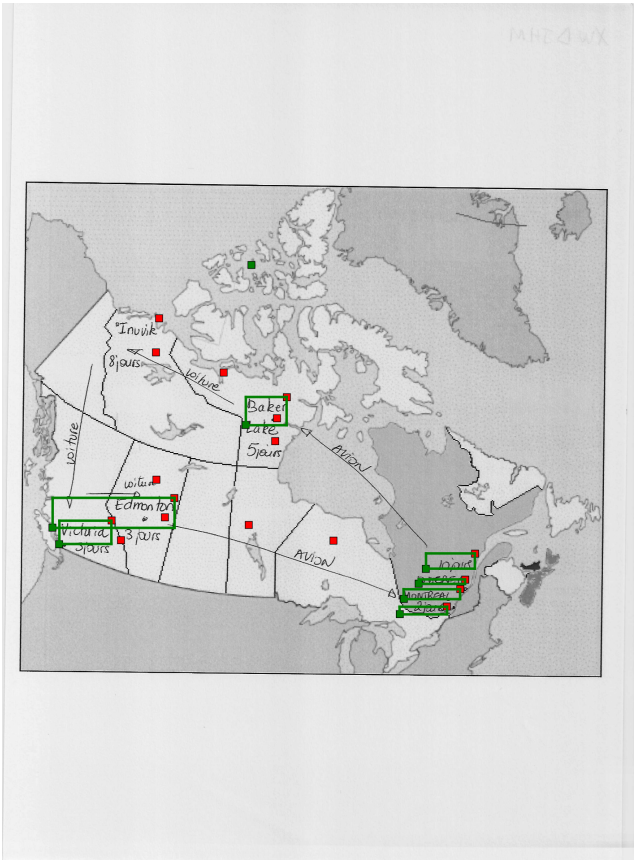
(c) Détection de côtés gauches

Fig. 7.10 – Illustration d'exemples de résultats de détections.

(a) Détection de boîtes



(b) Détection de coins et appariement



(c) Détection de côtés gauches

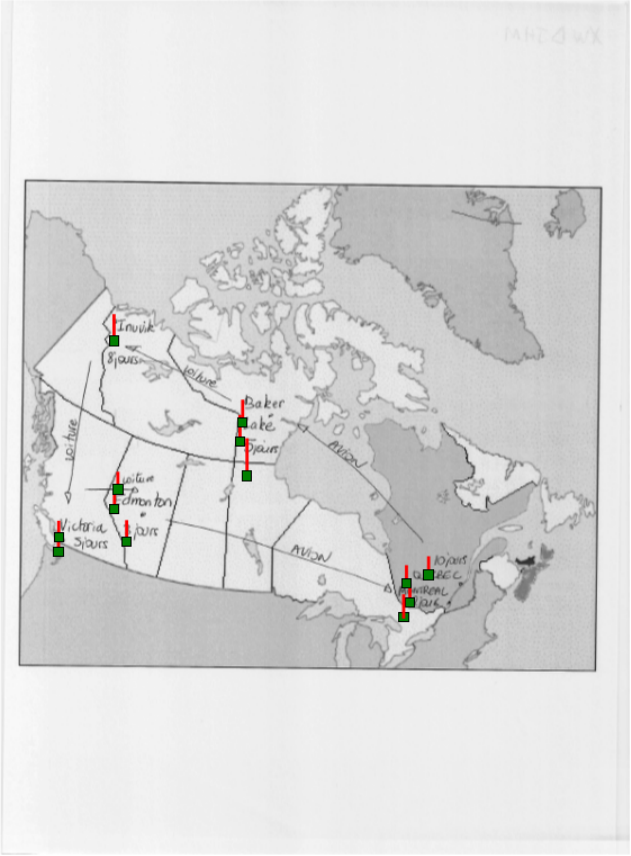
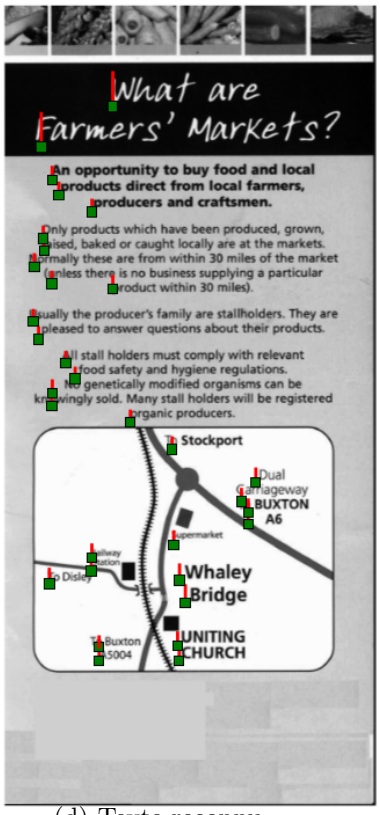


Fig. 7.11 – Illustration de reconnaissance de texte pleine page pour notre système basé sur la détection des côtés gauches (P=3) et la reconnaissance ayant appris à trouver la position de la droite des lignes.

(a) Image d'entrée et résultats de localisation

(b) Image d'entrée et localisation



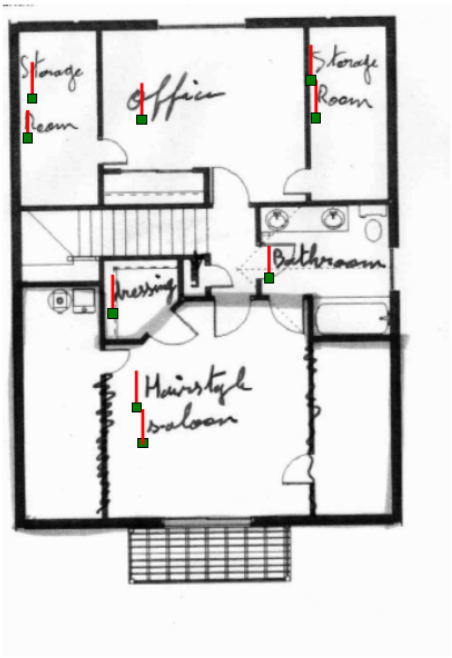
(c) Texte reconnu

- Invitation pour mon∞
- la Famille Rochefleur :∞
- Nous avons le plaisir de vous inviter à l'ouverture∞
- exceptionnelle du salon du mariage qui se tiendra à chargey∞
- le verte-end du 6 et 7 Octobre 2007 de 20h à 19h.∞
- Vous aurez ainsi le plaisir d'assister à un∞
- défilé de robes de mariées, toutes plus oublimes les unes.∞
- que les autres ; de rencontrer des animateurs de∞
- sinées, des décoracteurs, des joailliers, des traiteuns et :∞
- bien plus encore !∞
- Grâce à cette invitation vous pourrez bénéficier de :∞
- de prenmations exceptionnelles et vous pourrez∞
- également profiter d'un délicieux buffet, mis à disposition !∞
- dans le sait bout d'égaler vos papilles !∞
- Une Tombola sera également organisée vous donnant.∞
- ainsi l'opportunité de gagner de nombreux cadeaux.∞
- En plus d'un séjour pour deux personnes, tout frais∞
- passés à l'Ile haunité.∞
- En espérant votre présence au salon, je vous∞
- demanderai de remplir le talon réponse ci-joint dans∞
- les meilleurs délais !∞
- Cordialement,∞
- l'organisatrice de∞
- LEU******∞
- PREVOST Myriam∞
- *******∞

(d) Texte reconnu

- DY∞
- ****4481991991∞
- An opportunity to buy food and local∞
- products direct from local farmers,∞
- 1cts direct from local farmers,∞
- Only products which have been produced, grown,∞
- raised, baked or caught locally are at the markets.∞
- formally these are from within 30 miles of the market∞
- (unless there is no business supplying a particular∞
- product within 30 miles).∞
- Usually the producer's family are stallholders. They are∞
- pleased to answer questions about their products.∞
- All staall holders must comply with relevant∞
- food safety and hygiene regulations.∞
- No genetically modified organisms can be∞
- knowingly sold. Many staall holders will be registered∞
- organic producers.∞
- To Stockport∞
- Dual∞
- Carriageway∞
- to BUXTON∞
- A6∞
- supermarket∞
- Mailway∞
- ∞
- To Disien-∞
- Whalay∞
- Bridge∞
- To Burton∞
- A5004∞
- UNITTING∞
- CHURCH∞

(e) Image d'entrée et résultats de localisation



(f) Texte reconnu

- tonage∞
- Room∞
- offi***∞
- Storage∞
- Room∞
- Faithroom∞
- Merrill∞
- Mainstake∞
- Saloon∞

7.4 Etudes ablatives

Après avoir illustré en Section 7.3 la performance de nos systèmes, nous cherchons dans cette section à justifier l'apport de nos contributions principales.

7.4.1 Les couches récurrentes 2D-LSTMs

Comme notre modèle prédit la position des objets de manière locale, nous avons utilisé des couches de 2D-LSTMs dans notre réseau afin de transmettre les informations contextuelles. Ces informations contextuelles peuvent être la position des bords des objets plus grands que les champs réceptifs convolutionnels mais aussi des informations relatives à la structure de la page. Ces couches de LSTMs multi-dimensionnelles et la manière dont nous les utilisons sont décrites en Section 5.4. Nous allons justifier, expérimentalement, l'importance d'utiliser plusieurs de ces couches de récurrence dans notre réseau et nous justifions leur placement dans notre architecture. Nous

TABLE 7.9 – Erreur de localisation des points inférieurs gauches des boîtes englobantes en fonction du nombre et de la position des couches 2D-LSTMs sur l'ensemble de validation de la base Maurdor.

Position des 2D-LSTMs	Erreur = 1 - F-Mesure
Pas de LSTMs	31.2%
1	20.1%
1-2	14.9%
1-2-3	11.3%
1-2-3-4	9.1%
1-2-3-4-5	9.7%
2-3-4-5	10.9%
3-4-5	12.1%
4-5	16.4%
5	21.2%

mettons en lumière l'importance d'initialiser avec une valeur négative les biais des portes d'oublis de ces couches 2D-LSTMs et nous justifions la non utilisation de *peephole* dans nos cellules LSTMs.

Nombre et position des 2D-LSTMs

Dans cette section, nous étudions l'influence du nombre de couches 2D-LSTMs utilisées ainsi que celle de la position dans le réseau où elles sont placées. Les résultats sont donnés dans le Tableau 7.9 pour la localisation des points inférieurs gauches des boîtes englobantes. Ils sont donnés en fonction du numéro des couches convolutionnelles après lesquelles les couches récurrentes sont insérées.

On remarque qu'ajouter des couches de 2D-LSTMs permet de gagner en performance. Cela est vrai même si il s'agit ici de détecter des points et que ceux ci sont donc forcément inclus dans les champs réceptifs auxquels ils sont associés. C'est donc bien l'ajout du contexte qui permet d'améliorer les performances. On notera cependant que cela cause une augmentation du nombre d'époques nécessaires à l'entraînement. On observe que la récurrence sur la dernière couche n'apporte pas d'amélioration lorsque des récurrences sont présentes sur les couches précédentes. Nous choisissons donc d'utiliser des récurrences uniquement après les 4 premières couches convolutionnelles, comme mentionné dans le Tableau 5.3.

TABLE 7.10 – Localisation des boîtes englobantes en fonction de la valeur de biais utilisée pour initialiser les portes d'oubli des couches 2D-LSTMs sur l'ensemble de validation de la base Maurdor.

Biais pour l'initialisation	métrique ZoneMap
1	100%
0	100%
-1	62.2%
-10	56.5%

Initialisation des biais sur les portes d'oubli des 2D-LSTMs

Dans ces couches de contexte 2D-LSTMs, il s'est avéré important d'initialiser les biais sur les portes d'oubli. Le Tableau 7.10 compare les résultats sur la détection de boîtes de systèmes initialisés avec des biais différents sur lesdites portes. Initialiser les biais avec une valeur élevée positive signifie qu'au début de l'entraînement la porte restera ouverte et que les états internes des positions précédentes seront transmis. A l'inverse, initialiser avec un biais négatif élevé signifie que la porte sera fermée et que les états internes ne seront pas transmis et donc oubliés.

On observe qu'initialiser les réseaux avec des biais nuls ou positifs empêche les réseaux de converger. Cette convergence est permise par les biais négatifs. Cela signifie qu'oublier les états internes et donc la mémoire spatiale véhiculée par ceux-ci est préférable en début d'entraînement. Cela s'explique en raison du fait que cela simplifie le problème et facilite l'entraînement des couches convolutionnelles en début d'entraînement. Les couches de récurrence ajoutent ensuite les informations contextuelles au fur et à mesure que les biais sont réduits au cours de l'entraînement. Comme une fonction sigmoïde est appliquée après l'ajout de ces biais, une valeur de -1 laisse la porte entrouverte alors qu'elle est beaucoup plus fermement fermée avec un biais de -10. On remarque qu'une valeur de -1 pour les biais est préférable à une valeur de -10. Cela rend aussi la convergence plus rapide, ce qui s'explique par le fait qu'il est plus aisé de quitter le plateau de la sigmoïde dans le premier cas.

Peephole dans la LSTM

De manière à conserver l'information temporelle, et à améliorer la précision des prédictions temporelles, Gers et al. [Gers et al., 2002] propose d'utiliser du *peephole* dans les LSTMs. Ce principe consiste à ajouter les états internes de la LSTM, les mémoires, comme caractéristiques d'entrée des calculs des portes de la LSTM, en plus des sorties aux positions précédentes et des caractéristiques de la couche précédente à la position actuelle utilisées dans la LSTM simple.

TABLE 7.11 – Erreur de localisation des points inférieurs gauches des boîtes englobantes en fonction de l'utilisation ou non de *peephole* dans nos cellules 2D-LSTMs sur l'ensemble de validation de la base Maurdor.

Peephole	1 - F-Mesure
Avec	19.3%
Sans	18.8%

Ce principe peut être généralisé en deux dimensions. Nous l'avons essayé dans l'espoir que cela permette d'améliorer la précision de nos prédictions spatiales.

Comme illustré dans le Tableau 7.11, très peu de changements sont apportés par cette méthode, aussi bien en terme de performances, y compris de localisation, qu'en terme de vitesse d'apprentissage. Retirer ce *peephole* de nos couches récurrentes nous a permis de réduire le nombre de paramètres libres utilisés dans notre réseau.

7.4.2 Prédictions locales et couche de sortie

Après avoir justifié l'importance des couches de récurrence dans nos modèles, nous nous intéressons dans cette section à l'importance des autres principaux choix que nous avons effectués au niveau de l'architecture de nos réseaux. En particulier la couche de sortie locale décrite en Section 5.3 et dont l'utilité a été démontrée par comparaison avec MultiBox [Erhan et al., 2014] en Section 7.3. Nous étudions l'influence du nombre de sorties de notre réseau et évaluons l'utilisation de biais sur cette couche de sortie.

Nombre de sorties par position

Dans cette section, nous évaluons l'influence du nombre d'objets hypothèses prédits pour chaque localisation sur la prédiction des coins inférieurs gauches des boîtes englobantes. Prédire plus d'objets hypothèses permet d'être en mesure de prédire un nombre maximal d'objets plus important et permet aux sorties de se spécialiser sur des types et des formes d'objets différents. A l'inverse, prédire moins d'objets facilite l'entraînement en limitant le nombre de paramètres sur la dernière couche de notre réseau et permet aux sorties d'être plus souvent utilisées et donc de recevoir plus de gradients afin d'apprendre à prédire la position des objets. On observe dans le Tableau 7.12 que prédire 10 objets pour chaque localisation semble un bon compromis.

TABLE 7.12 – Erreur de localisation des coins inférieurs gauches des boîtes englobantes en fonction du nombre de boîtes hypothèses prédites pour chaque localisation sur l'ensemble de validation de la base Maurdor.

Nombre d'objets prédits	1 - F-Mesure
5	13.4%
10	13.2%
20	14.6%

TABLE 7.13 – Localisation des boîtes englobantes en fonction de l'initialisation des biais de position sur la couche de sortie de notre réseau sur l'ensemble de validation de la base Maurdor.

Biais de position	ZoneMap
Initialisés à zéro	50.7%
Initialisés arbitrairement	50.9%

Initialisation des biais sur la sortie

Comme mentionné précédemment, utiliser plus de sorties qu'il n'y a d'objets permet à ces sorties de se spécialiser vis à vis des formes des objets. Par exemple, certaines sorties auront tendance à prédire des objets longs et à droite de l'image alors que d'autres prédiront des objets courts et à gauche de l'image. Afin de faciliter cette répartition, nous initialisons de manière arbitraire les biais sur la couche de sortie qui prédit les positions de manière à couvrir l'espace des prédictions. Comme illustré dans le Tableau 7.13, les performances, comparées à une initialisation de ces biais à zéro sont sensiblement les mêmes. Cependant, la convergence est accélérée par cette initialisation.

7.4.3 Fonction de coût

Dans cette section, nous évaluons les choix faits au niveau de la fonction de coût qui est utilisée pour optimiser notre réseau et que nous avons détaillée précédemment en Section 4.2. En particulier nous justifions le fait d'utiliser un coefficient de pondération entre nos coûts de confiance et de position plus élevé lors de l'appariement et nous justifions le fait de ne pas utiliser les ancres présentées dans MultiBox.

Quelques remarques supplémentaires sur l'apprentissage, nous constatons, sans surprise,

TABLE 7.14 – Localisation des boîtes englobantes en fonction de la valeur du coefficient de pondération des coûts α_{match} sur l'ensemble de validation de la base Maurdor pour $\alpha = 1$.

α_{match}	ZoneMap
3	75.6%
10	61.9%
100	43.3%
300	44.3%
1000	51.5%
3000	54.0%
10000	53.9%

qu'ajouter du *Dropout* permet d'améliorer les performances du réseau de manière très significative. Cela s'explique par les propriétés régularisantes du *Dropout* qui sont particulièrement utiles dans un cas comme le notre où la taille de l'ensemble d'entraînement est réduite. Nous mentionnons également qu'utiliser RmsProp comme fonction d'optimisation s'est avéré nettement meilleur que d'utiliser une descente de gradient stochastique simple. Cela pourrait s'expliquer par le fait qu'une fonction d'optimisation comme RmsProp peut, en faisant varier les taux d'apprentissages séparément, s'adapter à l'évolution des dynamiques entre les coûts de confiance et de localisation.

Valeur des coefficients de pondération des coûts

La fonction de coût que nous utilisons pour entraîner notre réseau, décrite en Section 4.2, et en particulier dans l'équation 4.11, est la composition d'un coût de confiance et d'un coût de position. Le paramètre α qui pondère ces deux coûts est un paramètre important. Ces coûts de confiance et de position sont également pondérés par un paramètre α_{match} dans le calcul de la distance utilisé pour l'appariement des boîtes références et hypothèses. Utiliser un paramètre α_{match} grand signifie que l'appariement sera fait principalement sur des critères de localisation alors qu'utiliser un alpha petit signifie qu'on favorisera la confiance du réseau dans le fait que ces objets existent.

On observe dans le tableau 7.14 qu'utiliser un α_{match} plus grand que α améliore les performances du système. Cela s'explique par le fait qu'utiliser un α_{match} plus petit va avoir tendance à associer toujours les mêmes sorties du réseau, et ce dès le début de l'entraînement, empêchant la diversification des sorties. Utiliser un α_{match} plus élevé permet à toutes les sorties d'être utilisées et donc au réseau de maximiser son potentiel.

TABLE 7.15 – Erreur de localisation des boîtes englobantes en fonction de l'utilisation d'ancres générées par K-moyennes, sur l'ensemble de validation de la base Maurdor

Ancres	ZoneMap
Avec	49.0%
Sans	43.1%

Utilisation des ancres introduites dans MultiBox

Dans MultiBox [Erhan et al., 2014], les objets hypothèses sont appariés à des ancres, calculées par partitionnement en K-moyennes, plutôt que directement aux objets hypothèses. Cela a pour but de spécialiser les sorties à détecter des objets présents à une certaine position. Nous avons constaté une diminution des performances de notre système lorsque ces ancres sont utilisées, comme illustré dans le Tableau 7.15 et avons donc choisi de ne pas les utiliser.

L'inutilité de ces ancres peut s'expliquer de plusieurs manières. Tout d'abord, nos sorties sont déjà associées, par construction et en raison du caractère local de nos prédictions, à des positions différentes dans l'image. De plus, nous avons plus d'objets à détecter que dans les tâches de détection d'objets dans des images naturelles, ce qui force plus de sorties différentes à être utilisées. Enfin, comme expliqué dans la partie précédente, nous utilisons un α_{match} grand pour forcer les sorties à être utilisées, les deux peuvent être redondants et finissent par ajouter des contraintes non nécessaires.

7.4.4 Ensembles d'entraînement

Spécialisation des entraînements

Les objets présents dans nos images, sont de natures différentes puisqu'ils peuvent être manuscrits ou imprimés, et dans plusieurs langues différentes comme cela est illustré par le Tableau 7.1. Ils peuvent également appartenir à des catégories de documents différentes décrites dans le tableau 7.2.

Nous avons cherché à comparer la performance de systèmes entraînés à tout détecter simultanément ou entraînés à ne détecter qu'une seule de ces catégories. L'avantage d'un réseau spécialisé est qu'il est entraîné à voir des objets plus homogènes alors que le réseau généraliste aura vu plus de données afin de généraliser.

Dans le tableau 7.16, nous comparons les résultats des systèmes spécialisés sur le script et le

TABLE 7.16 – Erreur de localisation des coins inférieurs gauches des boîtes englobantes pour des réseaux spécialisés sur une langue et un type sur l'ensemble de validation de la base Maurdor.

Sous-ensemble	1-F-Mesure réseau spécialisé	1-F-Mesure réseau générique
Latin manuscrit	36.7%	17.4%
Latin imprimé	12.7%	11.6%
Arabe manuscrit	58.6%	24.7 %
Arabe imprimé	35.0%	23.2%

TABLE 7.17 – Erreur de localisation des coins inférieurs gauches des boîtes englobantes pour des réseaux spécialisés sur une catégorie de documents sur l'ensemble de validation de la base Maurdor.

Catégorie	1-F-Mesure réseau spécialisé	1-F-Mesure réseau générique
C1	16.0%	16.2%
C2	18.2%	16.1%
C3	13.3%	10.1%
C4	12.8%	12.6%

type d'écriture ; le français et l'anglais ayant été regroupés car très similaires. On observe que les résultats sont particulièrement dégradés pour les ensembles avec peu de données d'entraînement.

Dans le tableau 7.17 nous avons comparé la spécialisation par rapport aux catégories. Nous remarquons une dégradation des performances par rapport au système générique. Voir des données différentes permet donc au réseau de s'améliorer.

7.4.5 Influence de la taille de l'ensemble d'entraînement

Nous avons aussi étudié l'importance du nombre de documents disponibles vis à vis de notre système. Plusieurs entraînements ont été effectués en sélectionnant aléatoirement une partie de notre ensemble d'entraînement. Les résultats sont illustrés dans le Tableau 7.18. Sans surprise, plus il y a de données meilleur est notre système.

TABLE 7.18 – Erreur de localisation des boîtes englobantes pour des réseaux entraînés avec un nombre variable de documents sur l'ensemble de validation de la base Maurdor.

Nombre de pages d'entraînement	ZoneMap
1000	68.9%
2000	67.3%
3000	53.2%
3995	50.7%

7.4.6 Résultats négatifs

Couleur

Des images de la base Maurdor sont en couleurs, d'autres en niveaux de gris et d'autres en binaire. Utiliser les images en couleurs (et les convertir dans cette représentation le cas échéant) permet de garder toute l'information à notre disposition dans l'image. Néanmoins utiliser des caractéristiques couleurs ne permet pas de gain en performance. Cela est probablement dû au fait que, pour l'essentiel des images en couleurs, leur transformation en niveaux de gris ne rend pas le texte illisible. Comme le temps de calcul est principalement le fait des premières couches de convolutions et qu'utiliser des images couleurs multiplie par trois le nombre d'opérations à réaliser dans la première couche de convolution, nous avons choisi de travailler uniquement avec des images en niveaux de gris.

Coordinate maps

Toujours dans l'optique d'aider notre réseau à améliorer la précision de ses prédictions spatiales, nous avons essayé d'utiliser la technique de *Coordinate maps* [Liang et al., 2015]. Cela consiste à ajouter comme caractéristiques en entrée d'une couche la position, spatiale, dans laquelle on se trouve dans la carte de caractéristiques. L'idée est d'aider le réseau à pouvoir prédire des choses différentes, pour la même image en entrée, à des endroits différents de l'image. Cela n'a pas non plus permis d'améliorer les prédictions des positions. Cela s'explique probablement par la présence de couches récurrentes dans notre réseau qui permettent déjà de transmettre l'information de la position dans l'image si celle-ci est pertinente.

7.5 Temps de calcul

Les couches récurrentes de type 2D-LSTM ont pour particularité d'utiliser les informations contextuelles venant à la fois des sorties d'un élément adjacent horizontalement dans la carte de caractéristiques et d'un élément adjacent verticalement, comme cela a été décrit dans la Section 5.4 qui les détaille, et en particulier dans la Figure 5.3. Cela complique fortement la parallélisation des calculs puisque il faut attendre d'avoir calculer les résultats voisins avant de faire les calculs à une position donnée et, de fait, l'utilisation de GPU est rendue inadaptée à cette tâche.

Durant la majorité de ce travail, les entraînements ont été effectués sur CPU. En effet, les récurrences de type 2D-LSTM n'ont été introduits pour le GPU qu'à partir des travaux de Doetsch et al. [Doetsch et al., 2016], inspirés de Stollenga et al. [Stollenga et al., 2015]. L'idée est d'effectuer les calculs selon les diagonales des cartes de caractéristiques, qui peuvent être calculées parallèlement. Cela permet au calcul de se faire dans un temps asymptotiquement proportionnel à la dimension la plus grande de l'image. Néanmoins, lorsque la séquence est longue, la parallélisation reste inefficace. C'est en particulier le cas au niveau des premières couches de notre réseau où les tailles des cartes de caractéristiques restent grandes.

Les temps de calcul pour chaque couche de notre réseau sont indiqués dans le tableau 7.19, en utilisant un CPU Intel® Xeon® Processor E5-2640 v4 de fréquence 2.4 GHz. On observe que le temps de calcul est légèrement plus long pour la *backward* que pour la *forward* mais surtout que le temps se concentre au niveau des premières couches du réseau. Les couches convolutionnelles et les couches LSTMs ont des temps similaires pour des tailles de cartes de caractéristiques de sortie similaires.

Le tableau 7.20 nous montre la répartition des temps de calculs par rapport à la phase d'entraînement. On observe que la phase de calcul du coût, et des gradients associés, occupe un temps significatif de presque un quart du temps de calcul total. Cela s'explique par le fait que l'appariement à l'aide de l'algorithme Hongrois est inclus dans ce calcul de coût. Les itérations successives effectuées pour obtenir l'appariement donnent à cet algorithme un temps proportionnel au cube du nombre de boîtes hypothèses prédites par notre réseau.

De plus, comme cet algorithme est itératif, il se parallélise mal. Cela est illustré dans le tableau 7.21 qui illustre les temps de calculs des différentes phases de l'entraînement pour une version GPU de notre algorithme. Un GPU Nvidia® GeForce® TITAN X est utilisé. On observe que le temps alloué au calcul du coût correspond à plus de la moitié du temps de calcul total. Et on observe que cette étape a peu bénéficié du passage de CPU à GPU, les temps restent similaires. Nous avons de plus dans ce tableau 7.21 comparé un réseau sans couche récurrentes LSTMs et un réseau avec deux couches récurrentes insérées après les troisième et quatrième couches convolutionnelles. Bien que ces couches soient déjà vers la sortie du réseau et traitent donc des séquences de tailles

TABLE 7.19 – Temps de calcul, en secondes par époque (6592 images de taille 598x838), pour chaque couche de notre réseau.

Couche	Temps *Forward*	Temps *Backward*
Conv 1	217.00	268.96
LSTM 1	286.82	425.11
Conv 2	107.62	151.85
LSTM 2	124.69	156.33
Conv 3	56.92	81.95
LSTM 3	45.66	58.12
Conv 4	33.97	37.32
LSTM 4	10.17	19.5
Conv 5	0.33	0.51
Linear	0.06	0.1
Divers (pré-traitements, *padding*, *dropout*)	38.24	11.09
Total	921.48	1210.84

TABLE 7.20 – Temps de calcul, sous CPU, en secondes par époque (6592 images de taille 598x838), pour les différentes étapes de l'entraînement.

Etape	Temps (s)
Forward	921.48
Calcul du coût (appariement biparti inclus)	591.87
Backward	1210.84
Mise à jour des paramètres	3.26
Total	2727.45

TABLE 7.21 – Temps de calcul, sous GPU, en secondes par époque (6592 images de taille 598x838), pour les différentes étapes de l'entraînement.

Etape	Avec LSTMs	Sans LSTMs
Extraction des données	16.80	16.80
Forward	109.93	41.89
Calcul du coût (appariement biparti inclus)	475.72	475.72
Backward et mise à jour	159.3	42.69
Total	761.75	577.1

plus réduites que si elles avaient été ajoutées après les premières convolutions, on observe que le calcul de ces opérations récurrentes est responsable d'une grande partie des calculs pour la *forward* comme pour la *backward*.

Chapitre 8

Visualisation des représentations apprises par nos réseaux neuronaux

Après avoir justifié dans le chapitre 7, précédent, et en particulier dans la Section 7.4, les choix au niveau de l'architecture du réseau, nous tentons d'illustrer dans ce chapitre les connaissances et concepts appris par le réseau. En particulier, nous illustrons en Section 8.1 les morceaux d'images qui activent le plus les différents neurones de notre réseau. Cela permet de visualiser ce que chaque neurone a appris à discriminer. Nous illustrons aussi en Section 8.2 la rétro-propagation d'un gradient jusque dans nos images d'entrée. Cela permet de visualiser les informations de contexte transmises par les couches récurrentes de nos différents niveaux.

Visualiser les concepts appris par nos réseaux est intéressant puisque cela permet à la fois de justifier certains de nos choix de conception des modèles, de mieux comprendre pourquoi le réseau fonctionne ou commet des erreurs mais aussi d'envisager des améliorations du modèle. Pour cette raison, d'autres travaux se sont intéressés à cette problématique. En particulier, Zeiler et al. [Zeiler and Fergus, 2014] proposent de visualiser les images ayant les activations les plus fortes pour un neurone donné, ce qui s'apparente à ce que nous montrons en Section 8.1. Zeiler et al. [Zeiler and Fergus, 2014] proposent également une approche par déconvolution pour reconstruire l'image d'entrée à l'aide du signal existant dans un neurone. Cette approche est étendue par Yosinski et al. [Yosinski et al., 2015] en justifiant de normalisations pour améliorer la qualité des visualisations. Des approches proposent de calculer les images qui activeraient le plus les neurones en rétro-propageant dans l'image d'entrée les gradients et en mettant itérativement à jour cette image [Erhan et al., 2009], éventuellement en initialisant les images [Mordvintsev et al., 2015]. Enfin, Simonyan et al. [Simonyan et al., 2013] proposent d'utiliser un réseau entraîné à faire de la classification d'objets et de rétro-propager les gradients jusque dans l'image d'entrée pour obtenir une segmentation de l'objet en question. C'est de cette technique que s'inspire notre visualisation présentée en Section 8.2 qui cherche à visualiser les zones ayant participé à la prédiction des

localisations de nos objets.

8.1 Plus fortes activations des neurones

Afin de visualiser ce qu'apprennent les neurones de nos réseaux, nous visualisons, après chaque couche convolutionnelle, les images correspondant aux positions qui respectivement activent et désactivent le plus chaque neurone. Ces images correspondent aux champs réceptifs convolutionnels correspondant aux positions du neurone dans la carte de caractéristiques. L'objectif est de visualiser ce qui a été appris par les couches de convolutions, même si il y a interaction avec les couches récurrentes dont l'action sera visualisée dans la section suivante.

La sélection des morceaux d'images respectivement les plus activants et désactivants peut être décrite comme suit : nous considérons les images $\boldsymbol{X_n}$ de la base d'entraînement, $n \in \{1,N\}$. Pour chaque image $\boldsymbol{X_n}$, chaque couche l et chaque position (i,j), l'activation de la couche pour un neurone donné u, c'est à dire sa carte de caractéristiques, peut être calculée de la manière suivante :

$$h^{1,u}_{n,i,j} = \sigma\left(\sum_f W^1_{f,u} \cdot X^f_{n,i,j} + b^1_f\right) \tag{8.1}$$

$$h^{l,u}_{n,i,j} = \sigma\left(\sum_f W^l_{f,u} \cdot h^{l-1,f}_{n,i,j} + b^l_f\right) \tag{8.2}$$

Où les $W^l_{f,u}$ et b^l_f sont, respectivement, les paramètres et les biais. Et où σ est la fonction d'activation, une tangente hyperbolique dans notre cas.

Ensuite, pour chaque couche l, pour chaque neurone u (seuls les premiers sont conservés dans les illustrations), nous cherchons les morceaux d'images d'entrée (n,i,j) maximisant respectivement son activation et sa désactivation :

$$\underset{n,i,j}{max}\ h^{l,u}_{n,i,j} \qquad et \qquad \underset{n,i,j}{min}\ h^{l,u}_{n,i,j} \tag{8.3}$$

Nous conservons les 5 (n,i,j) ayant les plus fortes réponses et affichons pour chacun le morceau d'image dans $\boldsymbol{X_n}$ correspondant au champ réceptif de la position (i,j).

Cette expérience a été faite sur les $N = 100$ premières images de documents de la base d'entraînement pour les 4 premières couches de convolution et les résultats sont illustrés dans les

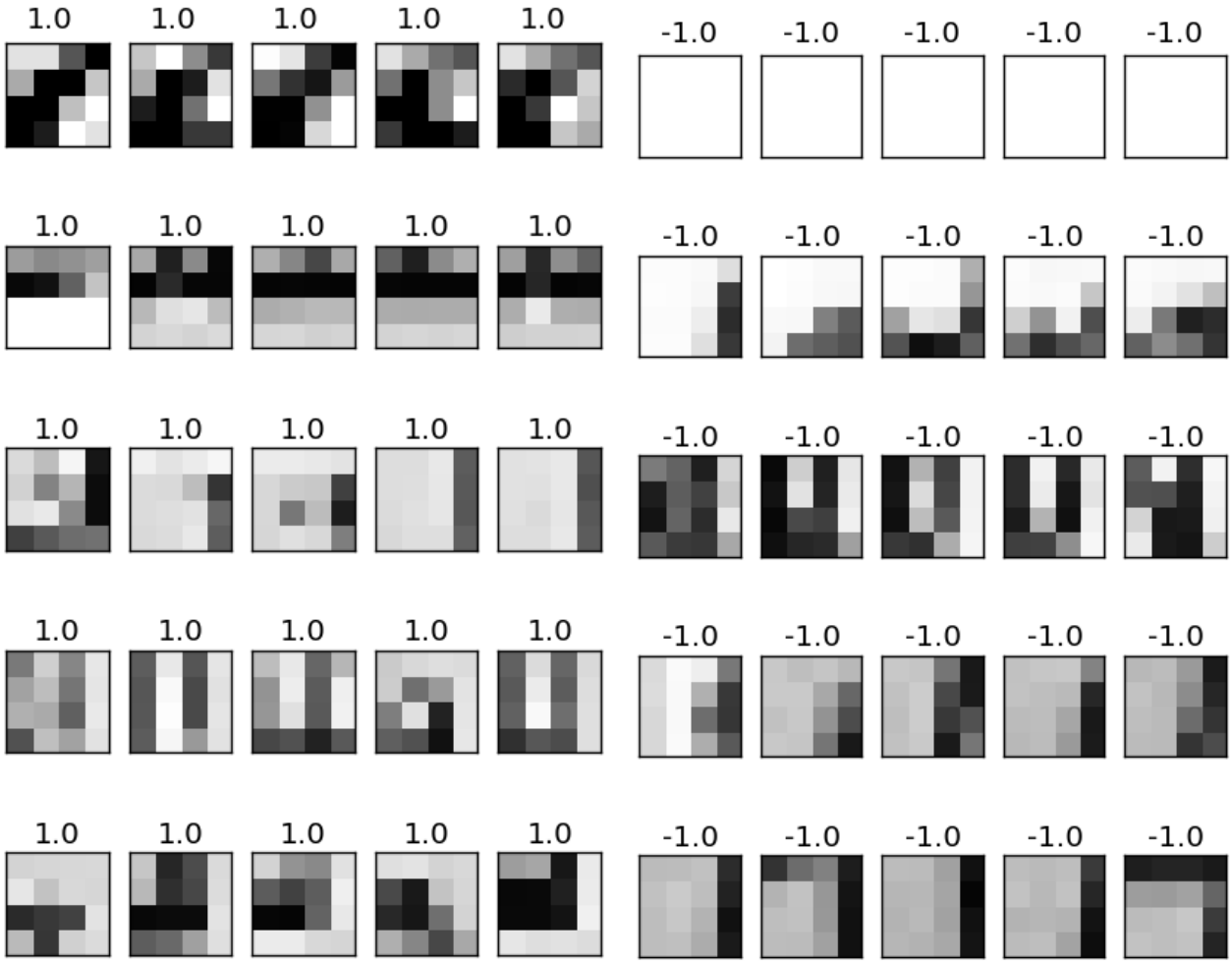


Fig. 8.1 – Illustration des morceaux d'images (de taille 4x4) qui respectivement activent (gauche) et désactivent (droite) le plus des neurones de la première couche convolutionnelle (1 neurone par ligne).

Figures 8.1 à 8.4. Pour chaque neurone nous montrons à la fois les images qui activent le plus et les images qui désactivent le plus les neurones afin de visualiser la discrimination effectuée par le neurone en question.

La Figure 8.1 nous montre ces morceaux d'images pour la première couche convolutionnelle. On observe que le réseau y apprend des caractéristiques de base comme l'intensité des niveaux de gris (premier neurone) ou des gradients horizontaux (deuxième neurone) ou verticaux (troisième et quatrième neurones).

Les morceaux d'images qui activent le plus les neurones de la deuxième couche convolutionnelle sont affichées en Figure 8.2. On voit que les neurones ont tendance à séparer des images comportant du texte et les images n'en comportant pas. Cette tâche de détection de texte est donc déjà en partie effectuée dès cette deuxième couche. On observe également de fortes similarités entre les images qui activent le plus les neurones. Par exemple, le deuxième neurone semble s'être spécialisé

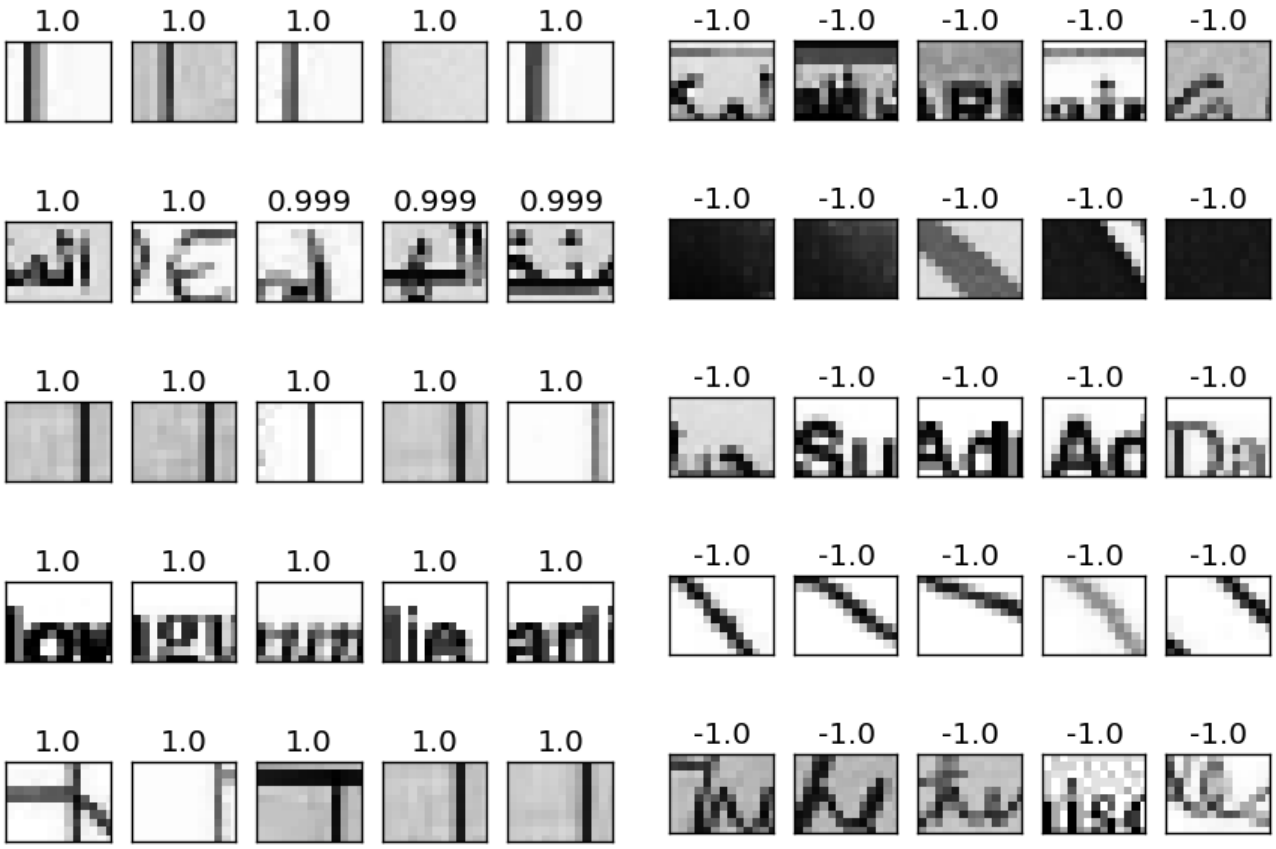


Fig. 8.2 – Illustration des morceaux d'images (de taille 13x10) qui respectivement activent (gauche) et désactivent (droite) le plus des neurones de la deuxième couche convolutionnelle (1 neurone par ligne).

dans le texte en latin alors que le troisième neurone semble être activé prioritairement par le texte en arabe. On observe également que le réseau commence à apprendre des caractéristiques utiles au positionnement des lignes. Le premier neurone détecte les lignes verticales sur la gauche de l'image alors que le troisième neurone détecte celles sur la droite. On observe également que les neurones ont tendance à préférer des images dont l'alignement est similaire, des lignes en bas de l'image par exemple pour le quatrième neurone.

Pour la troisième couche, illustrée en Figure 8.3 et pour la quatrième couche en Figure 8.4, on voit que le réseau apprend des concepts plus élaborés. Par exemple, on voit que le quatrième neurone de la troisième couche apprend à trouver la présence de deux colonnes distinctes dans l'image par rapport à de simples inter-mots. Pour la quatrième couche, on distingue par exemple que les premiers et quatrièmes neurones détectent la présence de débuts de lignes, à des positions différentes, alors que le deuxième neurone permet de savoir si l'on est dans un tableau de formulaire ou pas.

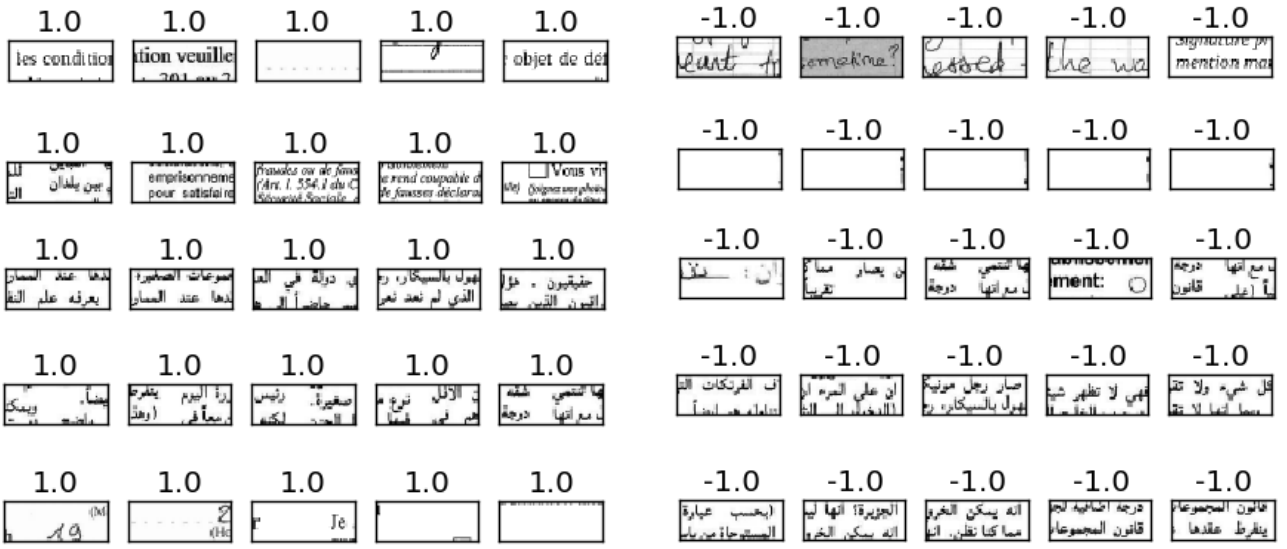


Fig. 8.3 – Illustration des morceaux d'images (de taille 58x22) qui respectivement activent (gauche) et désactivent (droite) le plus des neurones de la troisième couche convolutionnelle (1 neurone par ligne).

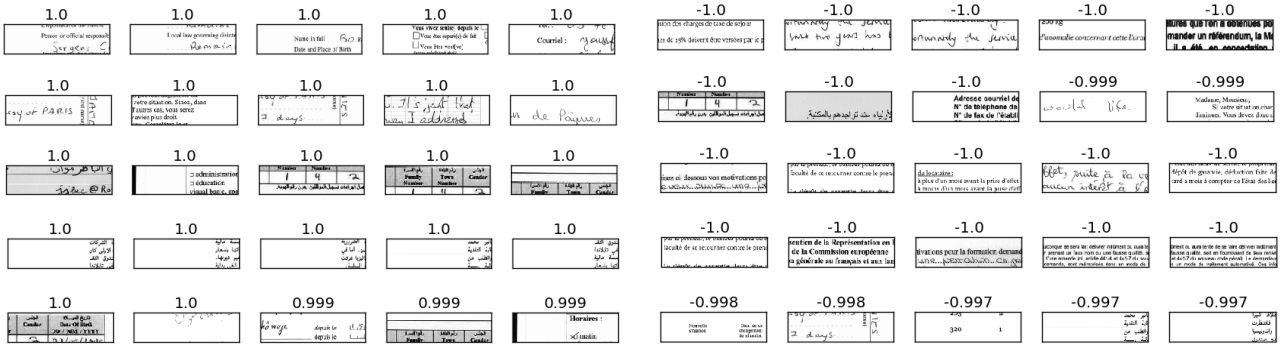


Fig. 8.4 – Illustration des morceaux d'images (de taille 166x46) qui respectivement activent (gauche) et désactivent (droite) le plus des neurones de la quatrième couche convolutionnelle (1 neurone par ligne).

8.2 Visualisation du transfert d'informations contextuelles par rétro-propagation de gradients dans l'image d'entrée

Une des composantes clés de notre réseau est la présence de couches récurrentes 2D-LSTMs qui améliorent fortement notre système en permettant de véhiculer une information de contexte globale. Le champs réceptifs correspondant à ces neurones devient donc l'ensemble de l'image d'entrée et il est plus dur de visualiser comme précédemment ce qu'apprend à véhiculer la LSTM en observant les images qui activent le plus les neurones.

Nous présentons donc une autre méthode dans l'optique de voir quelles parties de l'image d'entrée sont utilisées pour prédire la position d'un objet, ici une ligne de texte. Cette méthode est basée sur la visualisation des gradients dans l'espace de l'image d'entrée. Cela nous permettra également de comparer les différentes coordonnées entre elles et de voir si et quand le contexte est utile.

8.2.1 Méthodologie

Afin de visualiser quelles parties d'une image donnée sont responsables de la prédiction d'une valeur de sortie, un gradient de valeur 1 est appliqué à cette sortie (aucun gradient sur les autres sorties) et ce gradient, qui remplace le gradient habituel de la fonction de coût, est ensuite rétro-propagé (sans mise à jour des paramètres libres) jusque dans l'image d'entrée.

Nous pouvons ensuite afficher la valeur absolue de ces gradients dans l'image d'entrée.

8.2.2 Résultats et analyse

Ces gradients ont été visualisés dans les Figures 8.5 à 8.17. L'image originale est affichée avec, en rouge, la position de la boîte prédite correspondant aux sorties étudiées et, en pointillés verts, la position du champs réceptif convolutionnel correspondant à cette sortie. Nous rappelons que la taille des champs réceptifs convolutionnels est très limitée, ce qui est compensé par l'effet des couches récurrentes 2D-LSTM. Sont ensuite affichés, de gauche à droite et de haut en bas, les gradients dans l'image d'entrée correspondant aux sorties prédisant respectivement la gauche, la droite, le haut, le bas et la confiance des boîtes.

On peut tout d'abord observer, en Figure 8.14, que pour une image simple et une ligne relativement isolée, les gradients rétro-propagés sont très localisés à l'endroit de la ligne et même au

niveau des coordonnées relatives à la sortie. On peut donc supposer que pour des objets simples comme celui ci, le contexte et les 2D-LSTMs qui le transmettent ne sont pas utiles.

On remarque en Figure 8.6 qu'un peu plus d'information contextuelle est utilisée. En particulier, on remarque dans l'image des gradients correspondant à la prédiction de la gauche de la boîte que les positions des gauches des lignes précédentes ont été utilisées. Cela signifie que le réseau a appris à effectuer sa prédiction en fonction du contexte, qu'il a appris à s'aider de l'alignement des lignes pour améliorer sa prédiction. De manière analogue, on observe dans la Figure 8.7 que pour la prédiction de la droite de la ligne, l'attention se porte sur la bordure à droite. C'est grâce à ce contexte que le réseau apprend à ne pas inclure les composantes connexes associées à cette bordure dans la ligne.

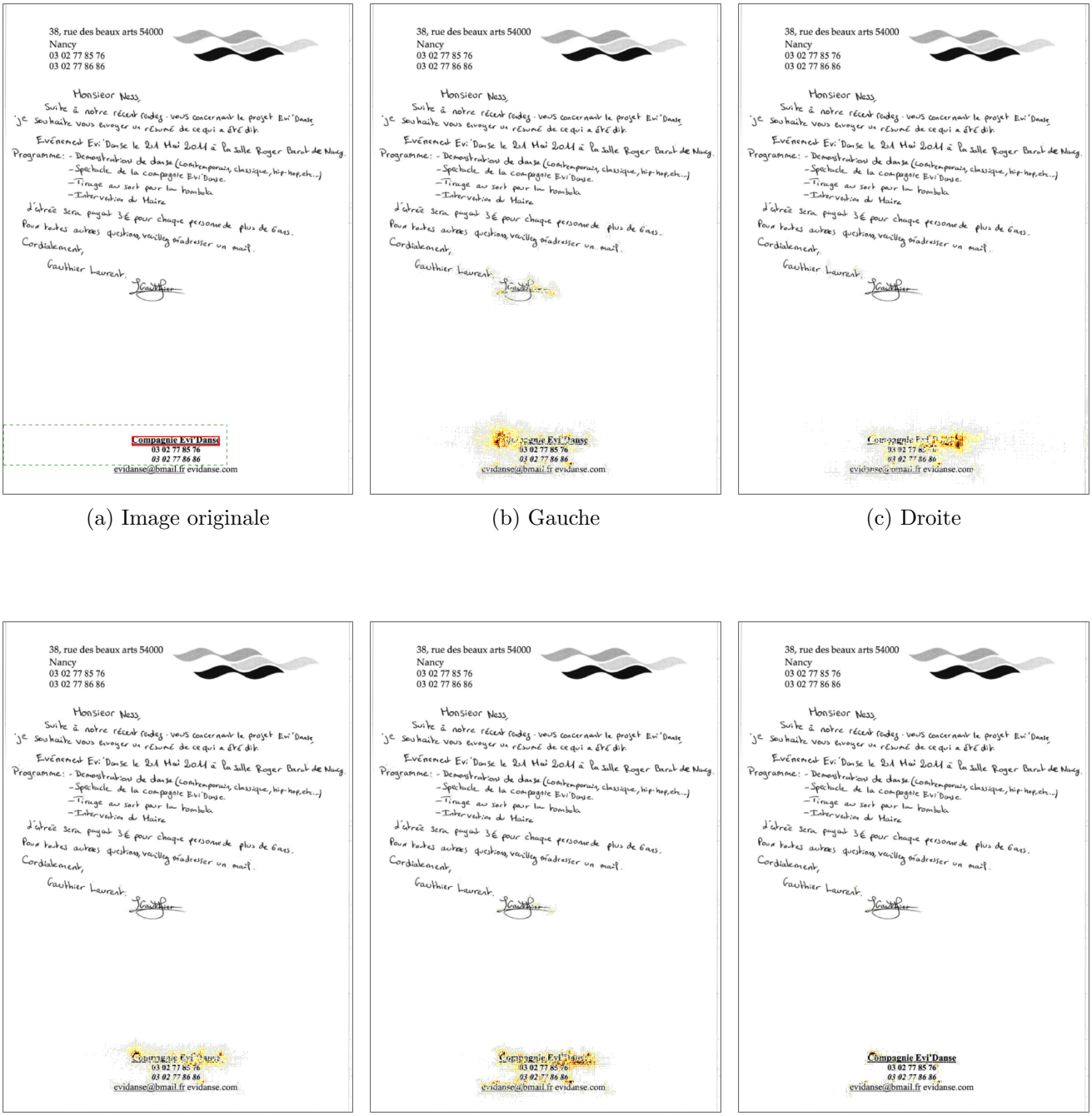
Cette manière d'utiliser le contexte se visualise aussi en Figure 8.8 où, pour la prédiction de la confiance, on remarque que l'attention se porte sur le tiret au début de la ligne. Le réseau a appris que la présence d'un tiret permettait très distinctement de prédire qu'une ligne existait.

Dans les images 8.9 et 8.10 qui correspondent à des images de lettres manuscrites, on peut observer la manière dont le système tient compte de l'inclinaison des lignes. Dans la Figure 8.9, la ligne descend alors que celle de la Figure 8.10 monte. On observe que l'attention associée à la position du haut de la ligne est principalement à gauche de la ligne pour la première image et à droite pour la deuxième. L'analyse est similaire pour la position du bas de la ligne. Cela signifie que le système a appris comment s'adapter à l'inclinaison du texte.

On remarque aussi que dans le cas de la Figure 8.10, la position de la droite de la ligne est nettement en dehors du champs réceptif convolutionnel associé à la sortie qui prédit la boîte et indiqué en pointillés verts. On voit que les attentions liées aux positions de la droite et du haut de la boîte sont principalement en dehors de ce champs réceptif, ce qui souligne la capacité du modèle récurrent à transmettre ces informations.

Cette information de contexte est aussi fortement utilisée dans le cas de documents très structurés comme les tableaux. Ces documents sont de plus parfois plus ambigus et le contexte joue un rôle important dans la définition de ce qu'est une ligne de texte. On remarque dans les Figures 8.11, 8.12 et 8.13 que beaucoup d'attention se porte sur le reste des tableaux. En particulier, pour la Figure 8.12 où l'on cherche à détecter un petit objet, c'est la présence d'objets similaires au dessus et en dessous qui permet de confirmer que cela est bien une ligne. Pour la Figure 8.11, et la prédiction de la position de la gauche de la ligne, c'est la nature du texte à gauche de la ligne qui indique qu'il appartient à une ligne différente et qu'il ne doit pas être inclus ; c'est donc sur ce texte à gauche que se porte l'attention.

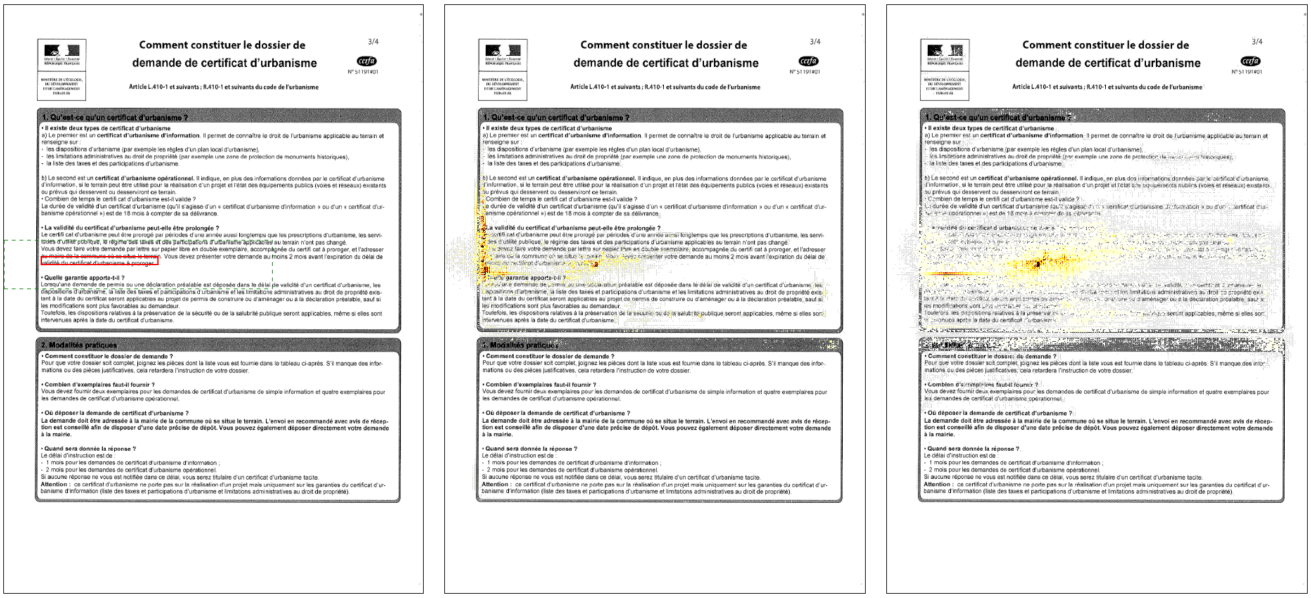
Pour d'autres images compliquées comme des cartes (Figures 8.14 et 8.15) ou lorsque les fonds sont très structurés (Figures 8.16 et 8.17), on observe que l'attention se porte sur un environnement



(a) Image originale (b) Gauche (c) Droite

(d) Haut (e) Bas (f) Confiance

Fig. 8.5 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts (Mieux vu en couleurs).



(a) Image originale (b) Gauche (c) Droite

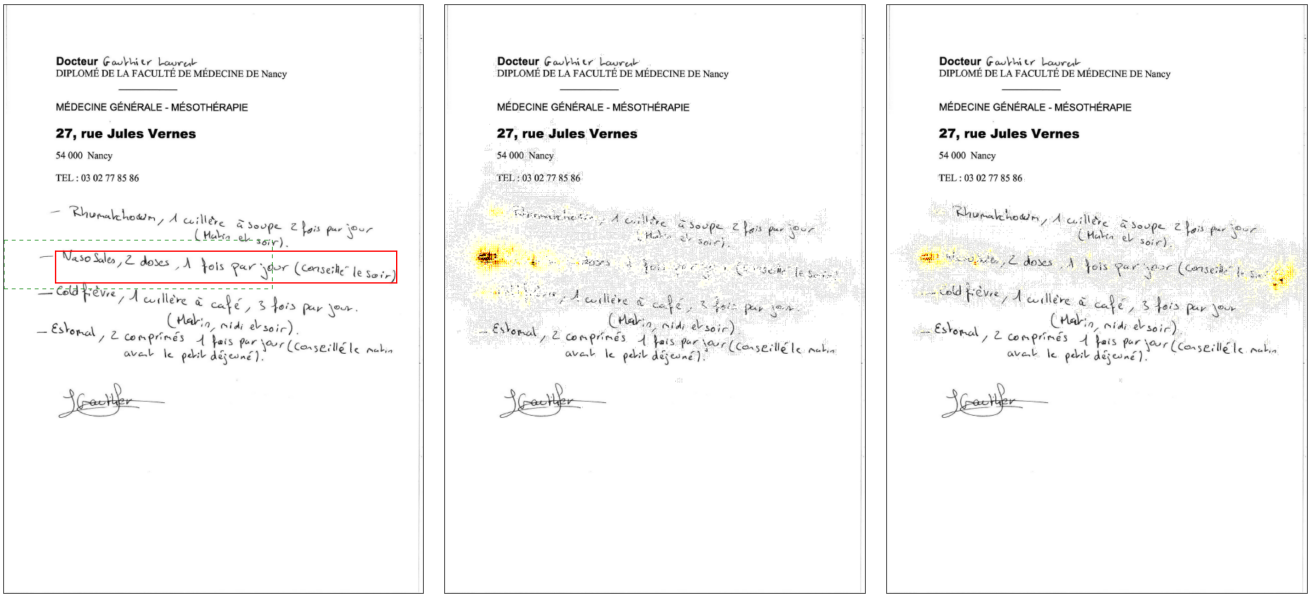
(d) Haut (e) Bas (f) Confiance

Fig. 8.6 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts.

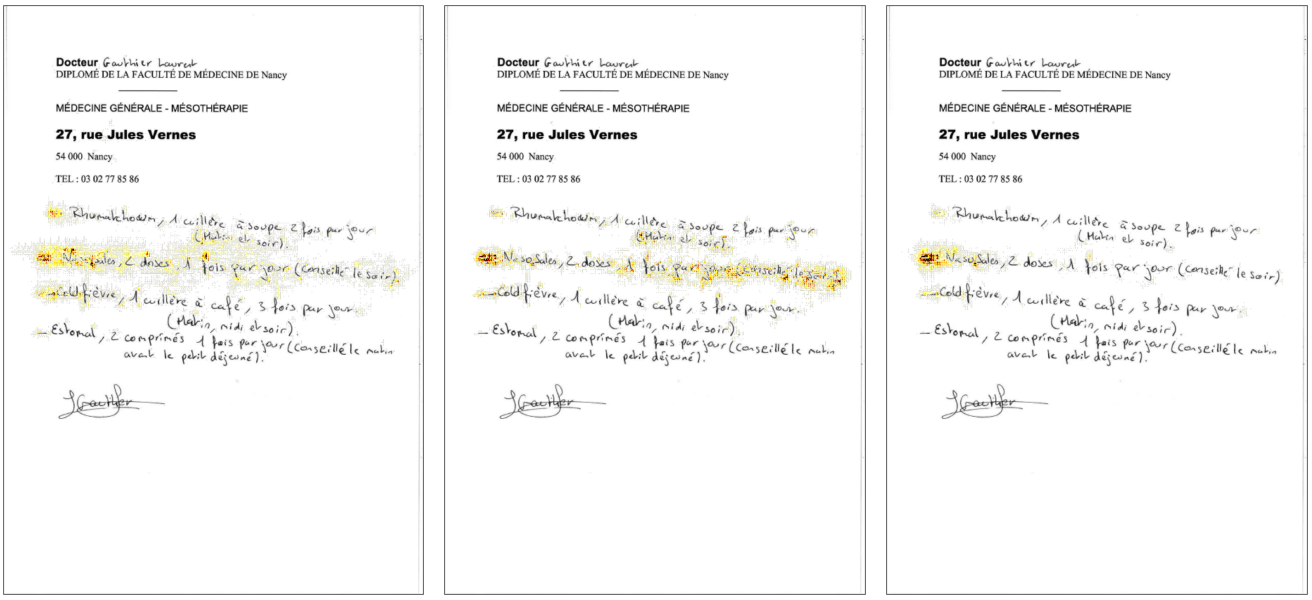
(a) Image originale (b) Gauche (c) Droite

(d) Haut (e) Bas (f) Confiance

Fig. 8.7 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts.

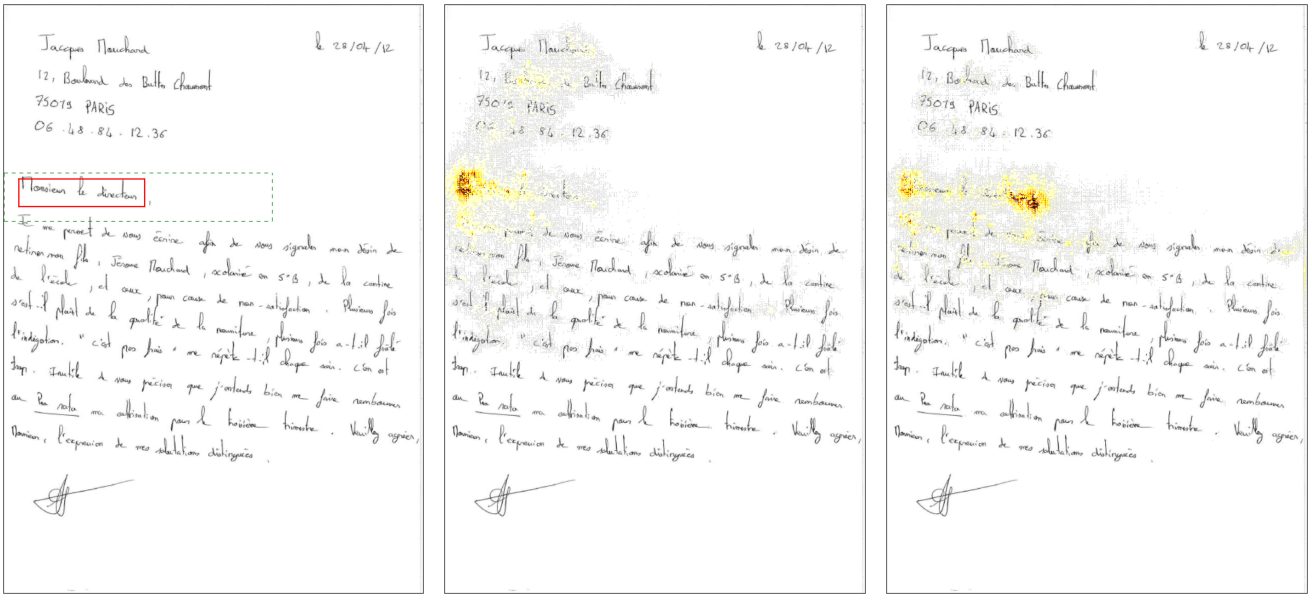


(a) Image originale (b) Gauche (c) Droite



(d) Haut (e) Bas (f) Confiance

Fig. 8.8 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts.



(a) Image originale (b) Gauche (c) Droite

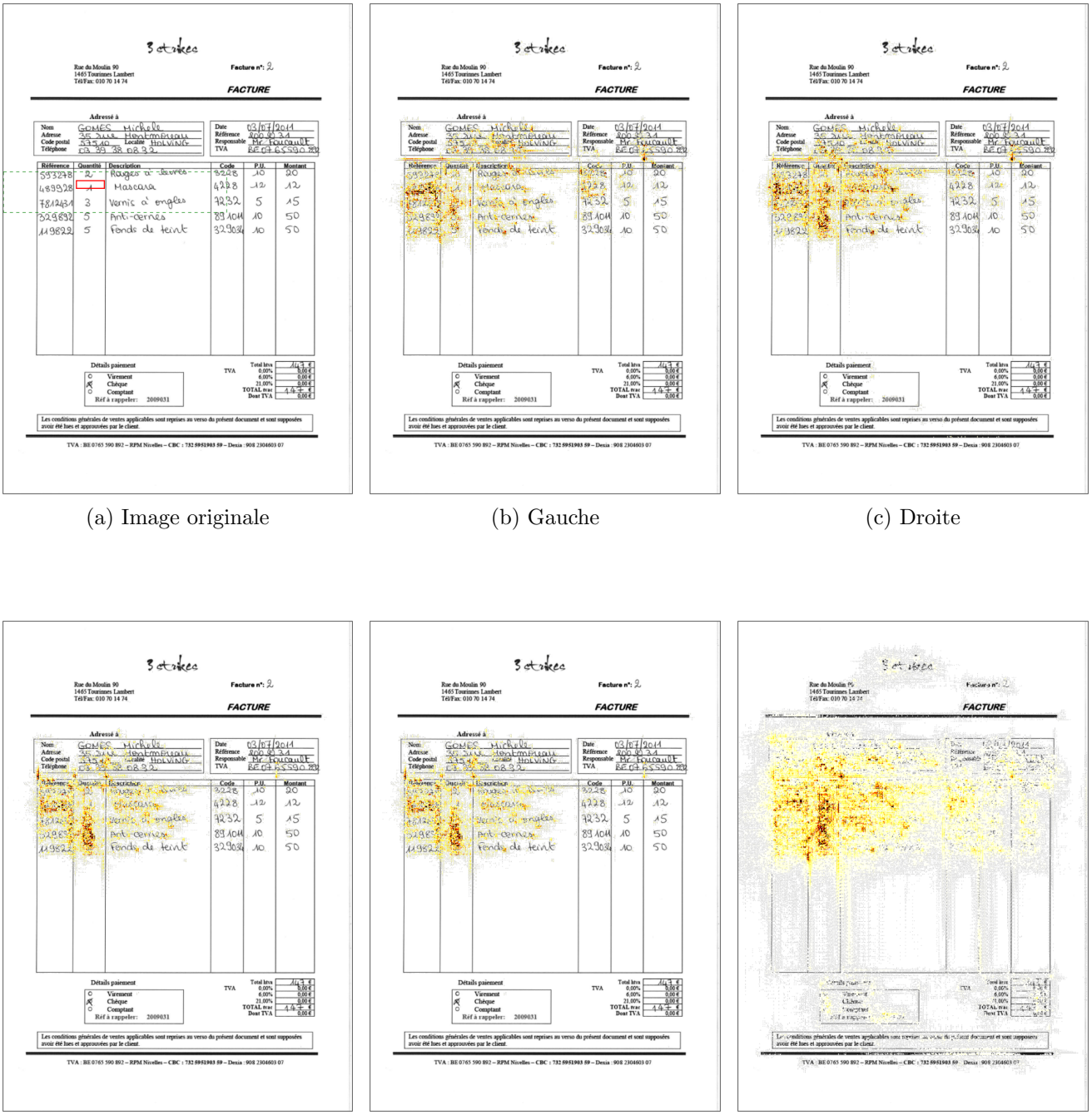
(d) Haut (e) Bas (f) Confiance

Fig. 8.9 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts.

(a) Image originale (b) Gauche (c) Droite

(d) Haut (e) Bas (f) Confiance

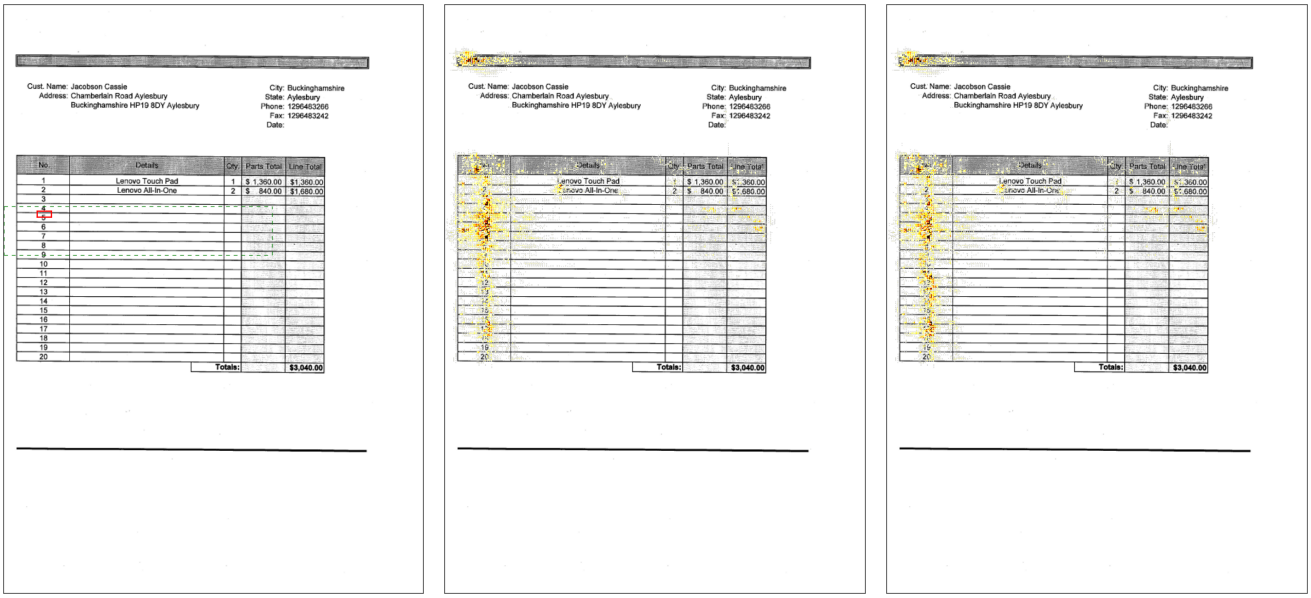
Fig. 8.10 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts.



(a) Image originale (b) Gauche (c) Droite

(d) Haut (e) Bas (f) Confiance

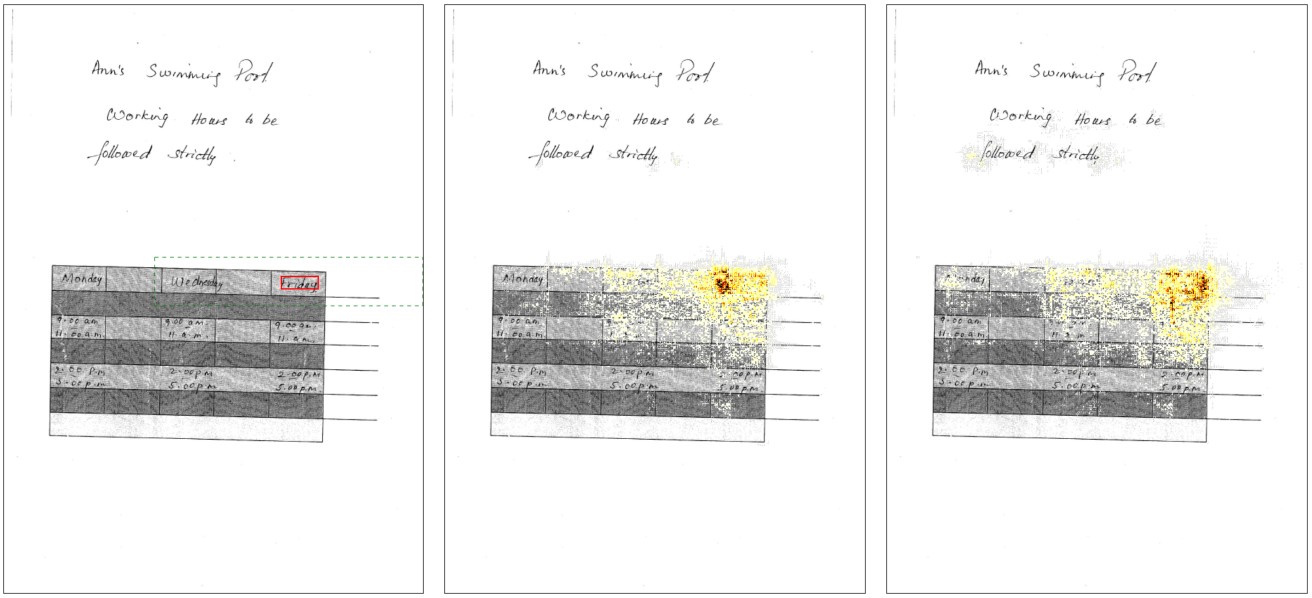
Fig. 8.11 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts.



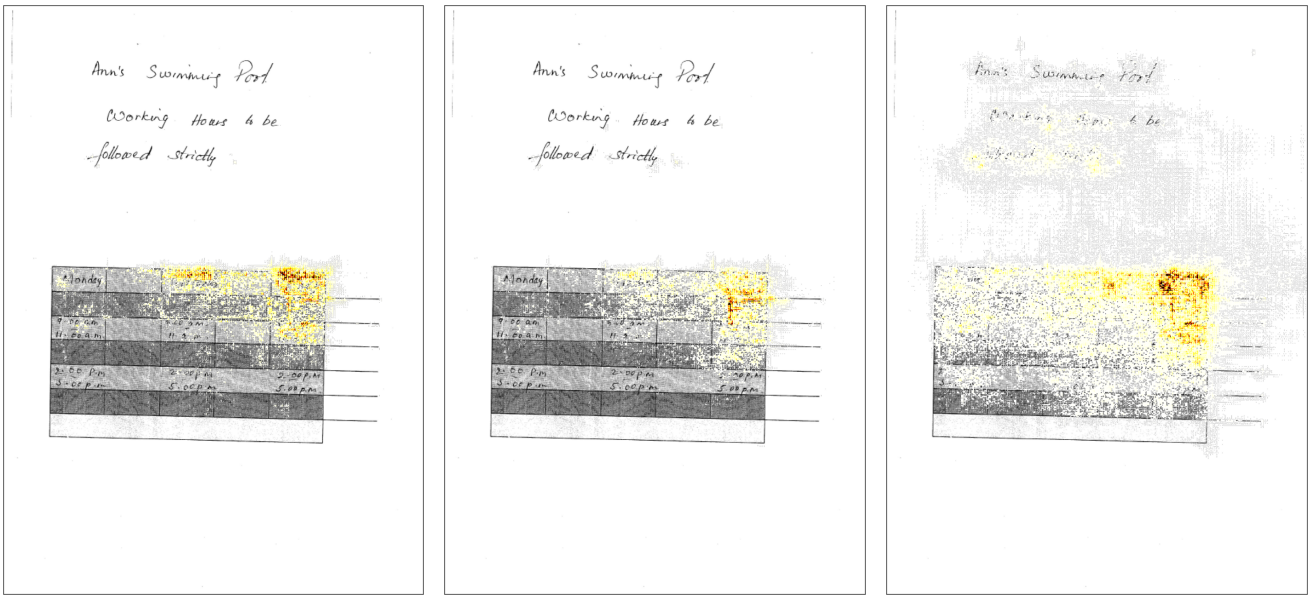
(a) Image originale (b) Gauche (c) Droite

(d) Haut (e) Bas (f) Confiance

Fig. 8.12 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts.



(a) Image originale (b) Gauche (c) Droite



(d) Haut (e) Bas (f) Confiance

Fig. 8.13 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts.

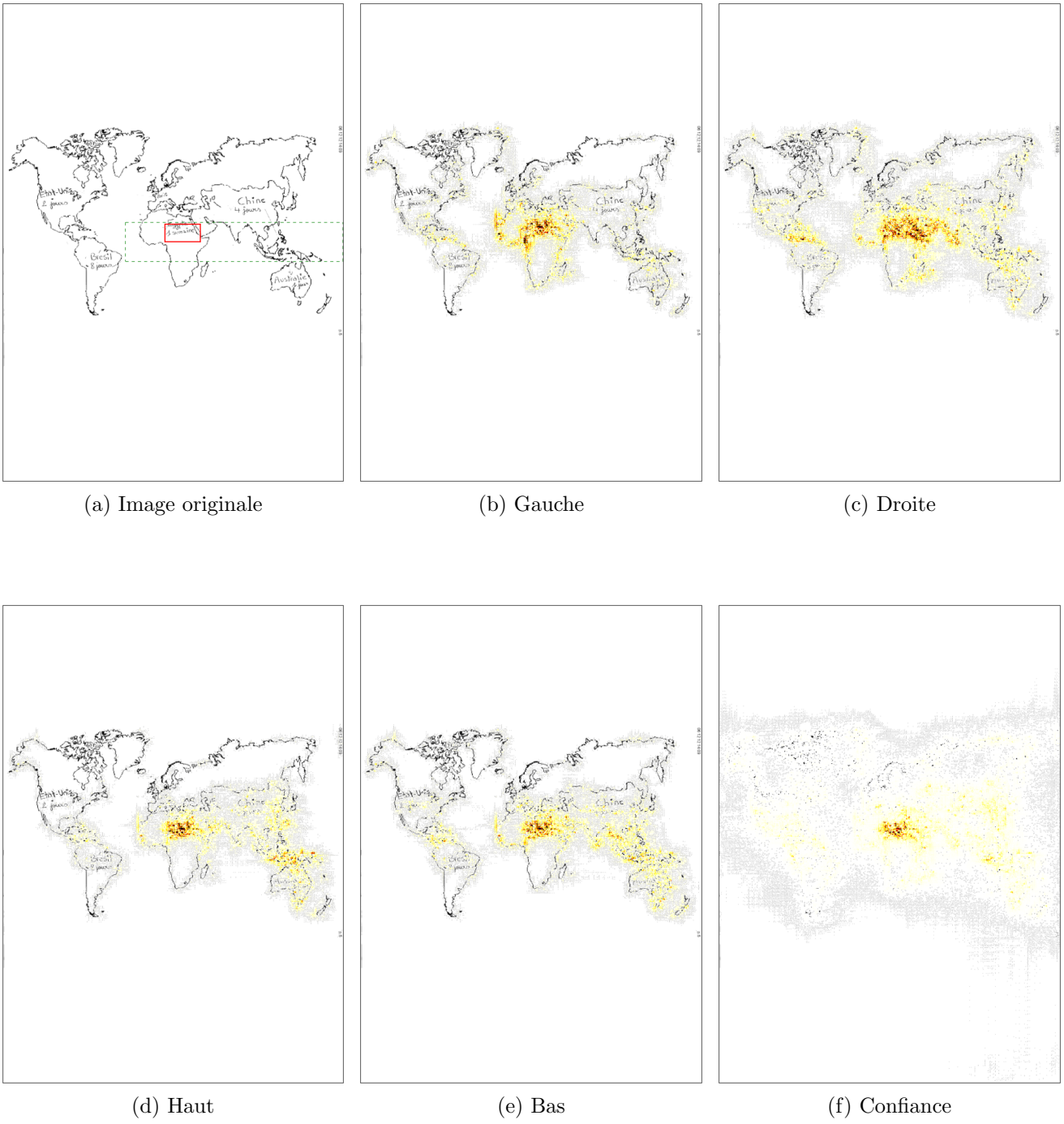
large de la ligne. C'est l'information globale qui permet de comprendre ce qu'est une ligne dans une carte et ce qui appartient ou pas à la texture du fond de l'image.

En conclusion, on observe, d'une part, que le système utilise fortement la capacité des récurrences à transmettre de l'information en dehors de leur champs réceptifs convolutionnels et à, ainsi, transmettre des informations sur le contexte. Cela justifie de nouveau la pertinence du modèle décrit en Section 5 et explique l'amélioration des performances montrées dans la Section 7.4.1. On observe aussi la capacité du système à comprendre la structure du document et à utiliser les informations qui lui paraissent pertinentes pour affiner ses prédictions.

(a) Image originale (b) Gauche (c) Droite

(d) Haut (e) Bas (f) Confiance

Fig. 8.14 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts.



(a) Image originale (b) Gauche (c) Droite

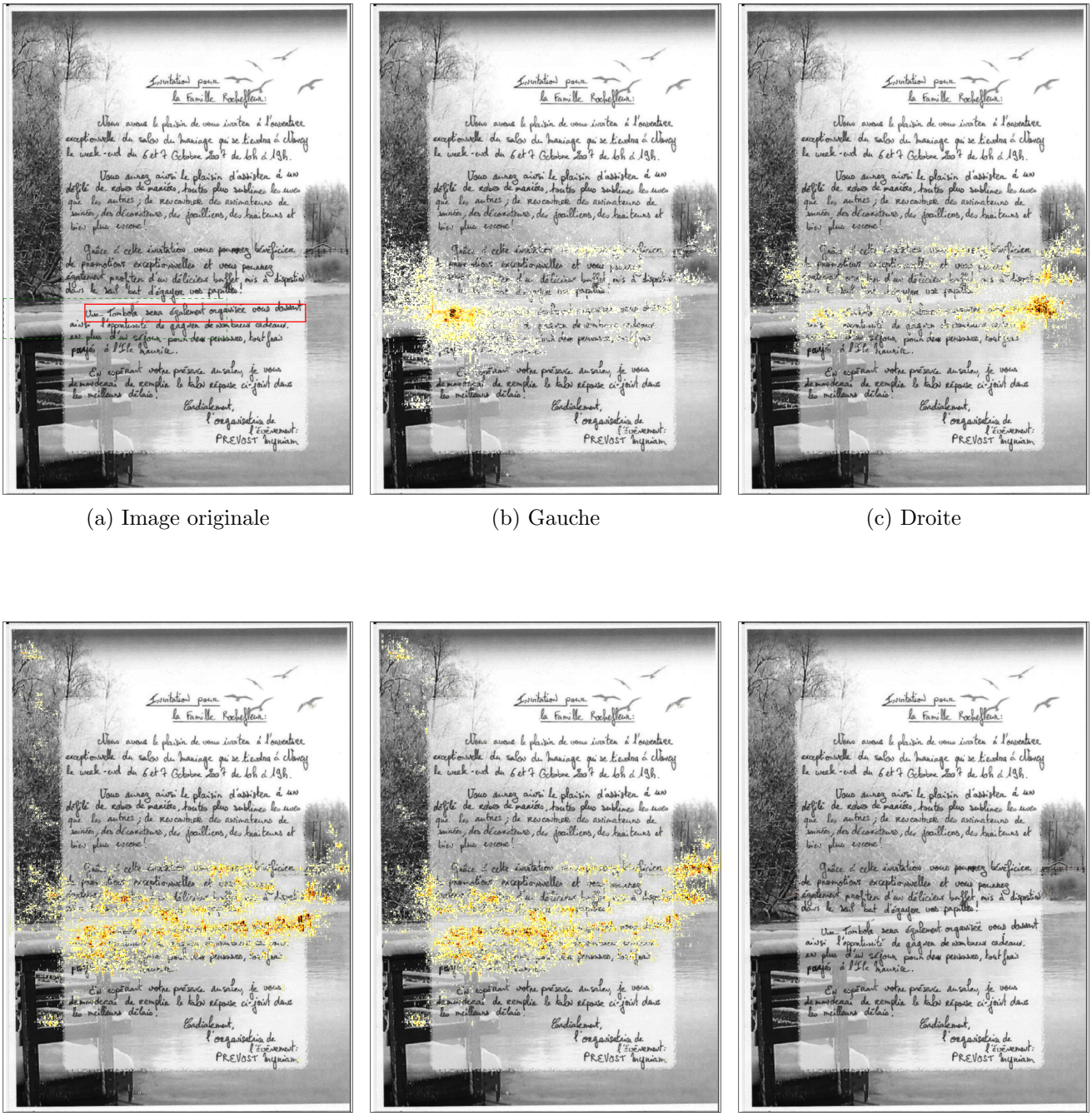
(d) Haut (e) Bas (f) Confiance

Fig. 8.15 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts.

(a) Image originale (b) Gauche (c) Droite

(d) Haut (e) Bas (f) Confiance

Fig. 8.16 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts.



(a) Image originale (b) Gauche (c) Droite

(d) Haut (e) Bas (f) Confiance

Fig. 8.17 – Visualisation de la rétro-propagation de gradients dans l'espace de l'image d'entrée pour les sorties relatives aux positions b) de la gauche c) du haut d) de la droite e) du bas ou f) de la confiance. La boîte correspondant à ces sorties est affichée en rouge dans l'image d'entrée (a) et son champs réceptif convolutionnel en pointillés verts.

Chapitre 9

Conclusions et perspectives

9.1 Conclusions

Dans cette thèse, nous avons proposé des systèmes à base de réseaux de neurones profonds destinés à détecter et localiser automatiquement les lignes de texte présentes dans des images de documents hétérogènes comprenant plusieurs langues, plusieurs types de documents et à la fois de l'écriture manuscrite et imprimée. Cela nous a amené à proposer des systèmes de reconnaissance pleine page du texte robustes et compétitifs.

Utiliser une approche par réseaux de neurones pour effectuer cette tâche permet de généraliser à des documents variés.

Nous avons répondu aux problématiques principales de la tâche de détection de lignes que sont, d'une part, le nombre plus restreint d'exemples annotés pour l'entraînement, et d'autre part, la nécessité de détecter précisément un nombre important et variable de petits objets.

Pour la première, à savoir le faible nombre de données disponibles lorsque l'on se compare aux tâches de classification ou de détection d'objets dans des scènes naturelles, nous avons tout d'abord proposé une technique de segmentation de paragraphe de texte en lignes qui utilise un alignement dynamique et automatique des positions et, de ce fait ne nécessite pas la position explicite des boîtes. Seul le nombre de lignes est nécessaire.

Cette méthode ne s'applique qu'à une segmentation mono-dimensionnelle et ne fonctionne pas pour des documents de pages complètes. Pour cette raison, nous avons introduit un modèle qui prédit directement la valeur des coordonnées de la position des boîtes englobant les lignes de texte. Cette approche est entièrement supervisée et, pour répondre à la problématique du nombre de données annotées disponibles, nous avons proposé une approche locale de la détection

où chaque localisation de l'image est responsable de la localisation des objets qui se trouvent dans son voisinage. Cela nous a permis de partager les paramètres entre les localisations et de mettre en commun leur apprentissage. Cela est rendu possible par le fait que nos objets, les lignes de texte, sont similaires dans les différentes parties de l'image. De plus, nous avons proposé et utilisé une architecture de réseau de neurones avec un nombre réduit de paramètres. La perte de puissance du modèle étant compensé par un plus faible sur-apprentissage.

Pour la seconde problématique, la nécessité de prédire un nombre variable et important d'objets, nous l'avons traitée par l'apprentissage d'une valeur de confiance dans le fait que la boîte existe aux côtés des coordonnées de cette boîte. Un nombre important et fixe de boîtes hypothèses peut ainsi être prédit et la confiance permettra d'en conserver un nombre variable. Comme les objets sont prédits localement, il est possible d'en prédire un nombre important sans faire exploser le nombre de paramètres. Nous avons effectué l'association entre boîtes références et boîtes hypothèses de manière globale à l'aide d'un algorithme d'appariement biparti tout au long de l'apprentissage. La caractère biparti, et le fait que chaque référence soit associée à une et une seule hypothèse, implique que le système va apprendre à prédire le nombre d'objets existants. Cela évite des phases de post-traitements pour choisir les boîtes et permet donc de prédire des objets qui se superposent.

Nous avons constaté l'impact du nombre de coordonnées prédites sur la précision de la prédiction de ces coordonnées. Cela nous a amené à proposer des stratégies alternatives à la détection des boîtes englobant les lignes de texte. Nous avons proposé de détecter les coins de ces boîtes et de les coupler. Mais nous avons surtout proposé un système de reconnaissance pleine page où le détecteur de lignes ne localise que les côtés gauches des lignes de texte et où c'est le reconnaisseur de texte qui est chargé d'ignorer ce qui se trouve à droite de la ligne, y compris d'autres objets textuels éventuels.

Enfin, nous avons proposé l'utilisation de couches de neurones récurrentes spatiales, les 2D-LSTMs, dans notre modèle de détection des lignes. Celles-ci permettent de transmettre des informations contextuelles entre les différentes localisations et ce à différents niveaux de représentation puisque ces couches ont été ajoutées après plusieurs des couches convolutionnelles de notre réseau. Surtout, cela permet de transmettre les informations de mise en page à notre modèle local de détection.

Nous avons visualisé le rôle de ces couches récurrentes en observant les gradients qui sont rétropropagés jusque dans l'image d'entrée. Cela nous a permis d'observer comment l'information contextuelle était transmise à travers le réseau, s'aidant de la structure des documents, en particulier pour les images complexes.

Nous avons obtenus de bons résultats de reconnaissance pleine page sur la base Maurdor, dépassant nettement les performances des systèmes auxquels nous nous sommes comparés, sur

plusieurs types de documents et dans plusieurs langues.

9.2 Perspectives

Cette thèse a été réalisée dans un cadre industriel au sein de la société A2iA. Une perspective directe de cette thèse est l'application des algorithmes présentés aux produits. Cela nécessite de vérifier comment le transfert de l'apprentissage se déroule lorsque l'on applique le réseau de détection et de localisation des lignes de texte à d'autres types d'images, des documents historiques par exemple, ou à d'autres langues. On pourra aussi chercher à savoir quand est-ce que le réentraînement ou l'adaptation des réseaux est nécessaire.

Nous avons défini notre système de localisation d'objets dans le but de détecter les lignes de texte. La problématique ayant amené à ces changements, à savoir comment détecter de nombreux petits objets avec peu d'exemples annotés pour entraîner, peut se retrouver dans d'autres tâches de vision par ordinateur. Des tâches comme la détection de cellules dans des images médicales, la détection de piétons dans des images de rues et d'autres peuvent partager certaines de ces contraintes. Vérifier que notre système peut s'appliquer à d'autres types de signaux pourrait être intéressant.

L'une des faiblesses principales de notre réseau réside dans le fait que deux de ces principaux éléments, les couches récurrentes 2D-LSTM et l'appariement biparti utilisé dans la fonction de coût de l'entraînement, se parallélisent mal et réduisent les gains obtenus par le passage au GPU. On peut se demander si d'autres méthodes que la 2D-LSTM, mieux parallélisables, peuvent être utilisées pour transmettre efficacement le contexte entre les positions. De même, on pourra chercher une manière plus efficace, d'un point de vue calcul, pour réaliser l'appariement entre boîtes références et boîtes hypothèses tout en conservant la propriété de prédire le bon nombre d'objets et on pourra étudier si une approximation de cette appariement pourrait être suffisant.

Nous avons choisi de gérer le problème du manque de données en réduisant le nombre de coordonnées et en partageant la supervision entre les différentes localisation. D'autres manières existent pour traiter ce manque de données. Utiliser un pré-entrainement non supervisé, par exemple avec des auto-encodeur variationnels, est une piste, tout comme la génération d'images synthétiques, éventuellement à l'aide de modèles génératifs adversariaux. On notera que l'utilisation de ces méthodes n'est pas incompatible avec notre méthode et on pourra chercher à étudier leurs synergies.

Enfin, cette approche de détection des lignes de texte directement dans des images de pages nous a permis de fusionner les étapes de détection des zones de texte et de segmentation de ces zones en ligne. Cela permet d'éviter de multiplier les erreurs propres à chaque étape et cela permet

de définir et d'apprendre les tâches de manière jointe puisque notre réseau va à la fois trouver les zones de texte et y allouer le nombre demandé de sorties. Une certaine analogie peut être trouvée avec d'autres tâches d'analyse de documents. Fusionner les étapes de cette manière avait déjà permis, pour les tâches de reconnaissance, de passer d'une classification de caractères à une reconnaissance de mots puis de lignes, s'affranchissant du besoin de segmenter les lignes en mots et en lettres. De même les réseaux de neurones convolutionnels ont permis de fusionner l'extraction des caractéristiques et la classification du contenu. Il reste deux tâches principales dans notre système de reconnaissance pleine page : la détection des lignes et la reconnaissance de leur contenu. Une piste de recherche intéressante viserait à fusionner ces étapes, qui présentent désormais la particularité d'être toutes deux réalisées avec des réseaux de neurones. Partager les paramètres, ou au moins des paramètres, des deux réseaux permettrait de mutualiser leur apprentissage. On peut également se demander comment rendre possible de faire que la reconnaissance du texte soit dérivable par rapport à la localisation des lignes et ainsi permettre de rétro-propager les gradients de bout en bout à travers les deux tâches pour apprendre celles-ci jointement.

Bibliographie

[Abuhaiba et al., 1995] Abuhaiba, I., Datta, S., and Holt, M. J. (1995). Line extraction and stroke ordering of text pages. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 1, pages 390–393. IEEE.

[Afzal et al., 2015] Afzal, M. Z., Pastor-Pellicer, J., Shafait, F., Breuel, T. M., Dengel, A., and Liwicki, M. (2015). Document image binarization using lstm : A sequence learning approach. In *Proceedings of the 3rd International Workshop on Historical Document Imaging and Processing*, pages 79–84. ACM.

[Alexe et al., 2010] Alexe, B., Deselaers, T., and Ferrari, V. (2010). What is an object ? In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 73–80. IEEE.

[Alexe et al., 2012] Alexe, B., Deselaers, T., and Ferrari, V. (2012). Measuring the objectness of image windows. *IEEE transactions on pattern analysis and machine intelligence*, 34(11) :2189–2202.

[Arvanitopoulos and Süsstrunk, 2014] Arvanitopoulos, N. and Süsstrunk, S. (2014). Seam carving for text line extraction on color and grayscale historical manuscripts. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 726–731. IEEE.

[Baechler et al., 2013] Baechler, M., Liwicki, M., and Ingold, R. (2013). Text line extraction using dmlp classifiers for historical manuscripts. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 1029–1033. IEEE.

[Baird, 1994] Baird, H. S. (1994). Background structure in document images. *International Journal of Pattern Recognition and Artificial Intelligence*, 8(05) :1013–1030.

[Bartz et al., 2017] Bartz, C., Yang, H., and Meinel, C. (2017). Stn-ocr : A single neural network for text detection and text recognition. *arXiv preprint arXiv :1707.08831*.

[Basu et al., 2007] Basu, S., Chaudhuri, C., Kundu, M., Nasipuri, M., and Basu, D. K. (2007). Text line extraction from multi-skewed handwritten documents. *Pattern Recognition*, 40(6) :1825–1839.

[Behnke, 2005] Behnke, S. (2005). Face localization and tracking in the neural abstraction pyramid. *Neural Computing & Applications*, 14(2) :97–103.

[Bell et al., 2016] Bell, S., Zitnick, L., Bala, K., and Girshick, R. (2016). Inside-outside net : Detecting objects in context with skip pooling and recurrent neural networks. In *IEEE Conf. on Computer Vision and Pattern Recognition*.

[Bengio et al., 2009] Bengio, Y. et al. (2009). Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1) :1–127.

[Bilen and Vedaldi, 2016] Bilen, H. and Vedaldi, A. (2016). Integrated perception with recurrent multi-task neural networks. In *Advances in neural information processing systems*, pages 235–243.

[Bloomberg et al., 1995] Bloomberg, D. S., Kopec, G. E., and Dasari, L. (1995). Measuring document image skew and orientation. *Document Recognition*, 2422 :302–316.

[Bluche, 2016] Bluche, T. (2016). Joint line segmentation and transcription for end-to-end handwritten paragraph recognition. In *Advances in Neural Information Processing Systems*, pages 838–846.

[Bluche et al., 2016a] Bluche, T., Kermorvant, C., Ney, H., and Louradour, J. (2016a). La ctc et son intrigant label "blank". In *Colloque International Francophone sur l'Ecrit et le Document (CIFED)*.

[Bluche et al., 2016b] Bluche, T., Louradour, J., and Messina, R. (2016b). Scan, attend and read : End-to-end handwritten paragraph recognition with mdlstm attention. *arXiv preprint arXiv :1604.03286*.

[Bluche et al., 2014] Bluche, T., Moysset, B., and Kermorvant, C. (2014). Automatic line segmentation and ground-truth alignment of handwritten documents. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 667–672. IEEE.

[Bottou, 1998] Bottou, L. (1998). Online learning and stochastic approximations. *On-line learning in neural networks*, 17(9) :142.

[Boulid et al., 2016] Boulid, Y., Souhar, A., and Elkettani, M. Y. (2016). Segmentation approach of arabic manuscripts text lines based on multi agent systems. *International Journal of Computer Information Systems and Industrial Management Applications*, 8 :173–183.

[Brodić and Milivojević, 2016] Brodić, D. and Milivojević, Z. N. (2016). Text line segmentation with the parametric water flow algorithm. *Information Technology And Control*, 45(1) :52–61.

[Brunessaux et al., 2014] Brunessaux, S., Giroux, P., Grilheres, B., Manta, M., Bodin, M., Choukri, K., Galibert, O., and Kahn, J. (2014). The maurdor project : Improving automatic processing of digital documents. In *Document Analysis Systems (DAS), 2014 11th IAPR International Workshop on*, pages 349–354. IEEE.

[Bulacu et al., 2007] Bulacu, M., van Koert, R., Schomaker, L., and van der Zant, T. (2007). Layout analysis of handwritten historical documents for searching the archive of the cabinet of the dutch queen. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 1, pages 357–361. IEEE.

[Byeon et al., 2015] Byeon, W., Breuel, T. M., Raue, F., and Liwicki, M. (2015). Scene labeling with lstm recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3547–3555.

[Carreira and Sminchisescu, 2010] Carreira, J. and Sminchisescu, C. (2010). Constrained parametric min-cuts for automatic object segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3241–3248. IEEE.

[Chatfield et al., 2014] Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Return of the devil in the details : Delving deep into convolutional nets. *arXiv preprint arXiv :1405.3531*.

[Chen and Seuret, 2017] Chen, K. and Seuret, M. (2017). Convolutional neural networks for page segmentation of historical document images. *arXiv preprint arXiv :1704.01474*.

[Cho et al., 2015] Cho, K., Courville, A., and Bengio, Y. (2015). Describing multimedia content using attention-based encoder-decoder networks. *IEEE Transactions on Multimedia*, 17(11) :1875–1886.

[Chung et al., 2014] Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv :1412.3555*.

[Dai et al., 2016] Dai, J., Li, Y., He, K., and Sun, J. (2016). R-fcn : Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387.

[Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE.

[Davis et al., 2015] Davis, B. L., Barrett, W. A., and Swingle, S. D. (2015). Min-cut segmentation of cursive handwriting in tabular documents. In *SPIE/IS&T Electronic Imaging*, pages 940208–940208. International Society for Optics and Photonics.

[Delakis and Garcia, 2008] Delakis, M. and Garcia, C. (2008). Text detection with convolutional neural networks. In *Int. Conf. on Computer Vision Theory and Applications*, pages 290–294.

[Doetsch et al., 2016] Doetsch, P., Zeyer, A., Voigtlaender, P., Kulikov, I., Schlüter, R., and Ney, H. (2016). Returnn : The rwth extensible training framework for universal recurrent neural networks. *arXiv :1608.00895*.

[Du et al., 2009] Du, X., Pan, W., and Bui, T. D. (2009). Text line segmentation in handwritten documents using mumford–shah model. *Pattern Recognition*, 42(12) :3136–3145.

[Duchi et al., 2011] Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul) :2121–2159.

[Endres and Hoiem, 2010] Endres, I. and Hoiem, D. (2010). Category independent object proposals. *Computer Vision–ECCV 2010*, pages 575–588.

[Erhan et al., 2009] Erhan, D., Bengio, Y., Courville, A., and Vincent, P. (2009). Visualizing higher-layer features of a deep network. *University of Montreal*, 1341 :3.

[Erhan et al., 2014] Erhan, D., Szegedy, C., Toshev, A., and Anguelov, D. (2014). Scalable object detection using deep neural networks. In *IEEE Conf. on Computer Vision and Pattern Recognition*.

[Eskenazi et al., 2017] Eskenazi, S., Gomez-Krämer, P., and Ogier, J.-M. (2017). A comprehensive survey of mostly textual document segmentation algorithms since 2008. *Pattern Recognition*, 64 :1–14.

[Everingham et al., 2010] Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2) :303–338.

[Farabet, 2013] Farabet, C. (2013). *Towards real-time image understanding with convolutional networks*. PhD thesis, Université Paris-Est.

[Farabet et al., 2013] Farabet, C., Couprie, C., Najman, L., and LeCun, Y. (2013). Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8) :1915–1929.

[Feldbach and Tonnies, 2001] Feldbach, M. and Tonnies, K. D. (2001). Line detection and segmentation in historical church registers. In *Document Analysis and Recognition, 2001. Proceedings. Sixth International Conference on*, pages 743–747. IEEE.

[Felzenszwalb et al., 2010] Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(9).

[Felzenszwalb and Huttenlocher, 2004] Felzenszwalb, P. F. and Huttenlocher, D. P. (2004). Efficient graph-based image segmentation. *International journal of computer vision*, 59(2) :167–181.

[Fischer et al., 2014] Fischer, A., Baechler, M., Garz, A., Liwicki, M., and Ingold, R. (2014). A combined system for text line extraction and handwriting recognition in historical documents. In *Document Analysis Systems (DAS), 2014 11th IAPR International Workshop on*, pages 71–75. IEEE.

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

[Fukushima, 1979] Fukushima, K. (1979). Neural network model for a mechanism of pattern recognition unaffected by shift in position- neocognitron. *Electron. & Commun. Japan*, 62(10) :11–18.

[Fukushima and Miyake, 1982] Fukushima, K. and Miyake, S. (1982). Neocognitron : A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer.

[Galibert et al., 2014] Galibert, O., Kahn, J., and Oparin, I. (2014). The zonemap metric for page segmentation and area classification in scanned documents. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 2594–2598. IEEE.

[Ganin et al., 2016] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59) :1–35.

[Gers et al., 1999] Gers, F. A., Schmidhuber, J., and Cummins, F. (1999). Learning to forget : Continual prediction with lstm.

[Gers et al., 2002] Gers, F. A., Schraudolph, N. N., and Schmidhuber, J. (2002). Learning precise timing with lstm recurrent networks. *Journal of machine learning research*, 3(Aug) :115–143.

[Girshick, 2015] Girshick, R. (2015). Fast R-CNN. In *Int. Conf. on Computer Vision*.

[Girshick et al., 2014] Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conf. on Computer Vision and Pattern Recognition*.

[Graves et al., 2006] Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). Connectionist temporal classification : labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM.

[Graves and Schmidhuber, 2009] Graves, A. and Schmidhuber, J. (2009). Offline handwriting recognition with multidimensional recurrent neural networks. In *Advances in neural information processing systems*, pages 545–552.

[Greff et al., 2016] Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. (2016). Lstm : A search space odyssey. *IEEE transactions on neural networks and learning systems*.

[Grosicki and El Abed, 2009] Grosicki, E. and El Abed, H. (2009). ICDAR 2009 handwriting recognition competition. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*, pages 1398–1402. IEEE.

[Grosicki and El-Abed, 2011] Grosicki, E. and El-Abed, H. (2011). Icdar 2011-french handwriting recognition competition. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 1459–1463. IEEE.

[Gupta et al., 2016] Gupta, A., Vedaldi, A., and Zisserman, A. (2016). Synthetic data for text localisation in natural images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2315–2324.

[Hannun et al., 2014] Hannun, A. Y., Maas, A. L., Jurafsky, D., and Ng, A. Y. (2014). First-pass large vocabulary continuous speech recognition using bi-directional recurrent dnns. *arXiv preprint arXiv :1408.2873*.

[He et al., 2016] He, T., Huang, W., Qiao, Y., and Yao, J. (2016). Text-attentional convolutional neural network for scene text detection. *IEEE transactions on image processing*, 25(6) :2529–2541.

[He et al., 2017] He, W., Zhang, X.-Y., Yin, F., and Liu, C.-L. (2017). Deep direct regression for multi-oriented scene text detection. *arXiv preprint arXiv :1703.08289*.

[Hebert et al., 2011] Hebert, D., Paquet, T., and Nicolas, S. (2011). Continuous crf with multi-scale quantization feature functions application to structure extraction in old newspaper. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 493–497. IEEE.

[Hinton et al., 2012] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv :1207.0580*.

[Hochreiter, 1991] Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen netzen. *Diploma, Technische Universität München*, 91.

[Hochreiter, 1998] Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02) :107–116.

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

[Huang et al., 2016] Huang, Q., Wang, W., Zhou, K., You, S., and Neumann, U. (2016). Scene labeling using recurrent neural networks with explicit long range contextual dependency. *arXiv preprint arXiv :1611.07485*.

[Jaderberg et al., 2016] Jaderberg, M., Simonyan, K., Vedaldi, A., and Zisserman, A. (2016). Reading text in the wild with convolutional neural networks. *International Journal of Computer Vision*, 116(1) :1–20.

[Jaderberg et al., 2015] Jaderberg, M., Simonyan, K., Zisserman, A., et al. (2015). Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025.

[Jones and Viola, 2001] Jones, M. J. and Viola, P. (2001). Robust real-time object detection. In *Workshop on statistical and computational theories of vision*, volume 266, page 56.

[Jung, 2001] Jung, K. (2001). Neural network-based text location in color images. *Pattern Recognition Letters*, 22(14) :1503–1515.

[Kalchbrenner et al., 2015] Kalchbrenner, N., Danihelka, I., and Graves, A. (2015). Grid long short-term memory. *arXiv preprint arXiv :1507.01526*.

[Karatzas et al., 2015] Karatzas, D., Gomez-Bigorda, L., Nicolaou, A., Ghosh, S., Bagdanov, A., Iwamura, M., Matas, J., Neumann, L., Chandrasekhar, V. R., Lu, S., et al. (2015). Icdar 2015 competition on robust reading. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 1156–1160. IEEE.

[Khayyat et al., 2012] Khayyat, M., Lam, L., Suen, C. Y., Yin, F., and Liu, C.-L. (2012). Arabic handwritten text line extraction by applying an adaptive mask to morphological dilation. In *Document Analysis Systems (DAS), 2012 10th IAPR International Workshop on*, pages 100–104. IEEE.

[Kim, 1997] Kim, S. N. S. G. (1997). Penman : A system for reading unconstrained handwritten page images. In *Proceedings 1997 Symposium on Document Image Understanding Technology*, page 142. UMD.

[Kingma and Ba, 2014] Kingma, D. and Ba, J. (2014). Adam : A method for stochastic optimization. *arXiv preprint arXiv :1412.6980*.

[Kingma and Welling, 2013] Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv :1312.6114*.

[Kise et al., 1998] Kise, K., Sato, A., and Iwata, M. (1998). Segmentation of page images using the area Voronoi diagram. *Computer Vision and Image Understanding*, 70(3) :370–382.

[Kiwiel, 2001] Kiwiel, K. C. (2001). Convergence and efficiency of subgradient methods for quasiconvex minimization. *Mathematical programming*, 90(1) :1–25.

[Krähenbühl and Koltun, 2011] Krähenbühl, P. and Koltun, V. (2011). Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in neural information processing systems*, pages 109–117.

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing System*.

[Krogh and Hertz, 1992] Krogh, A. and Hertz, J. A. (1992). A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957.

[Kuhn, 1955] Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval Research Logistics (NRL)*, 2(1-2) :83–97.

[Kumar et al., 2006] Kumar, K. S., Namboodiri, A. M., and Jawahar, C. (2006). Learning segmentation of documents with complex scripts. In *Computer Vision, Graphics and Image Processing*, pages 749–760. Springer.

[Lafferty et al., 2001] Lafferty, J., McCallum, A., and Pereira, F. C. (2001). Conditional random fields : Probabilistic models for segmenting and labeling sequence data.

[LeCun, 1998] LeCun, Y. (1998). The MNIST database of handwritten digits. *http ://yann. lecun. com/exdb/mnist/*.

[LeCun et al., 1989] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4) :541–551.

[Lemaitre and Camillerapp, 2006] Lemaitre, A. and Camillerapp, J. (2006). Text line extraction in handwritten document with kalman filter applied on low resolution image. In *Document*

Image Analysis for Libraries, 2006. DIAL'06. Second International Conference on, pages 8–pp. IEEE.

[Levenshtein, 1966] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

[Li et al., 2016] Li, S., Zhang, J., Guo, Q., Lei, J., and Tu, D. (2016). Generating image descriptions with multidirectional 2d long short-term memory. *IET Computer Vision*, 11(1) :104–111.

[Li et al., 2008] Li, Y., Zheng, Y., Doermann, D., and Jaeger, S. (2008). Script-independent text line segmentation in freestyle handwritten documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(8) :1313–1329.

[Liang et al., 2015] Liang, X., Wei, Y., Shen, X., Yang, J., Lin, L., and Yan, S. (2015). Proposal-free network for instance-level object segmentation. *arXiv preprint arXiv :1509.02636*.

[Likforman-Sulem et al., 1995] Likforman-Sulem, L., Hanimyan, A., and Faure, C. (1995). A Hough based algorithm for extracting text lines in handwritten documents. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 2, pages 774–777. IEEE.

[Likforman-Sulem et al., 2007] Likforman-Sulem, L., Zahour, A., and Taconet, B. (2007). Text line segmentation of historical documents : a survey. *Int. Journal of Document Analysis and Recognition*, 9(2-4) :123–138.

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

[Lin et al., 2013] Lin, M., Chen, Q., and Yan, S. (2013). Network in network. *arXiv preprint arXiv :1312.4400*.

[Lin et al., 2016] Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2016). Feature pyramid networks for object detection. *arXiv preprint arXiv :1612.03144*.

[Liu et al., 2016] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. (2016). Ssd : Single shot multibox detector. In *European Conf. on Computer Vision*.

[Liu and Jin, 2017] Liu, Y. and Jin, L. (2017). Deep matching prior network : Toward tighter multi-oriented text detection. *arXiv preprint arXiv :1703.01425*.

[Long et al., 2015] Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440.

[Louloudis et al., 2009] Louloudis, G., Gatos, B., Pratikakis, I., and Halatsis, C. (2009). Text line and word segmentation of handwritten documents. *Pattern Recognition*, 42(12) :3169–3183.

[Louloudis et al., 2006] Louloudis, G., Gatos, B., Pratikakis, I., and Halatsis, K. (2006). A block-based Hough transform mapping for text line detection in handwritten documents. In *Tenth International Workshop on Frontiers in Handwriting Recognition*. Suvisoft.

[Lowe, 2004] Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision*.

[Ma et al., 2017] Ma, J., Shao, W., Ye, H., Wang, L., Wang, H., Zheng, Y., and Xue, X. (2017). Arbitrary-oriented scene text detection via rotation proposals. *arXiv preprint arXiv :1703.01086*.

[Messelodi and Modena, 1999] Messelodi, S. and Modena, C. M. (1999). Automatic identification and skew estimation of text lines in real scene images. *Pattern Recognition*, 32(5) :791–810.

[Mordan et al., 2017] Mordan, T., Thome, N., Cord, M., and Henaff, G. (2017). Deformable part-based fully convolutional network for object detection. *arXiv preprint arXiv :1707.06175*.

[Mordvintsev et al., 2015] Mordvintsev, A., Olah, C., and Tyka, M. (2015). Inceptionism : Going deeper into neural networks. *Google Research Blog. Retrieved June*, 20 :14.

[Moysset et al., 2015a] Moysset, B., Adam, P., Wolf, C., and Louradour, J. (2015a). Space displacement localization neural networks to locate origin points of handwritten text lines in historical documents. In *Proceedings of the 3rd International Workshop on Historical Document Imaging and Processing*, pages 1–8. ACM.

[Moysset et al., 2014a] Moysset, B., Bluche, T., Knibbe, M., Benzeghiba, M. F., Messina, R., Louradour, J., and Kermorvant, C. (2014a). The A2iA multi-lingual text recognition system at the second Maurdor evaluation. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 297–302. IEEE.

[Moysset and Kermorvant, 2013] Moysset, B. and Kermorvant, C. (2013). On the evaluation of handwritten text line detection algorithms. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 185–189. IEEE.

[Moysset et al., 2017] Moysset, B., Kermorvant, C., and Wolf, C. (2017). Full-page text recognition : Learning where to start and when to stop. *Document Analysis and Recognition (ICDAR), 2017 14th International Conference on*.

[Moysset et al., 2018] Moysset, B., Kermorvant, C., and Wolf, C. (2018). Learning to detect, localize and recognize many text objects in document images from few examples. *Accepted in International Journal on Document Analysis and Recognition (IJDAR), Special issue on Deep Learning for Document Analysis and Recognition*.

[Moysset et al., 2015b] Moysset, B., Kermorvant, C., Wolf, C., and Louradour, J. (2015b). Paragraph text segmentation into lines with recurrent neural networks. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 456–460. IEEE.

[Moysset et al., 2016] Moysset, B., Louradour, J., Kermorvant, C., and Wolf, C. (2016). Learning text-line localization with shared and local regression neural networks. In *Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on*, pages 1–6. IEEE.

[Moysset et al., 2014b] Moysset, B., Messina, R., and Kermorvant, C. (2014b). A comparison of recognition strategies for printed/handwritten composite documents. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 158–163. IEEE.

[Munkres, 1957] Munkres, J. (1957). Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1) :32–38.

[Nagy et al., 1992] Nagy, G., Seth, S., and Viswanathan, M. (1992). A prototype document image analysis system for technical journals. *Computer*, 25(7) :10–22.

[Nguyen et al., 2010] Nguyen, T., Park, J., and Lee, G. (2010). Using 2d tensor voting in text detection. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 818–821. IEEE.

[Nicolaou and Gatos, 2009] Nicolaou, A. and Gatos, B. (2009). Handwritten text line segmentation by shredding text into its lines. In *Int. Conf. on Document Analysis and Recognition*.

[O'Gorman, 1993] O'Gorman, L. (1993). The document spectrum for page layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11) :1162–1173.

[Oord et al., 2016] Oord, A. v. d., Kalchbrenner, N., and Kavukcuoglu, K. (2016). Pixel recurrent neural networks. *arXiv preprint arXiv :1601.06759*.

[Pan et al., 2011] Pan, Y.-F., Hou, X., and Liu, C.-L. (2011). A hybrid approach to detect and localize texts in natural scene images. *IEEE Transactions on Image Processing*, 20(3) :800–813.

[Papageorgiou et al., 1998] Papageorgiou, C. P., Oren, M., and Poggio, T. (1998). A general framework for object detection. In *Computer vision, 1998. sixth international conference on*, pages 555–562. IEEE.

[Papavassiliou et al., 2010] Papavassiliou, V., Katsouros, V., and Carayannis, G. (2010). A morphological approach for text-line segmentation in handwritten documents. In *Frontiers in Handwriting Recognition (ICFHR), 2010 International Conference on*, pages 19–24. IEEE.

[Pham et al., 2014] Pham, V., Bluche, T., Kermorvant, C., and Louradour, J. (2014). Dropout improves recurrent neural networks for handwriting recognition. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 285–290. IEEE.

[Pinheiro et al., 2015] Pinheiro, P., Collobert, R., and Dollar, P. (2015). Learning to segment object candidates. In *Advances in Neural Information Processing System*.

[Pinheiro et al., 2016] Pinheiro, P., Lin, T., Collobert, R., and Dollar, P. (2016). Learning to refine object segments. In *European Conference on Computer Vision*.

[Pletschacher et al., 2015] Pletschacher, S., Clausner, C., and Antonacopoulos, A. (2015). Europeana newspapers ocr workflow evaluation. In *Workshop on Historical Document Imaging and Processing*.

[Rabaev et al., 2013] Rabaev, I., Biller, O., El-Sana, J., Kedem, K., and Dinstein, I. (2013). Text line detection in corrupted and damaged historical manuscripts. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 812–816. IEEE.

[Redmon et al., 2016] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once : Unified, real-time object detection. In *IEEE Conf. on Computer Vision and Pattern Recognition*.

[Ren et al., 2015] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN : Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing System*.

[Russakovsky et al., 2015] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3) :211–252.

[Ryu et al., 2014] Ryu, J., Koo, H. I., and Cho, N. I. (2014). Language-independent text-line extraction algorithm for handwritten documents. *Signal Processing Letters*, 21(9) :1115–1119.

[Saabni and El-Sana, 2011] Saabni, R. and El-Sana, J. (2011). Language-independent text lines extraction using seam carving. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 563–568. IEEE.

[Sánchez et al., 2014] Sánchez, J. A., Romero, V., Toselli, A. H., and Vidal, E. (2014). ICFHR2014 competition on handwritten text recognition on transcriptorium datasets (htrts). In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 785–790. IEEE.

[Schuster and Paliwal, 1997] Schuster, M. and Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11) :2673–2681.

[Sermanet et al., 2013] Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2013). Overfeat : Integrated recognition, localization and detection using convolutional networks. *arXiv :1312.6229*.

[Shi et al., 2017] Shi, B., Bai, X., and Belongie, S. (2017). Detecting oriented text in natural images by linking segments. *arXiv preprint arXiv :1703.06520*.

[Shi and Govindaraju, 2004] Shi, Z. and Govindaraju, V. (2004). Line separation for complex document images using fuzzy runlength. In *Document Image Analysis for Libraries, 2004. Proceedings. First International Workshop on*, pages 306–312. IEEE.

[Shi et al., 2009] Shi, Z., Setlur, S., and Govindaraju, V. (2009). A steerable directional local profile technique for extraction of handwritten arabic text lines. In *Int. Conf. on Document Analysis and Recognition*.

[Simonyan et al., 2013] Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). Deep inside convolutional networks : Visualising image classification models and saliency maps. *arXiv preprint arXiv :1312.6034*.

[Stafylakis et al., 2008] Stafylakis, T., Papavassiliou, V., Katsouros, V., and Carayannis, G. (2008). Robust text-line and word segmentation for handwritten documents images. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 3393–3396. IEEE.

[Steinhaus, 1956] Steinhaus, H. (1956). Sur la division des corp materiels en parties. *Bull. Acad. Polon. Sci*, 1(804) :801.

[Stollenga et al., 2015] Stollenga, M. F., Byeon, W., Liwicki, M., and Schmidhuber, J. (2015). Parallel multi-dimensional lstm, with application to fast biomedical volumetric image segmentation. In *Advances in Neural Information Processing Systems*, pages 2998–3006.

[Surinta et al., 2014] Surinta, O., Holtkamp, M., Karabaa, F., Van Oosten, J.-P., Schomaker, L., and Wiering, M. (2014). A path planning for line segmentation of handwritten documents. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 175–180. IEEE.

[Szegedy et al., 2015a] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015a). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

[Szegedy et al., 2015b] Szegedy, C., Reed, S., Erhan, D., and Anguelov, D. (2015b). Scalable, high-quality object detection. *arXiv :1412.1441*.

[Szegedy et al., 2013] Szegedy, C., Toshev, A., and Erhan, D. (2013). Deep neural networks for object detection. In *Advances in neural information processing systems*, pages 2553–2561.

[Theis and Bethge, 2015] Theis, L. and Bethge, M. (2015). Generative image modeling using spatial lstms. In *Advances in Neural Information Processing Systems*, pages 1927–1935.

[Tian et al., 2016] Tian, Z., Huang, W., He, T., He, P., and Qiao, Y. (2016). Detecting text in natural image with connectionist text proposal network. In *European Conference on Computer Vision*, pages 56–72. Springer.

[Tieleman and Hinton, 2012] Tieleman, T. and Hinton, G. (2012). Lecture 6.5—RmsProp : Divide the gradient by a running average of its recent magnitude. COURSERA : Neural Networks for Machine Learning.

[Tong et al., 2014] Tong, A., Przybocki, M., Margner, V., and El Abed, H. (2014). NIST 2013 open handwriting recognition and translation (Open HaRT'13) evaluation. In *Document Analysis Systems (DAS), 2014 11th IAPR International Workshop on*, pages 81–85. IEEE.

[Uijlings et al., 2013] Uijlings, J. R., Van De Sande, K. E., Gevers, T., and Smeulders, A. W. (2013). Selective search for object recognition. *International journal of computer vision*, 104(2) :154–171.

[Viola and Jones, 2004] Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International journal of computer vision*, 57(2) :137–154.

[Visin et al., 2015] Visin, F., Kastner, K., Cho, K., Matteucci, M., Courville, A., and Bengio, Y. (2015). Renet : A recurrent neural network based alternative to convolutional networks. *arXiv preprint arXiv :1505.00393*.

[Vo and Lee, 2016] Vo, Q. N. and Lee, G. (2016). Dense prediction for text line segmentation in handwritten document images. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 3264–3268. IEEE.

[Weliwitage et al., 2005] Weliwitage, C., Harvey, A., and Jennings, A. (2005). Handwritten document offline text line segmentation. In *Digital Image Computing : Techniques and Applications, 2005. DICTA'05. Proceedings 2005*, pages 27–27. IEEE.

[Wolf and Jolion, 2006] Wolf, C. and Jolion, J.-M. (2006). Object count/Area Graphs for the Evaluation of Object Detection and Segmentation Algorithms. *Int. Journ. on Document Analysis and Recognition*, 8(4) :280–296.

[Wong et al., 1982] Wong, K. Y., Casey, R. G., and Wahl, F. M. (1982). Document analysis system. *IBM journal of research and development*, 26(6) :647–656.

[Xingjian et al., 2015] Xingjian, S., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., and Woo, W.-c. (2015). Convolutional lstm network : A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810.

[Yin and Liu, 2009] Yin, F. and Liu, C.-L. (2009). Handwritten chinese text line segmentation by clustering with distance metric learning. *Pattern Recognition*, 42(12) :3146–3157.

[Yosinski et al., 2014] Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks ? In *Advances in neural information processing systems*, pages 3320–3328.

[Yosinski et al., 2015] Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., and Lipson, H. (2015). Understanding neural networks through deep visualization. *arXiv preprint arXiv :1506.06579*.

[Zahour et al., 2007] Zahour, A., Likforman-Sulem, L., Boussellaa, W., and Taconet, B. (2007). Text line segmentation of historical arabic documents. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 1, pages 138–142. IEEE.

[Zahour et al., 2004] Zahour, A., Taconet, B., and Ramdane, S. (2004). Contribution à la segmentation de textes manuscrits anciens. In *Conférence Internationale Francophone sur l'Ecrit et le Document (CIFED 04)*.

[Zeiler and Fergus, 2014] Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.

[Zhang et al., 2016] Zhang, Z., Zhang, C., Shen, W., Yao, C., Liu, W., and Bai, X. (2016). Multi-oriented text detection with fully convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4159–4167.

[Zheng et al., 2015] Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., and Torr, P. H. (2015). Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1529–1537.

[Zitnick and Dollár, 2014] Zitnick, C. L. and Dollár, P. (2014). Edge boxes : Locating object proposals from edges. In *European Conference on Computer Vision*, pages 391–405. Springer.