Université de Montréal

Détection de tableaux dans des documents: une étude de TableBank

par

Eugénie Yockell

Département de mathématiques et de statistique
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en Informatique

28 avril 2023

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé

Détection de tableaux dans des documents: une étude de TableBank

présenté par

Eugénie Yockell

a été évalué par un jury composé des personnes suivantes :

Sébastien Roy

(président-rapporteur)

Philippe Langlais

(directeur de recherche)

Jian-Yun Nie

(membre du jury)

Résumé

L'extraction d'information dans des documents est une nécessité, particulièrement dans notre ère actuelle où il est commun d'employer un téléphone portable pour photographier des documents ou des factures. On trouve aussi une utilisation répandue de documents PDF qui nécessite de traiter une imposante quantité de documents digitaux. Par leur nature, les données des documents PDF sont complexes à extraire, nécessitant d'être analysés comme des images. Dans cette recherche, on se concentre sur une information particulière à prélever: des tableaux. En effet, les tableaux retrouvés dans les documents représentent une entité significative, car ils contiennent des informations décisives. L'utilisation de modèles neuronaux pour performer des extractions automatiques permet considérablement d'économiser du temps et des efforts.

Dans ce mémoire, on définit les métriques, les modèles et les ensembles de données utilisés pour la tâche de détection de tableaux. On se concentre notamment sur l'étude des ensembles de données TableBank et PubLayNet, en soulignant les problèmes d'annotations présents dans l'ensemble TableBank. On relève que différentes combinaisons d'ensembles d'entraînement avec TableBank et PubLayNet semblent améliorer les performances du modèle Faster R-CNN, ainsi que des méthodes d'augmentations de données. On compare aussi le modèle de Faster R-CNN avec le modèle CascadeTabNet pour la détection de tableaux où ce premier demeure supérieur.

D'autre part, on soulève un enjeu qui est peu discuté dans la tâche de détection d'objets, soit qu'il existe une trop grande quantité de métriques. Cette problématique rend la comparaison de modèles ardue. On génère ainsi les résultats de modèles selon plusieurs métriques afin de démontrer qu'elles conduisent généralement vers différents modèles gagnants, soit le modèle ayant les meilleures performances. On recommande aussi les métriques les plus pertinentes à observer pour la détection de tableaux, c'est-à-dire AP_{medium}/AP_{medium}, *Pascal AP85* ou *COCO AP85* et la métrique de TableBank.

Mots-clés : Détection d'objets, Détection de tableaux, Faster R-CNN, ResNeXt, CascadeTabNet, TableBank, PubLayNet

Abstract

Extracting information from documents is a necessity, especially in today's age where it is common to use a cell phone to photograph documents or invoices. There is also the widespread use of PDF documents that requires processing a large amount of digital documents. Due to their nature, the data in PDF documents are complex to retrieve, needing to be analyzed as images. In this research, we focus on a particular information to be extracted: tables. Indeed, the tables found in documents represent a significant entity, as they contain decisive information. The use of neural networks to perform automatic retrieval saves time and effort.

In this research, the metrics, models and datasets used for the table detection task are defined. In particular, we focus on the study of the TableBank and PubLayNet datasets, highlighting the problems of annotations present in the TableBank set. We point out that different combinations of training sets using TableBank and PubLayNet appear to improve the performance of the Faster R-CNN model, as well as data augmentation methods. We also compare the Faster R-CNN model with the CascadeTabNet model for table detection where the former remains superior.

In addition, we raise an issue that is not often discussed in the object detection task, namely that there are too many metrics. This problem makes model comparison difficult. We therefore generate results from models with several metrics in order to demonstrate the influence of these metrics in defining the best performing model. We also recommend the most relevant metrics to observe for table detection, AP_{medium}/AP_{medium}, *Pascal AP85* or *COCO AP85* and the TableBank metric.

Keywords : Object detection, Table Detection, Faster R-CNN, ResNeXt, CascadeTabNet, TableBank, PubLayNet

Table des matières

Liste des tableaux

Liste des figures

Liste des sigles et des abréviations

AP	Average precision
AR	Average recall
IA	Intelligence artificiel
IoU	Intersection over Union
mAP	Mean Average Precision
NLP	Natural language processing

Remerciements

Tout d'abord, je remercie mon superviseur de recherche, Philippe Langlais, pour sa patience et son support au cours de ce mémoire.
Je remercie également les membres du RALI pour leur accompagnement.
Je remercie aussi l'équipe de LexRock AI pour une collaboration fructueuse.
Finalement, je tiens à exprimer ma plus profonde gratitude envers mes parents pour leur soutien, leur bienveillance et leur générosité.

Chapitre 1

Introduction

1.1. Mise en contexte

L'intelligence artificielle (IA) est un domaine de l'informatique qui vise à automatiser des tâches complexes qui nécessiteraient autrement une intervention humaine. «Intelligence» invoque l'objectif de mimer les comportements humains, tandis que «artificielle» réfère à l'usage d'ordinateur et processus informatique. Les procédés souvent employés sont la compréhension des langues naturelles, l'analyse de sons comme les langues parlées ou la décomposition d'images et vidéos nommée la vision par ordinateur. Cette dernière est une branche de l'IA qui dénomme un ensemble de méthodes pour permettre aux ordinateurs d'obtenir une compréhension sur les éléments composant des images ou des vidéos et ainsi automatiser une compréhension sur les images et d'en extraire les informations.

Une tâche commune de la vision par ordinateur est la détection d'objets qui est une méthode permettant de détecter les entités visuelles (voitures, animaux, personnes, etc.) qui composent une image numérique afin d'en déterminer la nature. Cette approche nécessite souvent de cibler la position des instances recherchées. Ce mémoire est centré sur la vision par ordinateur. On s'intéresse plus particulièrement à l'analyse de textes numériques. En effet, la croissance des technologies a engendré une importante quantité de documents digitaux. L'extraction d'informations dans ces documents est presque impossible pour des humains à cause du monumental temps requis et des énormes coûts associés. Plusieurs domaines tels que les secteurs financier, juridique, d'assurance ou politique ont un intérêt notable à vouloir rapidement et efficacement extraire des informations spécifiques de leurs imposantes quantités de documents digitaux.

Pour notre part, on se concentre sur une information particulière à prélever: des tableaux. Les documents électroniques contiennent fréquemment des tableaux de toutes sortes pour résumer des données importantes. Posséder un outil capable de déterminer la position de tableaux dans ces documents est d'une utilité considérable. Il est à noter

que les grosses compagnies comme Google et Microsoft ont leur propre outil d'extraction d'information demeurant confidentiel. Ainsi, il est important pour la communauté scientifique d'avoir une bonne compréhension de nos outils afin de partager les connaissances et optimiser les modèles. Par conséquent, plusieurs scientifiques ont œuvré à développer des systèmes capables de déterminer la position de tableaux numériques dans un document. Toutefois, détecter la position des tableaux est une tâche ardue. Ils sont très variés dans leurs apparences et peuvent être confondus avec des graphiques, organigrammes ou de simples lignes horizontales et verticales. Il est ainsi nécessaire d'avoir des outils robustes et génériques qui incorporent cette diversité. C'est dans ces conditions que l'intelligence artificielle assume son rôle.

1.2. Méthodologie

Essentiellement, dans ce mémoire, on étudie principalement les réseaux de neurones pour résoudre la problématique. Ceux-ci ont fait des avancées extraordinaires dans les dernières années et sont incontournables dans les tâches de détection d'objets. Les réseaux de neurones artificiels sont composés d'une collection de neurones interconnectés et rangés en couche où chaque neurone se voit accordé un poids afin de former une nouvelle représentation de l'image passée en entrée, comme on peut le voir à la figure 1.1 où on classifie si l'image contient un chien, un oiseau ou un chat. Les poids de chaque neurone sont établis lors d'un entraînement de manière à ce que le modèle apprend à reconnaître les instances étudiées. Un ensemble de données d'entraînement rend possible cet apprentissage. Effectivement, l'ensemble de données d'entraînement contient des images ainsi que les informations véritables contenues. Si l'on continue avec l'exemple de la figure, un ensemble de données d'entraînement contiendrait des images variées contenant des chiens, des oiseaux et des chats, ainsi que les étiquettes, soit les classes véridiques des images (chien, oiseaux, chat). Le réseau de neurones traite l'image d'entraînement et retourne une prédiction, la classe, en sortie. Étant donné que la véritable classe est contenue dans les labels, les poids des neurones sont mis à jour afin de se rapprocher de la véritable classe. En passant ainsi la totalité des images de l'ensemble d'entraînement, les neurones ont des poids calibrés pour répondre à la tâche. Par conséquent, il est important d'avoir un grand ensemble de données d'entraînement diversifié afin de produire un réseau de neurones robuste aux généralités.

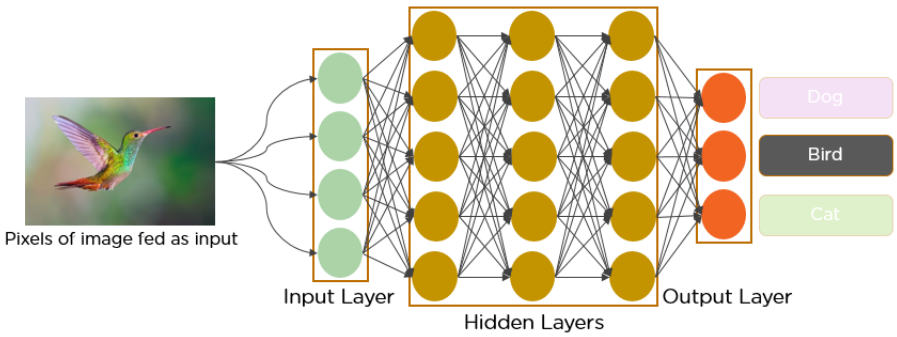


Fig. 1.1. Exemple de [**Mandal, 2021**] d'une image passée à un réseau de neurones afin de former une nouvelle représentation de l'image et d'en extraire les informations recherchées. On cherche ici à classifier si l'image contient un chien, un oiseau ou un chat.

Dans notre cas, on cherche à localiser des tableaux dans des documents textuels qui sont visualisés comme des images. Comme mentionné, le principe est de passer les images qu'on veut étudier à des modèles comme des réseaux de neurones profonds ou de simples algorithmes qui retournent des hypothèses de détection des objets étudiés. Ces propositions de détection, aussi nommées prédictions, contiennent la classe de l'objet ainsi que sa position dans l'image. Dans notre cas, la classe de l'objet est toujours un tableau. Par contre, les modèles peuvent retourner des hypothèses erronées, il est alors impératif de mesurer la qualité des prédictions à l'aide de métriques qui sont un ensemble de méthodes pour évaluer la performance.

À la figure 1.2 on présente un exemple d'une image avec une détection d'un tableau dans un document. Les images sont contenues dans les axes x et y, où l'origine en y est au coin gauche en haut et l'origine en x est au coin gauche en bas. Sur la figure, le tableau prédit est contenu dans le rectangle vert que l'on nomme boîte de détection. Les boîtes sont définies en fournissant les coordonnées $[x_1, y_1, x_2, y_2]$ où (x_1,y_1) définie un extrémum de la boîte spécifié par le rond bleu sur la figure et (x_2,y_2) définie un extrémum de la boîte spécifié par le rond rouge. Les boîtes peuvent aussi être définies par les coordonnées $[x,y,w,h]$ où (x,y) définie les coordonnées d'un extrémum comme représenté dans le rond bleu de la figure, tandis que (w,h) représente la largeur $(width)$ et la hauteur $(height)$ de la boîte de détection.

Afin d'évaluer la qualité d'un modèle de détection d'objets, il est nécessaire d'avoir des ensembles des données de test qui servent de références d'évaluation (*benchmark*). Ceux-ci sont composés de plusieurs images avec les coordonnées des boîtes qui encadrent l'objet étudié. Une image d'un tel ensemble de données ressemble à l'exemple de la figure 1.2, soit un document avec la boîte de label représentée par le rectangle vert. La boîte de label identifie la position réelle du tableau dans l'image. L'information de la boîte de label est aussi représentée par les coordonnées $[x_1, y_1, x_2, y_2]$ ou les coordonnées $[x,y,w,h]$.

Ainsi, on possède les éléments essentiels à la tâche de détection d'objet, soit des images de documents avec les boîtes de labels (la position réelle) et un modèle qui retourne les boîtes de détection (la position hypothétique). Les métriques se basent sur les coordonnées de ces deux types boîtes de détection et de label. On veut mesurer à quel point les deux boîtes se ressemblent en termes de position et dimension. Il existe plusieurs façons de mesurer la correspondance de ces boîtes et par conséquent une grande quantité de métriques.

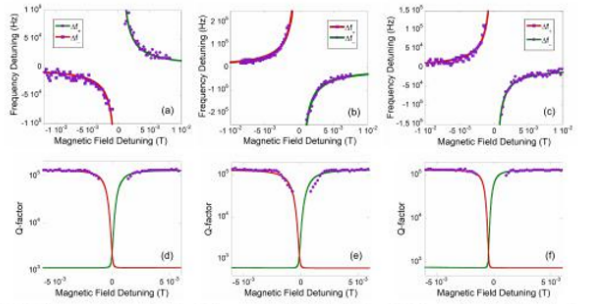


FIG. 7: Frequency shift as a function of the magnetic field: (a) A, (b) B and (c) C normal mode splittings due the interactions between the spins ensembles and the 15.18 GHz WGM. Δf_+ and Δf_- correspond to the two coupled modes; (d), (e) and (f) shows Q-factors of the coupled modes as a function of the magnetic field and correspond to the interactions A, B and C, respectively. The magnetic field zero-detunings are 0.516 T in (a) and (d), 0.88 T in (b) and (e) and 0.169 T in (c) and (f).

TABLE II: ESR data: spin-photon interactions of the 15.18 GHz WGM shown in Figures 6 and 7.

	B (Tesla)	$\delta_{spins}/2$ (MHz)	$\Gamma_{WGM}/2$ (MHz)	g (MHz)	Q_{spins}	Q_{WGM}
A	0.516	6.7	0.058	2.1	1130	1.3×10^5
B	0.088	12.5	0.058	4.2	606	1.3×10^5
C	0.169	10.3	0.058	3.4	740	1.3×10^5

to the fact that the coupling between the resonator photons and spin ensemble is smaller than the inhomogeneous spin linewidth, means that the effective Q-factor of the spin ensemble dominates, and the transmission spectra at the point of the mode crossing is severely degraded and becomes an inaccurate technique to characterise the system due to the very poor signal to noise ratio. This technique is based on a two-mode coupled harmonic oscillators model, an older paper shows how the parameters derived from reflection coefficient or transmission spectra of a coupled mode system can also be successfully obtained by solving the characteristic eigenvalue equations[34,35]. This is a very good technique in this limit, as we can use the full range of the magnetic field and the normal mode frequency and linewidth (or effective Q-factor) variations with sufficient signal-to-noise ratio to fit the relevant parameters during the interaction. Thus, to determine the properties of the 15.18 GHz WGM interacting with the measured spins ensembles, the ALCs were fitted using a similar model.

This model assumes two representative LCR oscillators (labeled with subscripts 1 and 2) coupled reactively through the magnetic field (mutual inductance) with a dimensionless coupling coefficient of $2\Delta_g$[34,35]. The calculated characteristic equation for the coupled mode system is written in Eq. 2, ignoring all terms of order $1/Q^2$. This is a very good approximation for any system of relatively high Q-factor (i.e. above 10).

$$\omega^4 + \omega^3\left(\frac{\omega_1}{Q_1} + \frac{\omega_2}{Q_2}\right) + \omega^2\left(\omega_1^2 + \omega_2^2\right) + \omega\left(\frac{\omega_2\omega_1^2}{Q_2} + \frac{\omega_2^2\omega_1}{Q_1}\right) + \omega_1^2\omega_2^2 - 2\Delta_g\omega_1^2\omega_2^2 \quad (2)$$

Fig. 1.2. Exemple d'une image d'un ensemble de données de détection de tableaux dans un document où le tableau est encadré par le rectangle vert.

1.3. Motivation

Au cours de la recherche effectuée, une problématique est devenue claire: il existe un trop grand nombre d'ensembles de tests et de métriques. En effet, les différents articles récents sur la détection de tableaux se concentrent majoritairement à créer des ensembles de données d'entraînement généreux dans la quantité et la diversité, et des réseaux de neurones performants en termes de temps et de qualité. Le souci est que chacun d'eux utilise différents ensembles de test et différentes métriques. Il existe ainsi peu de façons de comparer les performances des modèles à l'aide de références d'évaluation ou de métriques fiables.

L'univers de la détection d'objets comporte plusieurs compétitions qui sont chacune associée à des métriques et références d'évaluation spécifiques. De plus, les ensembles de

données vont rarement être représentés dans le même format. Le nom des images et les labels associés sont contenus dans différents types de formats qui vont représenter les données de manière unique. Par exemple, deux formats populaires sont le format COCO et le format Pascal VOC. Le premier encapsule les informations dans des documents *JSON*, tandis que le dernier présente les informations dans des documents *XML*. La différence entre ces deux formats réside dans le fait que les données ne contiennent pas les mêmes informations. Un exemple de ces deux formats de données est présenté à la figure 1.3. Aussi, comme mentionné plus haut, les boîtes de détection peuvent être définies par des coordonnées $[x_1, y_1, x_2, y_2]$ représentant les extremums du rectangle de détection. Par contre, d'autres formats définissent les boîtes de détection en utilisant les coordonnées du centre de la boîte ou en utilisant la largeur et la longueur de la boîte. Certains scientifiques vont avoir tendance à reporter seulement les métriques compatibles avec le format de données utilisé. En résumé, cette disparité se distingue dans tout le domaine de détection d'objets qui ne peut être que ralenti par ce manque d'homogénéité.

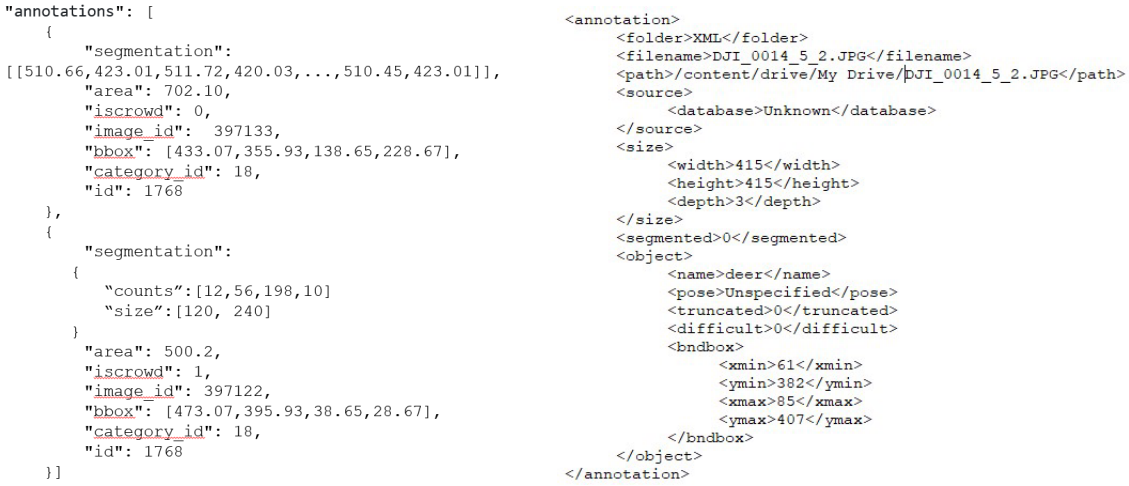


Fig. 1.3. Exemple du format COCO [**Khandelwal, 2021**] à gauche et du format Pascal VOC [**Rančić et al., 2023**] à droite. On remarque que les attributs ne sont pas définis de la même manière et que certains sont uniques à un format.

Ce conflit a été soulevé par quelques scientifiques assez récemment. Notamment, [**Padilla et al., 2021a**] mentionne l'importance d'avoir un consensus dans les métriques et leurs implémentations, ainsi que dans les ensembles de données. Les auteurs créés un outil contenant une grande quantité de métriques et de formats de données afin de centraliser l'évaluation. En effet, ils remarquent que des scientifiques voulant évaluer leurs modèles avec certains ensembles de tests devront eux-mêmes implémenter les métriques afin de les rendre compatibles, ce qui requière un temps considérable et peut causer des variations ou des biais dans les résultats. Cette conséquence est ressortie dans le mémoire. De plus, il existe une grande disparité dans la manière dont les ensembles de données sont définis, ce qui complique l'usage de plusieurs ensembles de données pour

les entraînements et tests. L'outil de [**Padilla et al., 2021a**] sera discuté plus en détail au cours du mémoire. Ce manque d'uniformité dans les références d'évaluation est aussi pointé dans [**Borchmann et al., 2021**]. Pour pallier ce problème, les auteurs proposent DUE (Document Understanding Evaluation), un système d'évaluation rassemblant des ensembles de données de test pour les tâches de compréhension de documents comme détection de question/réponse visuelle, extraction d'information, détection de tableaux, graphiques, listes, etc.

Afin d'explorer cet enjeu, on commence par introduire au chapitre 2 les différentes métriques existantes pour la détection d'objets. Au chapitre 3 on définit les modèles étudiés pour la détection des tableaux. Tandis qu'au chapitre 4 on définit les ensembles de données les plus pertinents pour la détection de tableaux. Au chapitre 5, on se concentre sur la reproduction des résultats des auteurs de TableBank [**Li et al., 2019b**], afin de trouver des améliorations possibles aux modèles utilisés, soit Faster R-CNN, en utilisant différentes combinaisons d'ensembles d'entraînements et des techniques d'augmentations de données. On souligne aussi des erreurs d'annotations contenues dans l'ensemble TableBank. L'étude de modèles et la recherche d'améliorations nous amène au chapitre 6 suivant, qui décrit l'impact et l'importance des métriques. On compare plusieurs métriques afin d'exposer leurs valeurs et les différents résultats qu'elles peuvent engendrer. On complète en pointant les métriques que l'on juge les plus appropriées pour la tâche de détection de tableaux.

Chapitre 2

Métriques

Les métriques d'évaluation permettent d'estimer la performance des modèles d'apprentissage automatique. Elles mesurent la ressemblance des prédictions faites par les modèles aux labels, soit les vraies valeurs. C'est un outil essentiel dans un domaine où les recherches académiques et commerciales sont majoritairement guidées par les performances sur des jeux de données. En effet, certaines métriques peuvent parfois mettre en valeur ou négliger certaines caractéristiques permettant ainsi d'amplifier les résultats. Donc, dans ce chapitre, on présente les multiples métriques existantes pour la détection d'objets dans une image. Par la suite, dans le chapitre 6, on étudie plus profondément la pertinence de ces métriques et les problèmes créés par l'existence de ce nombre élevé de métriques.

On rappelle que les boîtes de détection et de label sont définis par les coordonnées [x,y,w,h] où (x,y) définie les coordonnées d'un extrémum, tandis que (w,h) représente la largeur (*width*) et la hauteur (*height*) de la boîte. La boîte de détection j de l'image i est définie comme $D_{ij} = [x_{ij}, y_{ij}, w_{ij}, h_{ij}]$ et l'aire de cette boîte est $A(D_{ij}) = w_{ij}$ x h_{ij}. La boîte de label j de l'image i est définie comme $L_{ij} = [x_{ij}, y_{ij}, w_{ij}, h_{ij}]$ et l'aire de cette boîte est $A(L_{ij}) = w_{ij}$ x h_{ij}.

2.1. Métrique TableBank

Cette métrique a été introduite dans [**Gilani et al., 2017**] spécifiquement pour la détection de tableaux dans des documents. C'est la métrique utilisée par les auteurs de TableBank [**Li et al., 2019b**], un ensemble de données décrit à la section 4.4. La métrique utilise l'espace de chevauchement de la boîte de prédiction et de la boîte de label comme représenté à la figure 2.1 où l'espace en vert représente l'espace de chevauchement.

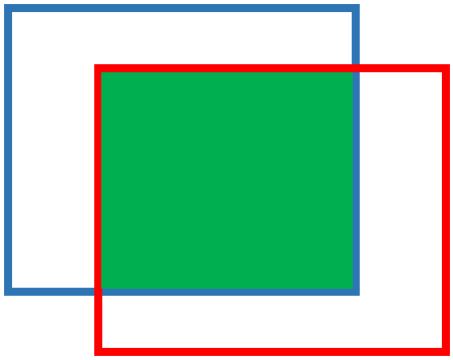


Fig. 2.1. Exemple d'un chevauchement d'une boîte de label (en bleu) et d'une boîte prédiction (en rouge). On représente le chevauchement des deux boîtes en vert.

La précision (*precision*) et le rappel (*recall*) sont:

$$\begin{aligned} \textit{Précision} &= \frac{\sum_{i=1}^{n}\sum_{j=1}^{m_i} A(D_{ij} \cap L_{ij})}{\sum_{i=1}^{n}\sum_{j=1}^{m_i} A(D_{ij})} \\ &= \frac{\text{Aire d'intersection des boîtes détections et des boîtes de labels}}{\text{Aire de toutes les boîtes de détections}} \end{aligned}$$

$$\begin{aligned} \textit{Rappel} &= \frac{\sum_{i=1}^{n}\sum_{j=1}^{m_i} A(D_{ij} \cap L_{ij})}{\sum_{i=1}^{n}\sum_{j=1}^{m_i} A(L_{ij})} \\ &= \frac{\text{Aire d'intersection des boîtes détections et des boîtes de labels}}{\text{Aire de toutes les boîtes des labels}} \end{aligned}$$

$$\textit{F1 Score} = \frac{2 \times \text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

où m_i représente le nombre de boîtes dans l'image i et n représente le nombre total d'images. La précision trouve la proportion des tableaux détectés qui appartiennent à un label de l'image. Tandis que le rappel évalue la proportion des labels associés à une détection. Cette métrique permet d'obtenir un ratio d'intersection entre les prédictions et les labels. Il est à noter que cette métrique ne retourne presque jamais une prédiction parfaite puisque chaque écart est pris en compte.

2.2. IoU

IoU (*Intersection over Union*) est une métrique qui mesure à quel point la boîte de détection chevauche la boîte de label. Une visualisation est présentée à la figure 2.2.

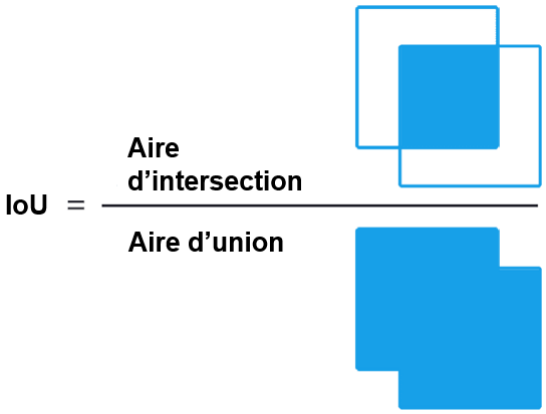


Fig. 2.2. Visualisation de la métrique d'IoU où on mesure le ratio de l'aire d'intersection et d'union d'une boîte de détection et d'une boîte de label.

On mesure l'IoU comme:

$$\begin{aligned} IoU &= \frac{A(D_{ij} \cap L_{ij})}{A(D_{ij} \cup L_{ij})} \\ &= \frac{\text{Aire d'intersection de la boîte détection et de la boîte de label}}{\text{Aire d'union de la boîte détection et de la boîte de label}} \end{aligned}$$

où $0 \leq IoU \leq 1$ et $IoU = 1$ représente une prédiction qui superpose parfaitement le label.

Pour mesurer la qualité d'une détection, il est commun d'utiliser un seuil sur l'IoU pour associer une détection à un label et déterminer si on considère la région comme détectée correctement. On peut ensuite mesurer la précision et le rappel en utilisant les prédictions qui ont une valeur d'IoU supérieur à un seuil:

$$\begin{aligned} \textit{Précision} &= \frac{\text{nombre de détections}_{IoU \geq i}}{\text{nombre de détections}} \\ \textit{Rappel} &= \frac{\text{nombre de détections}_{IoU \geq i}}{\text{nombre de labels}} \end{aligned}$$

où i représente le seuil de IoU désiré. Il est commun de mesurer la précision et le rappel avec différents IoU, souvent 0.6, 0.7, 0.8 et 0.9. C'est la métrique utilisée dans la compétition ICDAR2013 et ICDAR2019 décrites aux sections 4.1 et 4.2. On peut avoir une prédiction parfaite puisque qu'un petit écart peut être considéré comme insignifiant selon le seuil.

2.3. Weighted-average F1

La métrique de *weighted-average F1* est une métrique où on assigne un poids pour chaque précision et rappel selon le seuil d'IoU. Le poids est défini en utilisant la valeur

du IoU. Il est proposé que les seuils d'IoU plus élevés doivent se voir accorder une plus grande importance, c'est-à-dire, avoir un plus grand poids. Cette hypothèse est logique puisqu'un IoU plus grand sous-entend une meilleure prédiction. Par exemple, les auteurs de CascadeTabNet [**Prasad et al., 2020b**], un modèle décrit à la section 3.4, utilisent le IoU comme métrique avec différents seuils de 0.6, 0.7, 0.8 et 0.9 et le *weighted-average F1.*

$$\text{W Avg F1} = \frac{\sum_i IoU_i \cdot F1@IoU_i}{\sum_i IoU_i}$$

où i est l'ensemble des différents seuils d'IoU utilisés.

2.4. Complétude et pureté

Les métriques de complétude (*completeness*) et pureté (*purity*) sont introduites dans [**Costa e Silva, 2011**]. Dans la détection d'objets dans des documents, deux cas problématiques se produisent. Le premier cas est d'avoir des détections impures, parce que d'autres éléments sont inclus dans la détection, et le deuxième cas est d'avoir des détections incomplètes, parce que l'objet n'est pas complétement détecté. Une détection parfaite est alors complète et pure. Des exemples de détection impure et incomplète sont respectivement présentés à la figure 2.3. La complétude et pureté sont définies ci-dessous:

$$\textit{Complétude} = \frac{\text{nombre d'éléments complétement identifiés}}{\text{nombre de labels}}$$
$$\textit{Pureté} = \frac{\text{nombre d'éléments purement identifiés}}{\text{nombre de détections}}$$

Dans ICDAR2013, un ensemble de données décrit à la section 4.1, [**Göbel et al., 2013**] notent qu'un problème de cette mesure est qu'elle ne fait pas de différence entre une grande et une petite partie manquante sur une détection. Ils ont alors jugé qu'il était nécessaire d'ajouter la précision et le rappel pour mesurer la qualité d'une détection, car la complétude ne va pas toujours bien représenter la qualité. Par exemple, les auteurs de ICDAR2013 ont remarqué qu'il est possible d'avoir un bon score F1 avec une mauvaise complétude, car aucune détection n'englobe parfaitement toute une table.

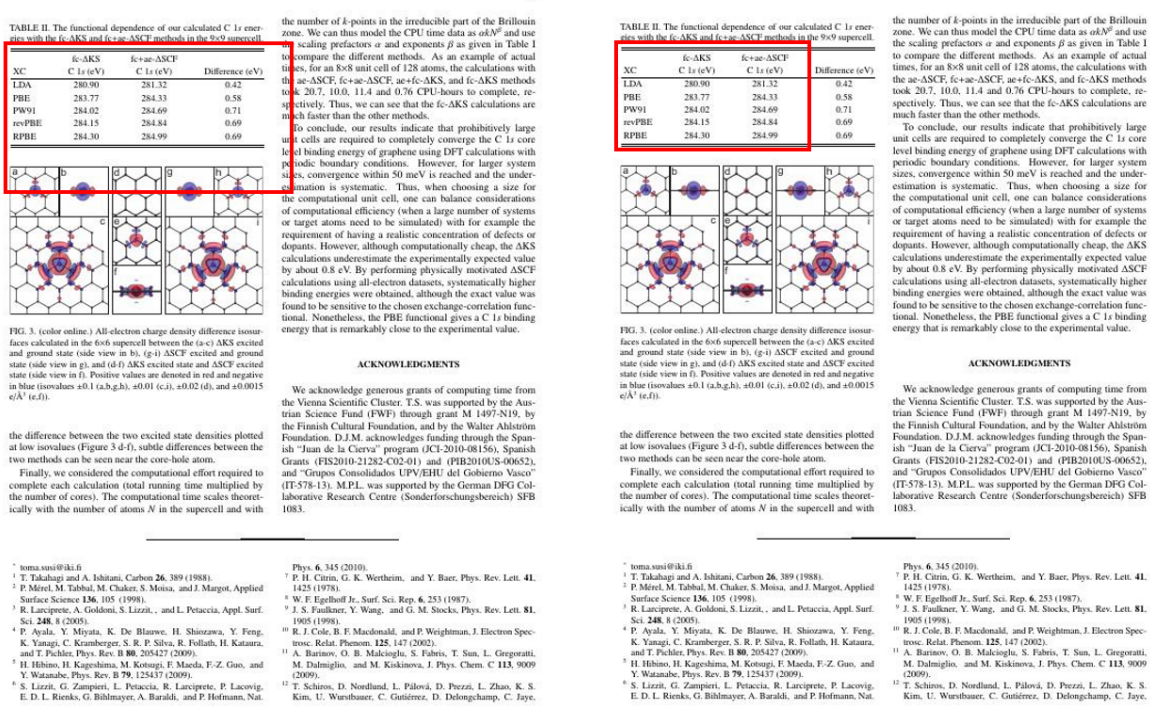


TABLE II. The functional dependence of our calculated C 1*s* energies with the fc-ΔKS and fc+ae-ΔSCF methods in the 9×9 supercell.

XC	fc-ΔKS C 1*s* (eV)	fc+ae-ΔSCF C 1*s* (eV)	Difference (eV)
LDA	280.90	281.32	0.42
PBE	283.77	284.33	0.58
PW91	284.02	284.69	0.71
revPBE	284.15	284.84	0.69
RPBE	284.30	284.99	0.69

FIG. 3. (color online.) All-electron charge density difference isosurfaces calculated in the 6×6 supercell between the (a-c) ΔKS excited and ground state (side view in b), (g-i) ΔSCF excited and ground state (side view in g), and (d-f) ΔKS excited state and ΔSCF excited state (side view in f). Positive values are denoted in red and negative in blue (isovalues ±0.1 (a,b,g,h), ±0.01 (c,i), ±0.02 (d), and ±0.0015 e/Å^3 (e,f)).

the difference between the two excited state densities plotted at low isovalues (Figure 3 d-f), subtle differences between the two methods can be seen near the core-hole atom.

Finally, we considered the computational effort required to complete each calculation (total running time multiplied by the number of cores). The computational time scales theoretically with the number of atoms N in the supercell and with the number of k-points in the irreducible part of the Brillouin zone. We can thus model the CPU time data as αkN^β and use the scaling prefactors α and exponents β as given in Table I to compare the different methods. As an example of actual times, for an 8×8 unit cell of 128 atoms, the calculations with the ae-ΔSCF, fc+ae-ΔSCF, ae+fc-ΔKS, and fc-ΔKS methods took 20.7, 10.0, 11.4 and 0.76 CPU-hours to complete, respectively. Thus, we can see that the fc-ΔKS calculations are much faster than the other methods.

To conclude, our results indicate that prohibitively large unit cells are required to completely converge the C 1*s* core level binding energy of graphene using DFT calculations with periodic boundary conditions. However, for larger system sizes, convergence within 50 meV is reached and the underestimation is systematic. Thus, when choosing a size for the computational unit cell, one can balance considerations of computational efficiency (when a large number of systems or target atoms need to be simulated) with for example the requirement of having a realistic concentration of defects or dopants. However, although computationally cheap, the ΔKS calculations underestimate the experimentally expected value by about 0.8 eV. By performing physically motivated ΔSCF calculations using all-electron datasets, systematically higher binding energies were obtained, although the exact value was found to be sensitive to the chosen exchange-correlation functional. Nonetheless, the PBE functional gives a C 1*s* binding energy that is remarkably close to the experimental value.

ACKNOWLEDGMENTS

We acknowledge generous grants of computing time from the Vienna Scientific Cluster. T.S. was supported by the Austrian Science Fund (FWF) through grant M 1497-N19, by the Finnish Cultural Foundation, and by the Walter Ahlström Foundation. D.J.M. acknowledges funding through the Spanish "Juan de la Cierva" program (JCI-2010-08156), Spanish Grants (FIS2010-21282-C02-01) and (PIB2010US-00652), and "Grupos Consolidados UPV/EHU del Gobierno Vasco" (IT-578-13). M.P.L. was supported by the German DFG Collaborative Research Centre (Sonderforschungsbereich) SFB 1083.

* toma.susi@iki.fi

[1] T. Takahagi and A. Ishitani, Carbon **26**, 389 (1988).
[2] P. Mérel, M. Tabbal, M. Chaker, S. Moisa, and J. Margot, Applied Surface Science **136**, 105 (1998).
[3] R. Larciprete, A. Goldoni, S. Lizzit, , and L. Petaccia, Appl. Surf. Sci. **248**, 8 (2005).
[4] P. Ayala, Y. Miyata, K. De Blauwe, H. Shiozawa, Y. Feng, K. Yanagi, C. Kramberger, S. R. P. Silva, R. Follath, H. Kataura, and T. Pichler, Phys. Rev. B **80**, 205427 (2009).
[5] H. Hibino, H. Kageshima, M. Kotsugi, F. Maeda, F.-Z. Guo, and Y. Watanabe, Phys. Rev. B **79**, 125437 (2009).
[6] S. Lizzit, G. Zampieri, L. Petaccia, R. Larciprete, P. Lacovig, E. D. L. Rienks, G. Bihlmayer, A. Baraldi, and P. Hofmann, Nat. Phys. **6**, 345 (2010).
[7] P. H. Citrin, G. K. Wertheim, and Y. Baer, Phys. Rev. Lett. **41**, 1425 (1978).
[8] W. F. Egelhoff Jr., Surf. Sci. Rep. **6**, 253 (1987).
[9] J. S. Faulkner, Y. Wang, and G. M. Stocks, Phys. Rev. Lett. **81**, 1905 (1998).
[10] R. J. Cole, B. F. Macdonald, and P. Weightman, J. Electron Spectrosc. Relat. Phenom. **125**, 147 (2002).
[11] A. Barinov, O. B. Malcioglu, S. Fabris, T. Sun, L. Gregoratti, M. Dalmiglio, and M. Kiskinova, J. Phys. Chem. C **113**, 9009 (2009).
[12] T. Schiros, D. Nordlund, L. Pálová, D. Prezzi, L. Zhao, K. S. Kim, U. Wurstbauer, C. Gutiérrez, D. Delongchamp, C. Jaye,

Fig. 2.3. À gauche, un exemple d'une détection impure d'un tableau, car d'autres éléments sont inclus dans la boîte de détection définie par le rectangle rouge. À droite, un exemple d'une détection incomplète d'un tableau, car il manque des éléments au tableau dans la détection définie par le rectangle rouge.

2.5. AP et AR

AP (*Average Precision*), mAP (*mean average precision*) et AR (*Average Recall*) sont les métriques les plus souvent utilisées pour évaluer les modèles de détection d'objets. L'AP représente l'aire sous la courbe de précision-rappel qui consiste à tracer la précision en fonction du rappel selon différents seuils de confiance des prédictions. En effet, les modèles produisant les prédictions incluent un taux de confiance pour chaque détection, plus le taux est haut, plus les détections sont bonnes. Le seuil utilisé pour déterminer si une détection est valide impacte le score de précision et de rappel. Un seuil très élevé est associé à une haute précision, car chaque détection est très bonne. Par contre, on risque d'ignorer des prédictions valides, et ainsi avoir un grand nombre de faux négatifs, et conséquemment un faible rappel. Un seuil trop bas engendre l'effet contraire. Un exemple d'une courbe précision-rappel est présenté à la figure 2.4.

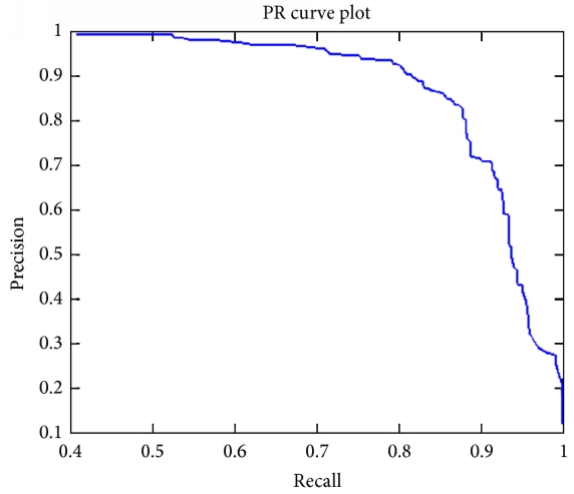


Fig. 2.4. Exemple d'une courbe de précision rappel.

Les métriques de AP peuvent être calculées de différentes façons. Le rappel est une fonction décroissante selon le seuil de confiance, tandis que la précision tend aussi souvent à décroitre, mais une précision qui reste haute cause une grande aire sous la courbe (area under curve AUC) et ainsi une haute valeur de AP. Calculer l'air sous la courbe peut s'avérer difficile, surtout parce qu'elle prend souvent des formes non conventionnelles comme en zig-zag. La méthode consiste à prendre les paires de points de précision $P(\tau(k))$ et de rappel $R(\tau(k))$ où $\tau(k)$ représentent les seuils de confiance ordonnés tel,

$$\tau(k) \text{ où } k = 1,2,...,K \text{ et } \tau(i) > \tau(j) \text{ lorsque } i > j \tag{2.5.1}$$

De cette façon $P(\tau(k))$ est la précision au seuil de confiance $\tau(k)$ et $R(\tau(k))$ est le rappel au seuil de confiance $\tau(k)$. Les paires de valeurs de précision et rappel doivent être interpolées afin que la courbe soit monotone. On obtient ainsi une fonction continue $P_{interp}(R)$ où $R \in [0,1]$ définie comme:

$$P_{interp}(R) = \max_{k|R(\tau(k))\geq R} \{P(\tau(k))\}$$

On a ainsi la précision interpolée au rappel R qui est la valeur maximale de la précision $P_{interp}(k)$ dont la valeur de rappel correspondante est celle plus grande ou égale à R. Par la suite, on peut mesurer la précision moyenne (AP) en échantillonnant $P_{interp}(R)$ et en utilisant l'intégrale de Riemann sur $P_{interp}(R)$ pour chaque K valeur de rappel $R(\tau(k))$, soit

$$AP = \sum_{k=0}^{K} \Big(R(k) - R(k+1)\Big)\Big(P_{interp}(R(k))\Big) \tag{2.5.2}$$

Il y a deux méthodes pour calculer l'intégrale de Riemann. Par une interpolation avec N points et une interpolation avec tous les points.

L'interpolation avec N points consiste à définir l'ensemble des valeurs de rappel $R(n)$ dans 2.5.2 comme un ensemble de points identiquement espacés:

$$R(n) = \frac{N-n}{N-1} \text{ où } n = 1,2,...,N \tag{2.5.3}$$

Ainsi, on définit l'aire sous la courbe comme:

$$AP = \frac{1}{N} \sum_{n=1}^{N} P_{interp}(R(n)) \tag{2.5.4}$$

Les différentes manières de mesurer le AP proviennent du nombre de points utilisés pour l'interpolation. Généralement, on utilise $N = 11$, $N = 101$ ou tous les points. L'interpolation avec tous les points se fait en utilisant toutes les valeurs de rappel produites par tous les seuils de confiance K possible, soit les valeurs de $\tau(0) = 0$ à $\tau(K+1) = 1$. On définit les valeurs de rappel comme:

$$\begin{aligned} R(0) &= 1 \\ R(k) &= R(\tau(k)) \text{ avec } k = 1,2,...,K \\ R(K+1) &= 0 \end{aligned} \tag{2.5.5}$$

Ainsi, on mesure avec l'équation 2.5.2 pour chaque valeur de rappel.

Ces métriques vont généralement être définies, par exemple, comme $AP50$, $AP75$, $AP[0.50 : 0.95]$ et mAP. Les métriques de $AP50$ et $AP75$ signifient de mesurer l'aire sous la courbe comme définis précédemment en utilisant respectivement un IoU de 0.5 et 0.75. Le IoU a été décrit à la section 2.2. La métrique $AP@[0.5 : 0.95]$ signifie la moyenne des AP avec 10 IoU entre 0.5 et 0.95 avec des bonds de 0.5. La métrique de mAP utilise normalement un IoU de 0.5 et ainsi correspond à $AP50$ moyenné sur toutes les classes C des labels, soit $mAP = \frac{1}{C} \sum_{i=1}^{C} AP_i$. Elle est seulement utilisée lorsqu'il y a plus d'une classe.

2.6. $AP_{small,medium,large}$, $AR_{small,medium,large}$

Les valeurs de AP (*Average Precision*) et AR (*Average Recall*) sont présentées à la section précédente. Les métriques de $AP_{small,medium,large}$ et $AR_{small,medium,large}$ sont aussi des métriques de *Average Precision* et *Average Recall* mais avec différentes contraintes. En effet, la notation *small*, *medium*, *large* représente la grandeur des boîtes de labels. La métrique AP_{small} réfère aux boîtes de labels avec une aire inférieure à 322 pixels,

AP_{medium} réfère aux boîtes de labels avec une aire entre 322 pixels et 962 pixels et AP_{large} réfère aux boîtes de labels avec une aire supérieure à 962 pixels.

On mesure ainsi le AP_{small}, AP_{medium}, AP_{large} et AR_{small}, AR_{medium}, AR_{large} en appliquant une moyenne $AP@[0.5 : 0.95]$ (comme expliqué à la section 2.5) selon la dimension des boîtes de labels. C'est une métrique intéressante à utiliser afin de connaître la performance d'un modèle selon différentes tailles d'objets. En effet, il peut être important d'avoir un modèle résistant à différentes tailles d'objets. Un exemple d'un tableau moyen entre 32^2 pixels et 96^2 pixels est présenté à la figure 6.1. On verra dans le chapitre 6 que les modèles sont nettement moins performants sur les détections de plus petites tailles.

Chapitre 3

Modèles

Les modèles particulièrement étudiés pour la détection des tableaux sont des réseaux de neurones qui sont entraînés à reconnaître des caractéristiques en leur fournissant des données pour acquérir des connaissances spécifiques. Dans ce chapitre, on introduit des modèles génériques utilisés comme modèles de référence, ainsi que des modèles spécialement conçus pour la tâche de détection de tableaux.

3.1. Modèle de références

Cette section décrit quelques modèles et squelettes souvent utilisés dans le domaine de la détection d'objets. Des références à ces modèles sont faites au cours de ce mémoire, car plusieurs auteurs entraînent ces réseaux de neurones pour la détection de tableaux afin de définir des modèles de référence.

3.1.1. Fast R-CNN

R-CNN [**Girshick et al., 2013**] représente Region-based convolutional neural networks. R-CNN est un modèle pour la détection d'objets qui utilise un réseau neuronal convolutif profond, mais qui possède quelques défauts, dont un entraînement lent et lourd, et une détection d'objet lente.

Fast R-CNN représente Fast Region-based Convolutional Network [**Girshick, 2015**] et se veut une amélioration du R-CNN. Fast R-CNN est aussi un modèle créé pour la détection d'objets. Ce modèle entraîne un réseau de neurones très profond, VGG16 [**Simonyan and Zisserman, 2014**] de manière rapide. Cette rapidité est due au fait qu'au lieu de passer plusieurs régions (*region proposals*) au CNN, l'image est passée directement au CNN pour générer la *map* de convolution. Ainsi, Fast R-CNN répond au défaut d'un R-CNN et possède une meilleure qualité de détection.

3.1.2. Faster R-CNN

Faster R-CNN est un des meilleurs modèles pour la détection d'objet [**Ren et al., 2015**]. Ce modèle est une amélioration de Fast R-CNN. C'est un modèle qui utilise la recherche sélective (selective search) pour trouver les régions importantes, plus précisément ce modèle introduit le Region Proposal Network (RPN) qui permet d'utiliser toute l'image sur les features de la convolution. Le RPN agit comme un FCN (Fully Convolutional Network) qui peut être entraîné pour générer les propositions de détections. Le fonctionnement est d'avoir tout d'abord un FCN qui propose les régions d'intérêts qui sont ensuite passées à Fast R-CNN. C'est ainsi un modèle plus rapide et de meilleure qualité.

3.1.3. Mask R-CNN

Mask R-CNN est un modèle développé par [**He et al., 2017**] qui permet d'identifier la segmentation d'un objet, soit associer chaque pixel d'une image à une catégorie. À la figure 3.1 on présente un résultat du modèle Mask R-CNN. Obtenir le masque d'une détection est très utile pour des formes avec beaucoup de courbes ou simplement des objets pouvant être présentés sous différents angles. Par exemple, une tâche de détection d'un permis de conduire qui peut être exposé selon différentes diagonales. La segmentation d'objets est une tâche qui nécessite aussi la détection d'objets. Mask R-CNN est un modèle qui a été développé par l'équipe de Detectron décrit à la section 3.2.

Fig. 3.1. Exemple de prédiction du modèle Mask R-CNN.

3.1.4. ResNet et ResNeXt

ResNet (residual neural network) [**He et al., 2015**] est un réseau de neurones qui introduit des blocs résiduels où la sortie d'une couche est l'entrée quelques couches plus loin et non la suivante tel un réseau traditionnel. Cette technique permet de palier au problème de fuite du gradient (vanishing gradient).

Tandis que ResNeXt [**Xie et al., 2016**] est un variant de ResNet où les créateurs ajoutent une dimension de cardinalité aux dimensions de la largeur et profondeur. La cardinalité définie la taille de l'ensemble transformé, soit le nombre de chemins parallèles d'un bloc appelé branches résiduelles. Cette technique permet de fortement augmenter la précision des résultats. Les modèles peuvent être nommés comme ResNeXt-101-64x4d où 101 fait référence à la profondeur du réseau, 64 fait référence à la cardinalité et 4 fait référence à la largeur du bloc résiduel.

3.2. Detectron

Detectron agit comme une librairie créée par le groupe de recherche de Facebook [**Wu et al., 2019**]. Il offre des modèles préentraînés de différentes tailles telles que Mask R-CNN, Faster R-CNN, RPN, Fast R-CNN et R-FCN. Ceux-ci utilisent les réseaux ResNet{50,101,152}, ResNeXt{50,101,152} et VGG16 comme squelette. Les modèles sont préentraînés sur ImageNet [**Lab et al., 2011**] qui est un ensemble de données avec près de 14 millions d'images avec environs 20 000 catégories comme 'ballon', 'chien', 'livre', etc. Les modèles disponibles préentraînés agissent comme des modèles de références. Detectron offre aussi des méthodes pour affiner les modèles préentraînés avec un ensemble de données quelconque et plusieurs paramètres avec lesquels tester. Il est aussi possible d'entraîner un modèle de zéro. Il est pertinent d'utiliser ces modèles préentraînés, parce que les ensembles de données de détection de tableaux ne sont pas assez grands pour entraîner de tels modèles complexes. Ces modèles sont tellement généraux, qu'un petit ensemble de données pour affiner s'avère très efficace. À noter que Detectron accorde un pourcentage de confiance sur chaque détection, une bonne détection se voit accorder un haut pourcentage.

3.3. Modèle TableBank

La réalisation du modèle de TableBank est faite en utilisant le *framework* Detectron. Pour la tâche de détection de tableaux, les auteurs [**Li et al., 2019b**] utilisent Faster R-CNN avec ResNeXt comme architecture. Faster R-CNN est un modèle pour la détection d'objets qui utilise un réseau neuronal convolutif profond, décrit à la section 3.1.2. Son utilisation pour la détection de tableaux est illustrée à la figure 3.2.

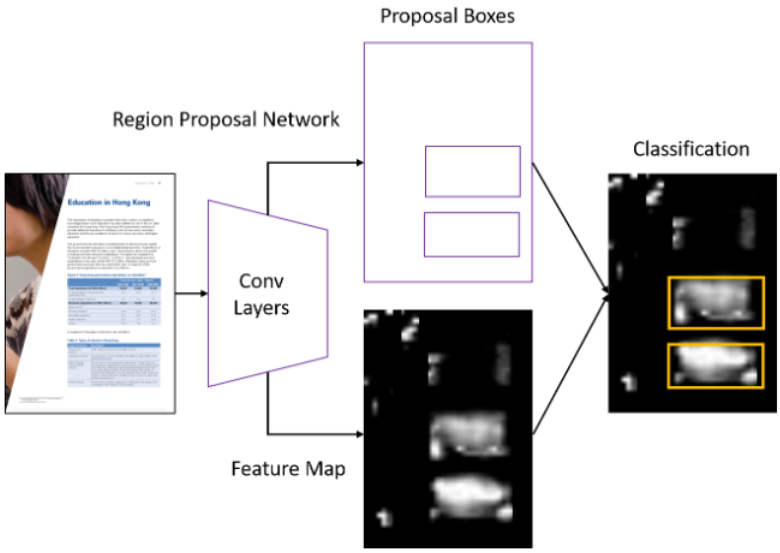


Fig. 3.2. Représentation du modèle Faster R-CNN pour la détection de tableaux par [**Li et al., 2019b**]

Les auteurs utilisent le réseau de neurones ResNeXt, décrit à la section 3.1.4, comme architecture où les paramètres sont préentraînés sur ImageNet [**Lab et al., 2011**]. Comme mentionné, c'est un ensemble de données avec des labels comme 'ballon', 'chien', 'livre', etc. Les paramètres sont ceux définis par défaut par Detectron. De plus, deux types d'architecture sont utilisés, ResNeXt-101 et ResNeXt-152 où 101 et 152 font référence à la profondeur du réseau. Au total, l'ensemble d'entraînement de TableBank est de 260 582 images. Il est aussi à noter qu'ils utilisent un seuil de confiance de 90% pour la prédiction qui est retournée par Detectron. Comme mentionné, les auteurs créent aussi les annotations pour la tâche de structure de table. Pour cette tâche, ils utilisent aussi un Faster R-CNN comme modèle de référence.

3.4. CascadeTabNet

3.4.1. Description

CascadeTabNet est un modèle entraîné spécifiquement pour la reconnaissance et la détection de tableaux [**Prasad et al., 2020b**]. Ce modèle utilise un réseau de convolution. Plus précisément, c'est un Cascade Mask R-CNN High-Resolution Network (HRNet). Leur approche consiste à utiliser un petit ensemble de données de manière efficace en utilisant une architecture de CNN et d'apprentissage par transfert. Leur modèle est inspiré de Cascade R-CNN [**Cai and Vasconcelos, 2018**], un modèle R-CNN où est introduit un modèle avec une architecture par étapes multiples (multi-stage) où on passe par différentes étapes de détection avec des seuils d'IoU de plus en plus sévères. Tandis que Cascade mask R-CNN est un modèle Cascade R-CNN où on ajoute une branche pour la segmentation. CascadeTabNet est un Cascade mask R-CNN avec

2 étapes d'apprentissage par transfert. Ces étapes sont décrites plus bas. Le modèle de Cascade mask R-CNN utilise *HRNet* (High-Resolution Network) [**Wang et al., 2019**] comme squelette. Un High-Resolution Network est un CNN tout usage pour des tâches de segmentation sémantique, détection d'objets et classification d'images. C'est un modèle en mesure de conserver la haute résolution tout au long du processus. Plus précisément, CascadeTabNet utilise HRNetV2p_W32.

L'apprentissage par transfert en deux étapes permet d'avoir la détection et reconnaissance de tableaux avec peu de données. Les auteurs ont créé un ensemble de données général pour une tâche de détection de tableaux. Ils utilisent des documents provenant de Word et Latex. Les documents contiennent des tableaux complètement bordurés, à demi bordurés et sans aucune bordure. Pour l'ensemble de données général, ils utilisent l'ensemble de données ICDAR2019 (section 4.2), l'ensemble de données Marmot (section 4.3), et un ensemble de données de Github (section 4.6). Pour ICDAR2019, ils utilisent seulement le sous ensemble moderne qui est majoritairement constitué de documents Word et Latex en anglais et différents langages de Chine. Ils retirent les documents qui contiennent des erreurs dans les annotations. En tout, il y a 1934 images et 2835 annotations de tables.

La première itération consiste à affiner le modèle sur l'ensemble défini afin de détecter les tableaux. En effet, tous les tableaux de l'ensemble défini sont sous une même catégorie, soit un tableau. Le modèle est initialisé avec les poids d'un modèle préentraîné avec ImageNet. Ces poids proviennent de l'outil de détection d'objet MMDetection [**Chen et al., 2019**]. Cet outil possède des modèles préentraînés sur ImageNet. Utiliser des modèles préentraînés sur ImageNet est une méthode assez commune en détection d'objets. Une fois que le modèle est en mesure de détecter les tableaux dans un document, on passe à la deuxième étape.

Dans la deuxième itération, le modèle est affiné sur un ensemble de données plus spécifique afin qu'il apprenne à détecter les cellules d'un tableau non borduré ainsi que classer les types de tableaux avec bordure complète et sans bordure. Les tableaux avec un peu de bordures sont ici classés comme des tableaux sans bordure. C'est à la suite de cet affinage que le modèle est en mesure de détecter le type et la segmentation des tableaux. Lorsqu'un tableau est détecté comme étant avec bordure, un simple algorithme de détection de lignes est utilisé. Il est ainsi facile de détecter les cellules. Ils affirment qu'il est plus efficace d'utiliser un tel algorithme pour reconnaitre les cellules qu'un modèle profond. Tandis que lorsque le tableau est détecté sans bordure, les cellules sont détectées en utilisant un algorithme de détection de texte.

Ils ont aussi tenté d'utiliser des méthodes d'augmentation de données qui ont de grandes utilités dans l'apprentissage profond. Cependant, c'est un cas plus délicat parce que les données sont des documents. Par exemple, des méthodes de rotations sont illogiques. Les transformations proposées sont la transformation par élargissement et la transformation par bavure. La première consiste à grossir les régions contenant des pixels noirs en convertissant les images en noir et blanc. Tandis que pour la transformation par bavure, l'image est convertie en noir et blanc puis un effet de flou sur les pixels noirs est appliqué. L'effet de flou est créé en appliquant diverses transformations de distance comme *Euclidean Distance Transform*, *Linear Distance Transform* et *Max Distance Transform*. Ces méthodes ont été proposées dans [**Gilani et al., 2017**]. Pour appliquer ces transformations, ils créent 4 ensembles d'entraînement. Le premier ensemble est le groupe original. Le deuxième groupe est le groupe original où on ajoute les mêmes images avec la transformation par élargissement. Le troisième groupe est le groupe original où on ajoute les mêmes images avec la transformation par bavure. Le quatrième groupe est le groupe original, les images avec transformation par élargissement et les images avec transformation par bavure.

3.4.2. Résultats

Les auteurs évaluent CascadeTabNet sur l'ensemble de données TableBank (section 4.4). Pour ce faire, ils affinent le modèle sur un petit ensemble de TableBank. Pour tester sur Latex, ils affinent le modèle sur 1500 images de Latex choisies aléatoirement, et l'ensemble de test est composé de 1000 images de Latex choisies aléatoirement. Tandis que pour tester sur Word, ils affinent le modèle sur 1500 images de Word choisies aléatoirement, et l'ensemble de test est composé de 1000 images Word choisies aléatoirement. Ils ont découvert que les annotations de TableBank Word contiennent des erreurs. Ils n'incluent pas ces images dans l'ensemble de test. Pour tester sur Word et Latex combinés, ils affinent sur 1500 images de Latex choisies aléatoirement et 1500 images de Word choisies aléatoirement, et l'ensemble de test est composé de 1000 images Latex et 1000 images Word. Ils évaluent aussi le modèle sur ICDAR2013. Ils utilisent 40 images pour affiner le modèle et 198 images pour l'ensemble de test.

CascadeTabNet utilise le IoU comme métrique avec différents seuils de 0.6, 0.7, 0.8 et 0.9 et le weighted-average F1 (section 2.2 et 2.3). Sur l'ensemble ICDAR2013, ils utilisent la complétude et la pureté (section 2.4) qui sont les métriques utilisées cet ensemble. Tandis que sur l'ensemble de TableBank, ils utilisent la même métrique que TableBank (section 2.1). Ces résultats sont présentés à la figure 3.3. On se limite à présenter les résultats sur l'ensemble de données TableBank puisque c'est l'ensemble le plus étudié dans ce mémoire.

Sur l'ensemble ICDAR2019, leurs modèles battent seulement les deux meilleurs avec un seuil IoU de 0.9. Sur l'ensemble de TableBank, ils battent les modèles de TableBank selon le score F1. Ils obtiennent les meilleurs résultats pour ICDAR2013. Aussi, ils ont mesuré qu'ajouter les deux transformations dans l'ensemble d'entraînement permettait de bien améliorer les performances.

Dataset	Model	Precision	Recall	F1
Both	ResNeXt-101	95.93	90.44	93.11
	ResNeXt-152	96.72	88.95	92.67
	Ours	**92.99**	**95.71**	**94.33**
Latex	ResNeXt-101	87.44	95.12	91.12
	ResNeXt-152	87.20	96.24	91.49
	Ours	**95.92**	**97.28**	**96.60**
Word	ResNeXt-101	95.77	76.10	84.81
	ResNeXt-152	96.50	80.32	87.67
	Ours	**94.35**	**95.49**	**94.92**

Fig. 3.3. Résultat des auteurs du modèle CascadeTabNet sur l'ensemble de données TableBank.

Chapitre 4

Ensembles de données

Les ensembles de données sont une partie fondamentale de l'apprentissage machine. C'est grâce à l'accessibilité à de grands ensembles de données de qualité que l'on peut aujourd'hui constater les avancées remarquables dans le domaine. Les ensembles de données rendent possible l'entraînement des réseaux neurones et influencent fortement les performances. Ils sont intrinsèques à l'évaluation des modèles en tant que références d'évaluation et doivent être représentatifs du cas étudié. De plus, pour la détection d'objets, il est nécessaire d'avoir des ensembles de données contenant les labels des images et cette exigence demande un considérable effort d'annotation. Avec les années, les scientifiques ont commencé à développer des algorithmes pour annoter automatiquement les images et ce mécanisme a permis de créer de plus gros ensembles de données indispensables à l'entraînement de réseaux de neurones. Dans ce chapitre, on décrit les différents ensembles de données pour la détection de tableaux ainsi que les modèles et résultats obtenus pour tester la pertinence des données.

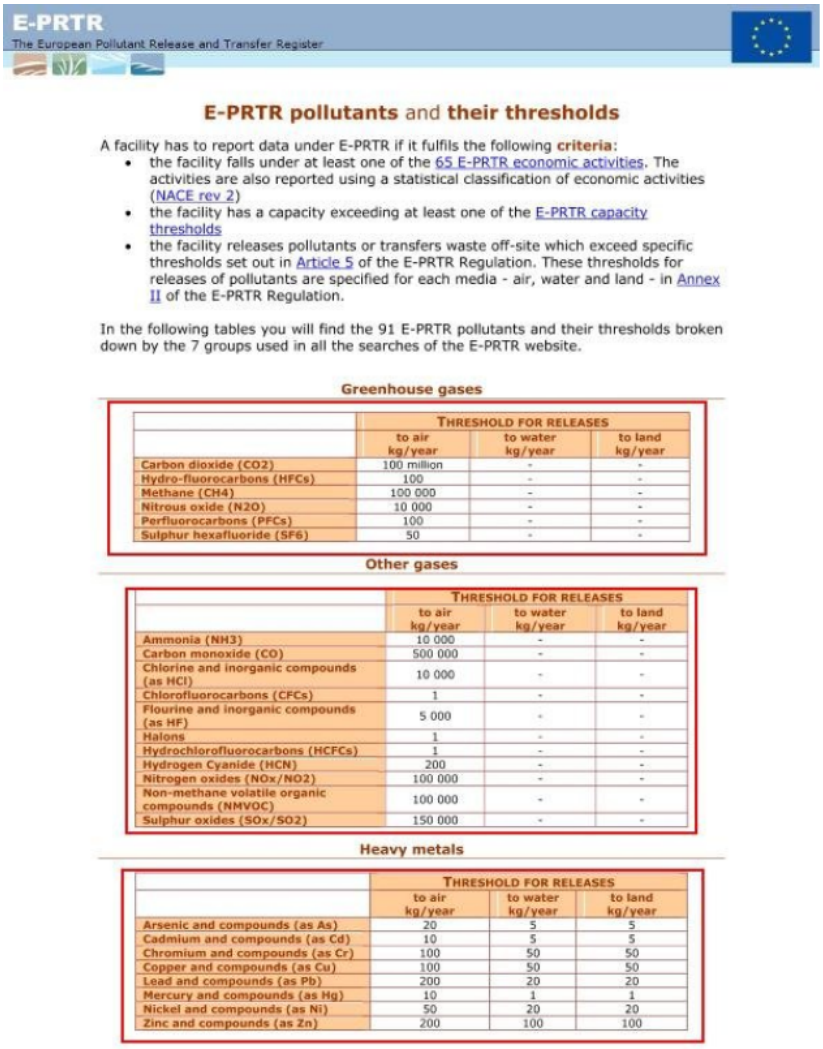
4.1. ICDAR2013

4.1.1. Description

ICDAR2013 (International Conference on Document Analysis and Recognition 2013) [**Göbel et al., 2013**] est une compétition qui adresse la détection et la reconnaissance de structure de tableaux. Un ensemble de données portant le même nom, ICDAR2013, a été introduit afin de réaliser la compétition. Un exemple est présenté à la figure 4.1. L'ensemble de données contient 150 tableaux. Il y a deux sous-ensembles qui sont dénombrés au tableau 4.1 où EU représente des documents de l'Union Européenne et US représente des documents des États-Unis. Au fil du temps, cet ensemble de données est devenu une référence d'évaluation (*benchmark*) souvent utilisée par les scientifiques pour comparer la qualité de leurs modèles de détections de tableaux.

Ensemble	EU	US
Nombre de documents	27	40
Nombre de tableaux	75	75

Tableau 4.1. Résumé de la taille de l'ensemble de données ICDAR2013 pour la détection de tableaux.



E-PRTR
The European Pollutant Release and Transfer Register

E-PRTR pollutants and their thresholds

A facility has to report data under E-PRTR if it fulfils the following **criteria**:

- the facility falls under at least one of the 65 E-PRTR economic activities. The activities are also reported using a statistical classification of economic activities (NACE rev 2)
- the facility has a capacity exceeding at least one of the E-PRTR capacity thresholds
- the facility releases pollutants or transfers waste off-site which exceed specific thresholds set out in Article 5 of the E-PRTR Regulation. These thresholds for releases of pollutants are specified for each media - air, water and land - in Annex II of the E-PRTR Regulation.

In the following tables you will find the 91 E-PRTR pollutants and their thresholds broken down by the 7 groups used in all the searches of the E-PRTR website.

Greenhouse gases

	Threshold for releases		
	to air kg/year	to water kg/year	to land kg/year
Carbon dioxide (CO2)	100 million	-	-
Hydro-fluorocarbons (HFCs)	100	-	-
Methane (CH4)	100 000	-	-
Nitrous oxide (N2O)	10 000	-	-
Perfluorocarbons (PFCs)	100	-	-
Sulphur hexafluoride (SF6)	50	-	-

Other gases

	Threshold for releases		
	to air kg/year	to water kg/year	to land kg/year
Ammonia (NH3)	10 000	-	-
Carbon monoxide (CO)	500 000	-	-
Chlorine and inorganic compounds (as HCl)	10 000	-	-
Chlorofluorocarbons (CFCs)	1	-	-
Flourine and inorganic compounds (as HF)	5 000	-	-
Halons	1	-	-
Hydrochlorofluorocarbons (HCFCs)	1	-	-
Hydrogen Cyanide (HCN)	200	-	-
Nitrogen oxides (NOx/NO2)	100 000	-	-
Non-methane volatile organic compounds (NMVOC)	100 000	-	-
Sulphur oxides (SOx/SO2)	150 000	-	-

Heavy metals

	Threshold for releases		
	to air kg/year	to water kg/year	to land kg/year
Arsenic and compounds (as As)	20	5	5
Cadmium and compounds (as Cd)	10	5	5
Chromium and compounds (as Cr)	100	50	50
Copper and compounds (as Cu)	100	50	50
Lead and compounds (as Pb)	200	20	20
Mercury and compounds (as Hg)	10	1	1
Nickel and compounds (as Ni)	50	20	20
Zinc and compounds (as Zn)	200	100	100

Fig. 4.1. Un exemple de l'ensemble de données de ICDAR2013 où les tableaux sont encadrés par les rectangles rouges.

L'ensemble de données ICDAR2013 a été fabriqué en appliquant une extraction de tableaux sur des documents PDF générés à partir de domaines publics de sources gouvernementales comme *site:europa.eu* et *:*.gov*. Ces documents PDF ont été transformés en image pour répondre aux tâches qui consistent à localiser des tableaux en retournant les boîtes de détections et à la reconnaissance de structure de tableaux en retournant les colonnes, lignes et leur contenu textuel.

4.1.2. Résultats

Dans la compétition, la majorité des concurrents utilisent du traitement d'images avec des heuristiques se basant sur les lignes et le texte pour détecter les tableaux.

Les résultats des concurrents pour la détection de tableaux sont mesurés sur différentes métriques. Les premières sont celles de complétude et de pureté qui sont décrites à la section 2.4. Les fondateurs de la compétition ajoutent aussi les métriques de précision et rappel, car les mesures de complétude et pureté ne font pas de différence entre une grande partie manquante sur une détection et une petite partie manquante sur une détection. Pour faire la moyenne des scores sur tous les documents, les auteurs ont décidé d'évaluer la moyenne en mesurant la précision et le rappel pour chaque document séparément et d'ensuite faire la moyenne. De cette façon, chaque document à un poids égal, et les résultats ne sont pas biaisés par des documents avec beaucoup de tableaux ou cellules.

Les auteurs observent aussi les résultats d'outils commerciaux tels que ABBYY FineReader 11.0 Corporate Edition, Adobe Acrobat XI Pro, OmniPage 19 Professional, Nitro Pro 8. Deux de ceux-ci se placent aux premiers rangs avec des scores-F1 de 98.48% et 96.06%, tandis que la troisième place revient à un concurrent de *Laboratory of Artificial Intelligence and Decision Support* avec un score-F1 de 95.54%. Pour le reste, les systèmes commerciaux sont généralement supérieurs à ceux des participants académiques. Ces premiers semblent être davantage dépendants de la présence de lignes, par contre, comme le fonctionnement de ces outils ne sont pas disponibles au public, on peut se questionner sur la raison pour laquelle ces méthodes fonctionnent aussi bien. Cette amélioration pourrait être due à un grand nombre de méthodes ad-hoc pour gérer tous les cas spéciaux, ou être due à un grand nombre de données auxquelles ces compagnies ont accès. Il est également possible que les méthodes et modèles soient réellement meilleures.

4.2. ICDAR2019

4.2.1. Description

ICDAR2019 (International Conference on Document Analysis and Recognition 2019) [**Gao et al., 2019**] est une compétition qui adresse la détection et la reconnaissance de structure de tableaux. Un ensemble de données portant le même nom, ICDAR2019, a été introduit afin de réaliser la compétition. C'est un ensemble utilisé comme une référence d'évaluation (benchmark) pour la détection de tableaux. L'ensemble de données contient deux sous ensembles, nommés le *Modern* et *Archive*. Le sous ensemble *Archive* contient

des documents historiques avec des tableaux et de l'écriture manuscrite provenant de plus de 23 institutions à travers le monde. Les documents contiennent des tableaux de comptabilité, marché boursier, horaire de train, etc. Le sous ensemble *Modern* est constitué de documents PDF anglais et chinois provenant de journaux scientifiques, des formulaires, des documents financiers, etc. Les deux sous ensembles sont distribués sous formes d'images. Un exemple de chaque ensemble est montré à la figure 4.2.

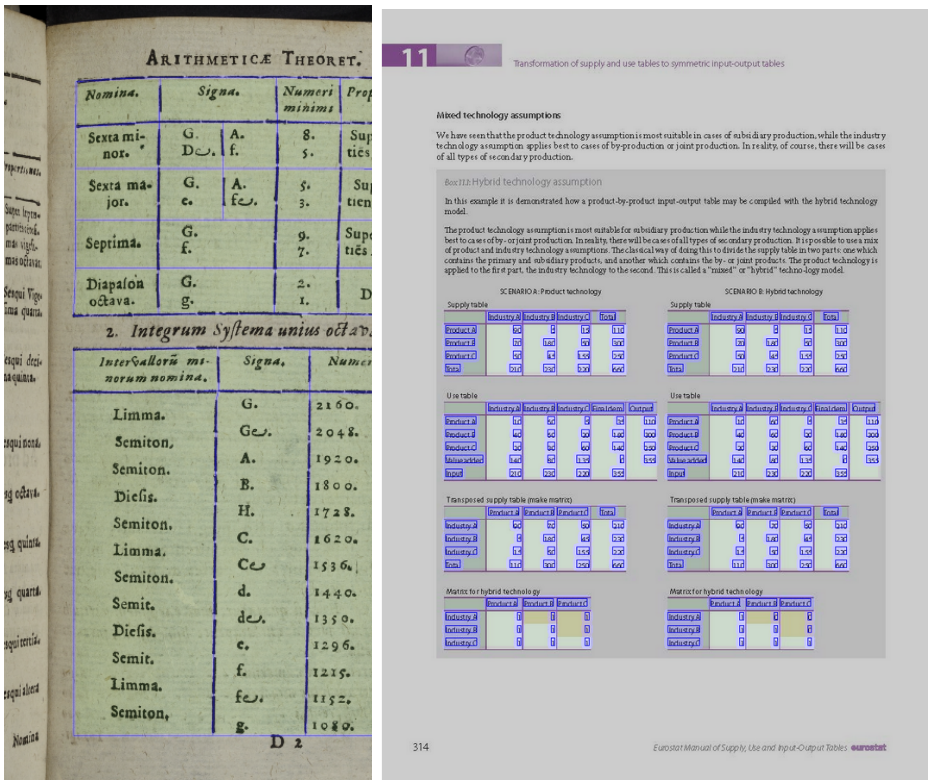


Fig. 4.2. Exemple de détection de tableaux de l'ensemble de données de ICDAR2019, où l'exemple de gauche appartient à l'ensemble *Archive* et l'exemple de droite appartient à l'ensemble *Modern*.

Ensemble	**Archive**		**Modern**	
	Train	Test	Train	Test
Nombre de documents	600	199	600	240

Tableau 4.2. Taille de l'ensemble de données ICDAR2019 pour la détection de tables.

L'ensemble historique permet de tester la robustesse avec des tableaux avec du bruit et du texte manuscrit. Par contre, on peut se questionner sur l'utilité d'un tel ensemble sachant que dans les domaines professionnels tout est électronique. Il n'y a pas de cas de tableaux imbriqués, comme c'est le cas avec TableBank, un ensemble de données décrit à la section 4.4. La taille de l'ensemble de détection de tableaux est décrit au tableau 4.2, où chaque document contient au moins une table.

4.2.2. Résultats

Pour la compétition, certaines équipes utilisent des transformations pour augmenter les données comme du brouillement, des rotations, différentes résolutions, etc. La majorité des équipes utilisent Faster R-CNN, un modèle pour la détection d'objets qui utilise un réseau neuronal convolutif profond (décrit à la section 3.1.2), afin d'entraîner le modèle à détecter les tableaux.

Comme métrique, les auteurs comparent les concurrents avec le IoU avec différents seuils de 0.6, 0.7, 0.8 et 0.9 et le *weighted-average F1*. Ces métriques sont décrites à la section 2.2 et 2.3.

Le meilleur concurrent entraîne deux modèles de Faster R-CNN pour détecter les tableaux dans les deux ensembles *Modern* et *Archive* avec un *weighted-average F1* de 94%. Tandis que le deuxième meilleur concurrent utilise un unique modèle Fully Convolutional Network (FCN) sur les deux ensembles avec un *weighted-average F1* de 93%.

4.3. Marmot

4.3.1. Description

Marmot est un ensemble de données qui a été publié par l'Institute of Computer science and Technologie de l'université de Peking [**Institute of Computer Science and Technology of Peking University, 2011**]. L'ensemble de données contient 2000 pages de documents en langues chinoises et en anglais. Les documents chinois proviennent de différents livres électroniques de *Founder Apabi* [1] où un maximum de 15 pages sont extraites de chaque livre. Tandis que les documents en anglais proviennent du site web *CiteSeerX* [2] qui est une librairie numérique d'articles scientifiques et académiques. L'ensemble de données contient des documents provenant de plus de 1500 conférences et journaux scientifiques dans différents domaines des années 1970 à 2011. Un exemple est présenté à la figure 4.3.

[1] www.apabi.com
[2] https://csxstatic.ist.psu.edu/

无法取用的冰川和高山顶上的冰雪储量，理论上可以开发利用的淡水不到地球总水量的1%；实际上，人类可以利用的淡水量远低于此理论值，主要是因为在总降水量中，有些是落在无人居住的地区如南极洲，或者降水集中于很短的时间内，由于缺乏有效的水利工程措施，很快地流入海洋之中。由此可见，尽管地球上的水是取之不尽的，但适合饮用的淡水水源是十分有限的，表 3-2 是世界各洲可用的淡水量。

表 3-2　世界各洲可用淡水量

洲名称	面积 /km²	人口 /百万	径流量 /(km³/a)	可用水量 /[人/(百万立方米·年)]	可用水量 /[立方米/(人·天)]
欧洲	10500	495	3210	152	18
亚洲	43475	3108	14410	211	13
非洲	30120	648	4570	144	19
北美及中美洲	24200	426	8200	52	53
南美洲	17800	297	11760	25	108
大洋洲	8950	26	2388	11	252
总计	13505	5003	44540	114	24

资料来源：Shiklomanov (1993)；da Cunha (1994)。

1997 年 10 月 20 日，《水信息报》刊登了世界 153 个国家水资源情况和各国用水量及其构成，国家水资源情况包括国土面积、水资源总量（总量，排序）、1995 年人口、1995 年人均水资源量（人均量，排序）、1995 年耕地面积和单位耕地面积水资源量，各国用水量及其构成包括年份、用水量、人均用水量、用水组成（生活用水、工业用水和农业用水）。这些国家指标前 10 名或者后 10 名情况见表 3-3。

表 3-3　世界 153 个国家中水资源指标排序

序号	排序指标	排　序	说　明
1	水资源量	巴西、俄罗斯、美国、印度尼西亚、加拿大、中国、孟加拉、印度、委内瑞拉、哥伦比亚	前 10 名
2	人均水资源量	科威特、利比亚、新加坡、沙特阿拉伯、约旦、也门共和国、以色列、突尼斯、阿尔及利亚、布隆迪	后 10 名
3	用水量	中国、美国、印度、巴基斯坦、俄罗斯、日本、乌兹别克斯坦、墨西哥、埃及	前 10 名
4	人均用水量	土库曼斯坦、乌兹别克斯坦、吉尔吉斯斯坦、塔吉克斯坦、阿塞拜疆、巴基斯坦、美国、阿富汗	前 10 名
5	人均年用水量	所罗门群岛、海地、刚果共和国、赤道几内亚、几内亚比绍、刚果、布隆迪、乌干达、中非共和国、贝宁	后 10 名

2. 世界水资源短缺

世界水资源供需状况并不乐观。1996 年 5 月，在纽约召开的“第三届自然资源委员会”上，联合国开发支持和管理服务部对 153 个国家（占世界人口的 98.93%）的水资源，采用人均占有水资源量、人均国民经济总产值、人均取（用）水量等指标进行综合分析，将世界各国分为四类，即水资源丰富国（包括吉布提等 100 多个国家）、水资源脆弱国（包括美国等 17 个国家）、水资源紧缺国（包括摩洛哥等 17 个国家）、水资源贫乏国（包括阿尔及利亚

49

Fig. 4.3. Un exemple de l'ensemble de données Marmot où deux tableaux sont encadrés par les rectangles bleus.

4.3.2. Résultats

Les auteurs ne présentent pas de modèle pour tester leur ensemble de données. Ils ajoutent un outil d'évaluation qu'ils ont créé et qui n'est pas basé sur des métriques existantes. Les détails de la métrique ne sont pas présentés parce qu'aucune trace de son utilisation n'a été rencontrée. Il n'y pas de résultats présentés, mais l'ensemble de données est utilisé pour entraîner le modèle CascadeTabNet, décrit à la section 3.4.

4.4. Tablebank

4.4.1. Description

TableBank est un ensemble de données créé par [**Li et al., 2019b**] pour la détection et la reconnaissance de tableaux de documents écrits avec Word et Latex. Étant donné le développement des réseaux de neurones, les auteurs voulaient créer un grand ensemble de données pour entraîner ces modèles afin qu'ils soient robustes aux variations dans les documents. La taille de l'ensemble de données est résumé au tableau 4.3. Tandis qu'un

exemple est présenté à la figure 4.4.

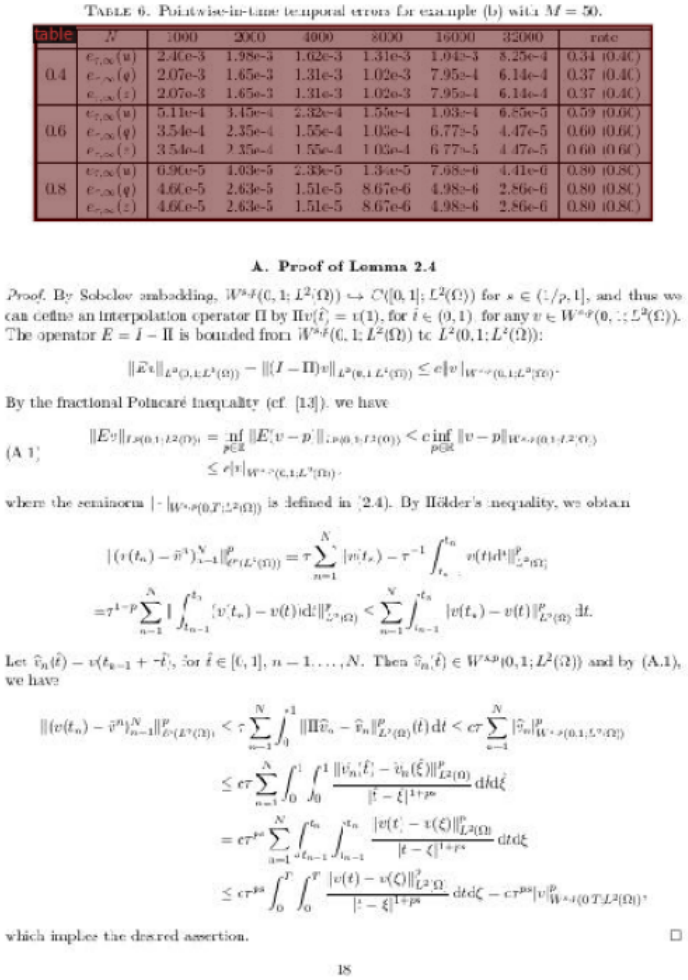


TABLE 6. Pointwise-in-time temporal errors for example (b) with $M = 50$.

table	N	1000	2000	4000	8000	16000	32000	rate
0.4	$e_{\tau,\infty}(u)$	2.40e-3	1.98e-3	1.62e-3	1.31e-3	1.04e-3	8.25e-4	0.34 (0.4C)
	$e_{\tau,\infty}(q)$	2.07e-3	1.65e-3	1.31e-3	1.02e-3	7.95e-4	6.14e-4	0.37 (0.4C)
	$e_{\tau,\infty}(z)$	2.07e-3	1.65e-3	1.31e-3	1.02e-3	7.95e-4	6.14e-4	0.37 (0.4C)
0.6	$e_{\tau,\infty}(u)$	5.11e-4	3.45e-4	2.32e-4	1.55e-4	1.03e-4	6.85e-5	0.59 (0.6C)
	$e_{\tau,\infty}(q)$	3.54e-4	2.35e-4	1.55e-4	1.03e-4	6.77e-5	4.47e-5	0.60 (0.6C)
	$e_{\tau,\infty}(z)$	3.54e-4	2.35e-4	1.55e-4	1.03e-4	6.77e-5	4.47e-5	0.60 (0.6C)
0.8	$e_{\tau,\infty}(u)$	0.90e-5	4.03e-5	2.33e-5	1.34e-5	7.68e-6	4.41e-6	0.80 (0.8C)
	$e_{\tau,\infty}(q)$	4.6e-5	2.63e-5	1.51e-5	8.67e-6	4.98e-6	2.86e-6	0.80 (0.8C)
	$e_{\tau,\infty}(z)$	4.6e-5	2.63e-5	1.51e-5	8.67e-6	4.98e-6	2.86e-6	0.80 (0.8C)

A. Proof of Lemma 2.4

Proof. By Sobolev embedding, $W^{s,p}(0,1;L^2(\Omega)) \hookrightarrow C([0,1];L^2(\Omega))$ for $s \in (1/p,1]$, and thus we can define an interpolation operator Π by $\Pi v(\hat{t}) = v(1)$, for $\hat{t} \in (0,1)$ for any $v \in W^{s,p}(0,1;L^2(\Omega))$. The operator $E = I - \Pi$ is bounded from $W^{s,p}(0,1;L^2(\Omega))$ to $L^2(0,1;L^2(\Omega))$:

$$\|\widehat{E}v\|_{L^2(0,1;L^2(\Omega))} = \|(I-\Pi)v\|_{L^2(0,1;L^2(\Omega))} \le c|v|_{W^{s,p}(0,1;L^2(\Omega))}.$$

By the fractional Poincaré inequality (cf. [13]), we have

$$\begin{aligned} \|Ev\|_{L^p(0,1;L^2(\Omega))} &= \inf_{p\in\mathbb{R}} \|E(v-p)\|_{L^p(0,1;L^2(\Omega))} \le c \inf_{p\in\mathbb{R}} \|v-p\|_{W^{s,p}(0,1;L^2(\Omega))} \\ &\le c|v|_{W^{s,p}(0,1;L^2(\Omega))}. \end{aligned} \tag{A.1}$$

where the seminorm $|\cdot|_{W^{s,p}(0,T;L^2(\Omega))}$ is defined in (2.4). By Hölder's inequality, we obtain

$$\begin{aligned} &|(v(t_n)-\bar{v}^n)_{n=1}^N\|_{\ell^p(L^2(\Omega))}^p = \tau\sum_{n=1}^N |v(t_n) - \tau^{-1}\int_{t_{n-1}}^{t_n} v(t)\mathrm{d}t|_{L^2(\Omega)}^p \\ &= \tau^{1-p}\sum_{n=1}^N \Big\| \int_{t_{n-1}}^{t_n} (v(t_n)-v(t))\mathrm{d}t\Big\|_{L^2(\Omega)}^p \le \sum_{n=1}^N \int_{t_{n-1}}^{t_n} |v(t_n)-v(t)|_{L^2(\Omega)}^p \,\mathrm{d}t. \end{aligned}$$

Let $\hat{v}_n(\hat{t}) = v(t_{n-1}+\tau\hat{t})$, for $\hat{t}\in[0,1]$, $n=1,\dots,N$. Then $\hat{v}_n(\hat{t}) \in W^{s,p}(0,1;L^2(\Omega))$ and by (A.1), we have

$$\begin{aligned} \|(v(t_n)-\bar{v}^n)_{n=1}^N\|_{\ell^p(L^2(\Omega))}^p &\le \tau\sum_{n=1}^N \int_0^1 \|\Pi\hat{v}_n - \hat{v}_n\|_{L^2(\Omega)}^p(\hat{t})\,\mathrm{d}\hat{t} \le c\tau\sum_{n=1}^N |\hat{v}_n|_{W^{s,p}(0,1;L^2(\Omega))}^p \\ &\le c\tau\sum_{n=1}^N \int_0^1\int_0^1 \frac{\|\hat{v}_n'(\hat{t}) - \hat{v}_n'(\hat{\xi})\|_{L^2(\Omega)}^p}{|\hat{t}-\hat{\xi}|^{1+ps}}\,\mathrm{d}\hat{t}\mathrm{d}\hat{\xi} \\ &= c\tau^{ps}\sum_{n=1}^N \int_{t_{n-1}}^{t_n}\int_{t_{n-1}}^{t_n} \frac{|v(t)-v(\xi)|_{L^2(\Omega)}^p}{|t-\xi|^{1+ps}}\,\mathrm{d}t\mathrm{d}\xi \\ &\le c\tau^{ps}\int_0^T\int_0^T \frac{|v(t)-v(\zeta)|_{L^2\Omega}^2}{|t-\xi|^{1+ps}}\,\mathrm{d}t\mathrm{d}\zeta = c\tau^{ps}|v|_{W^{s,p}(0,T;L^2(\Omega))}^p, \end{aligned}$$

which implies the desired assertion. □

18

Fig. 4.4. Un exemple de l'ensemble de données TableBank où un tableau est encadré par le rectangle rouge.

Ensemble	**Word**		**Latex**	
	Train	Test	Train	Test
Nombre de documents	73 383	2 281	187 199	5 719
Nombre de tableaux	95 404	2 930	237 431	7 246

Tableau 4.3. Résumé de la taille de l'ensemble de données TableBank pour la détection de tables.

Les documents Word sont trouvés sur internet en format *.docx* afin d'utiliser le code *xml* de Office. Les documents sont en anglais, japonais, arabe et langages de Chine. Les documents sont bien diversifiés. Tandis que les documents Latex proviennent de *arXiv.org*[3] où les PDF et les fichiers *.tex* de 2014 à 2018 sont disponibles. La grande

[3]arXiv.org

majorité des documents est en anglais. Tous les documents contiennent au moins une table. Les tableaux sont annotés en utilisant les codes sources des documents. Pour Word, les tableaux sont annotés en format xml comme *<w:tbl>* et pour Latex, les tableaux sont souvent annotés avec \begin{table} \end{table}. Les documents PDF sont ensuite transformés en image pour être utilisés avec des modèles de détection d'images.

Pour mesurer leur taux d'erreurs, ils ont aléatoirement sélectionné 1000 images de l'ensemble de données et ont trouvé 5 images avec des erreurs d'annotations. Ils assument alors un taux d'erreur de 0.5%. Ce taux d'erreurs est discuté à la section 5.2 et semble, en réalité, beaucoup plus élevé.

4.4.2. Résultats

Le modèle est décrit en détail à la section 3.3. Comme modèle de référence, les auteurs se servent du *framework* Detectron afin d'utiliser le modèle de Faster R-CNN avec deux types d'architecture, ResNeXt-101 et ResNeXt-152. Au total, l'ensemble d'entraînement de TableBank est de 260 582 images. Ils entraînent un modèle avec seulement les images de Word, un modèle avec seulement les images de Latex, et un modèle avec les deux types d'images. Ces différents modèles permettent d'observer l'impact de différents ensembles d'entraînement.

La métrique utilisée a été introduite dans [**Gilani et al., 2017**] spécifiquement pour la détection de tableaux dans des documents. Celle-ci est définie à la section 2.1. De plus, les auteurs de TableBank ont décidé de définir un seuil d'acceptation des prédictions de Detectron de 90%.

Pour évaluer l'ensemble de données, ils utilisent 1000 images de Word et 1000 images de Latex de l'ensemble test. Ils évaluent aussi leur modèle sur ICDAR2013. Les résultats sont présentés au tableau 4.4. On présente les détails des résultats obtenus par les auteurs parce que TableBank est un des ensembles de données principalement étudié dans ce mémoire.

Models	Word			Latex			Word+Latex		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
ResNeXt-101 (Word)	0.9496	0.8388	0.8908	0.9902	0.5948	0.7432	0.9594	0.7607	0.8486
ResNeXt-152 (Word)	0.9530	0.8829	**0.9166**	0.9808	0.6890	0.8094	0.9603	0.8209	0.8851
ResNeXt-101 (Latex)	0.8288	0.9395	0.8807	0.9854	0.9760	0.9807	0.8744	0.9512	0.9112
ResNeXt-152 (Latex)	0.8259	0.9562	0.8863	0.9867	0.9754	**0.9810**	0.8720	0.9624	0.9149
ResNeXt-101 (Word+Latex)	0.9557	0.8403	0.8943	0.9886	0.9694	0.9789	0.9670	0.8817	0.9224
ResNeXt-152 (Word+Latex)	0.9540	0.8639	0.9067	0.9885	0.9732	0.9808	0.9657	0.8989	**0.9311**

Tableau 4.4. Résultats, présentés dans [**Li et al., 2019b**], des modèles entraînés sur l'ensemble de données de TableBank. Les colonnes verticales de Word, Latex et Word+Latex représentent le contenu des différents ensembles de test.

4.5. PubLayNet

4.5.1. Description

PubLayNet [**Zhong et al., 2019**] est un ensemble de données de grande taille pour la reconnaissance de la structure de documents. Les auteurs désiraient un ensemble de données assez grand pour entraîner des réseaux de neurones profonds. L'annotation à la main n'est pas réaliste et est beaucoup trop coûteuse. PubLayNet contient les annotations pour des paragraphes, titres, listes, tableaux and figures. Un exemple est montré à la figure 4.5. On s'intéresse à cet ensemble de données, car il contient beaucoup de documents avec des tableaux. L'ensemble de données avec les images contenant des tableaux est résumé au tableau 4.5.

Ensemble	PubLayNet	
	Train	Test
Nombre de documents	85 195	4 988
Nombre de tableaux	100 975	6 291

Tableau 4.5. Résumé de la taille de l'extraction de l'ensemble de données PubLayNet pour la détection de tables.

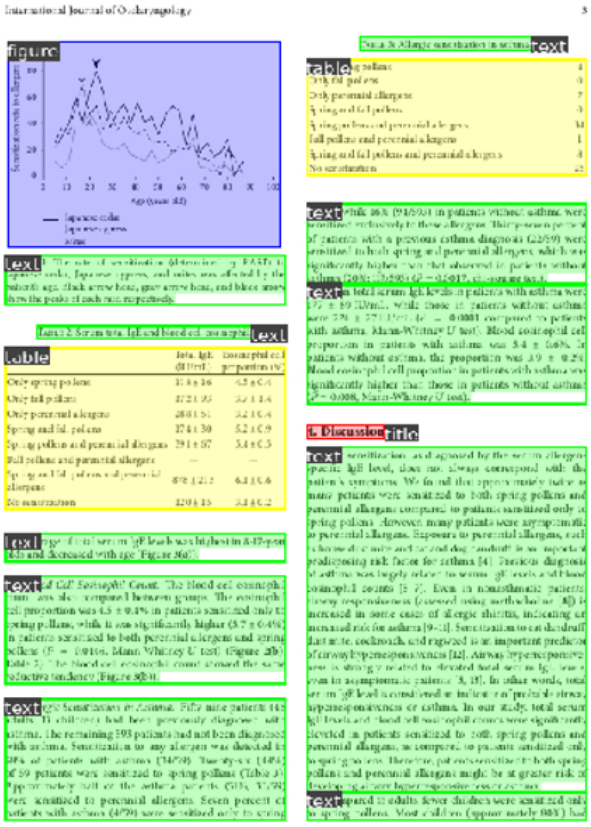


Fig. 4.5. Exemple de l'ensemble de données de PubLayNet où les tableaux sont encadrés en jaune, la figure est encadrée en bleu, les textes sont encadrés en vert et le titre de la section est encadré en rouge.

Cet ensemble de données contient des images de documents PDF de *PubMed Central* [**U.S. National Institutes of Health's National Library of Medicine, 2019**]. *PubMed Central Open Access* est un centre d'archive d'articles biomédicaux de plus de 34 millions d'articles. Les auteurs proposent une façon d'annoter automatiquement la structure de la mise en page d'un million d'articles. L'annotation automatique des documents se fait grâce au fait que *PubMed Central Open Access* contient les documents en formats PDF et XML. Le format XML contient les informations sur la mise en page. Les auteurs ont développé des méthodes pour aligner le format XML au PDF. La majorité des documents d'analyse de mise en page existant sont créés à l'aide d'annotations manuelles. Par exemple, c'était le cas avec les ensembles d'entraînement ICDAR2013 et ICDAR2019 mentionné plus haut.

4.5.2. Résultats

Les auteurs entraînent et testent des modèles Faster R-CNN et Mask R-CNN avec l'ensemble de données PubLayNet. Ces modèles sont décrits respectivement aux sections 3.1.2 et 3.1.3. Ils utilisent l'implémentation de Detectron, un *framework*, décrit à la

section 3.2. Les deux modèles utilisent ResNeXt-101-64x4d comme squelette, un modèle qui est décrit à la section 3.1.4.

La métrique utilisée par les auteurs est celle de mAP (mean average precision) avec des IoU [0.5:0.95]. Cette métrique est décrite à la section 2.5.

Les résultats des modèles sur l'ensemble sont présentés au tableau 4.6. Encore une fois, on présente les détails des résultats obtenus par les auteurs parce que PubLayNet est un des ensembles de données principalement étudié dans ce mémoire. On peut remarquer que les modèles ont plus de facilité à détecter les figures et tableaux étant donné leurs grandes formes rectangulaires consistantes.

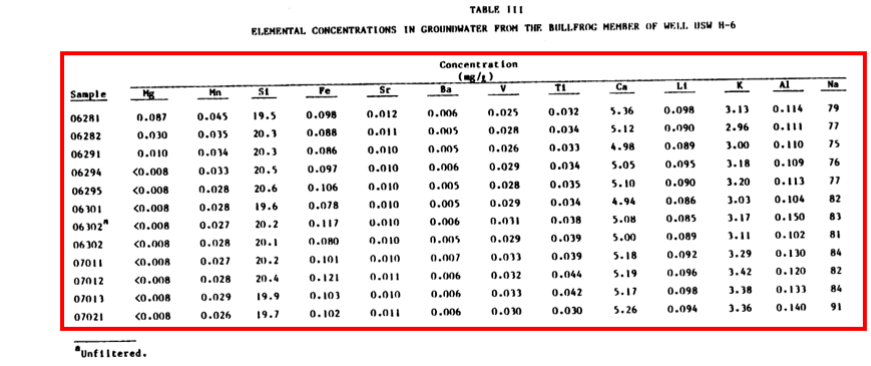
Category	Dev		Test	
	F-RCNN	M-RCNN	F-RCNN	M-RCNN
Text	0.910	0.916	0.913	0.917
Title	0.826	0.840	0.812	0.828
List	0.883	0.886	0.885	0.887
Table	0.954	0.960	0.943	0.947
Figure	0.937	0.949	0.945	0.955
Macro average	0.902	0.910	0.900	0.907

Tableau 4.6. Résultats présentés dans [**Zhong et al., 2019**] par les auteurs des modèles sur l'ensemble de données de PubLayNet.

4.6. Ensemble de données de Github

4.6.1. Description

Cet ensemble de données contient 403 images pour la tâche de détection de tableaux [**Panchal, 2019**]. Un exemple est présenté à la figure 4.6 qui représente la majorité de l'ensemble. Il est utilisé pour l'entraînement du modèle CascadeTabNet discuté à la section 3.4. Il n'y aucune information sur l'origine des documents.



18

TABLE III

ELEMENTAL CONCENTRATIONS IN GROUNDWATER FROM THE BULLFROG MEMBER OF WELL USW H-6

Sample	Concentration (µg/L) Mg	Mn	Si	Fe	Sr	Ba	V	Ti	Ca	Li	K	Al	Na
06281	0.087	0.045	19.5	0.098	0.012	0.006	0.025	0.032	5.36	0.098	3.13	0.114	79
06282	0.030	0.035	20.3	0.088	0.011	0.005	0.028	0.034	5.12	0.090	2.96	0.111	77
06291	0.010	0.034	20.3	0.086	0.010	0.005	0.026	0.033	4.98	0.089	3.00	0.110	75
06294	<0.008	0.033	20.5	0.097	0.010	0.006	0.029	0.034	5.05	0.095	3.18	0.109	76
06295	<0.008	0.028	20.6	0.106	0.010	0.005	0.028	0.035	5.10	0.090	3.20	0.113	77
06301	<0.008	0.028	19.6	0.078	0.010	0.005	0.029	0.034	4.94	0.086	3.03	0.104	82
06302[a]	<0.008	0.027	20.2	0.117	0.010	0.006	0.031	0.038	5.08	0.085	3.17	0.150	83
06302	<0.008	0.028	20.1	0.080	0.010	0.005	0.029	0.039	5.00	0.089	3.11	0.102	81
07011	<0.008	0.027	20.2	0.101	0.010	0.007	0.033	0.039	5.18	0.092	3.29	0.130	84
07012	<0.008	0.028	20.4	0.121	0.011	0.006	0.032	0.044	5.19	0.096	3.42	0.120	82
07013	<0.008	0.029	19.9	0.103	0.010	0.006	0.033	0.042	5.17	0.098	3.38	0.133	84
07021	<0.008	0.026	19.7	0.102	0.011	0.006	0.030	0.030	5.26	0.094	3.36	0.140	91

[a]Unfiltered.

Fig. 4.6. Un exemple de l'ensemble de données de Github où un tableau est encadré en rouge.

4.6.2. Résultats

Les auteurs ne présentent pas de modèle pour tester leur ensemble de données. Il n'y a pas de métrique associée à cet ensemble de données. Il n'y pas de résultats présentés, mais l'ensemble de données est utilisé pour entraîner le modèle CascadeTabNet (section 3.4).

4.7. Conclusion

Les nombreux ensembles de données de détection de tableaux sont tous associés à des métriques différentes et la majorité des travaux utilisent leurs propres références d'évaluation. Il est donc difficile de comparer les différents modèles et une étude plus systématique doit être effectuée afin d'y parvenir.

De plus, ces ensembles de données contiennent potentiellement des erreurs, car TableBank et PubLayNet sont générés automatiquement. Ces deux ensembles sont les plus pertinents à étudier étant donné leur grande taille et sachant qu'ils représentent la majeure partie des documents PDF d'aujourd'hui. Ces deux arguments sont importants dans une ère où les réseaux de neurones dominent.

Les prochains chapitres se concentrent alors à comparer et à analyser TableBank et PubLayNet ainsi qu'à explorer différentes pistes pour améliorer les modèles de détection de tableaux. En effet, on remarque que la plupart des auteurs utilisent des architectures de modèles semblables et c'est l'entraînement qui diffère.

Chapitre 5

Impact du corpus d'entraînement

Dans ce chapitre, on étudie l'influence de différents ensembles de données sur l'entraînement d'un modèle d'apprentissage profond pour la détection de tableaux dans des documents. Pour ce faire, on explore les ensembles de données TableBank et PubLayNet. Rappelons que TableBank (section 4.4) est composé de deux sous-ensembles provenant respectivement de Word et de Latex. Tandis que PubLayNet (section 4.5) est un ensemble de données pour la reconnaissance de la structure de documents qui possède un nombre considérable d'images contenant des tableaux. Ainsi, on crée un sous-ensemble de PubLayNet contenant seulement des annotations de tableaux. Au tableau 5.1 on présente la taille des sous-ensembles qui seront utilisés dans ce chapitre.

Ensemble	**TableBank**		**PubLayNet**
	Word	**Latex**	
Entraînement	73 383	187 199	85 195
Test	2 281	5 719	4 988

Tableau 5.1. Nombre d'exemples des ensembles de données d'entraînement et de test utilisés.

5.1. TableBank

Afin d'étudier l'importance du corpus d'entraînement, on commence par utiliser l'ensemble de données TableBank de [**Li et al., 2019b**]. Comme mentionné, les auteurs utilisent Detectron (section 3.2), une librairie qui offre des modèles préentraînés sur ImageNet de différentes tailles. L'ensemble de données de TableBank est disponible sur le GitHub des auteurs [**Li et al., 2019a**]. Les modèles entraînés par les auteurs sont également disponibles.

De plus, comme mentionné dans [**Padilla et al., 2021a**], le fait d'avoir des métriques non disponibles ou non compatibles avec un format, force les scientifiques à développer eux-mêmes les métriques, ce qui engendre des différences. Étant donné que les détails de la métrique utilisée par les auteurs de TableBank ne sont pas disponibles, la métrique a

été développée selon notre compréhension. La première étape est donc de s'assurer de la validité de la métrique développée en reproduisant les résultats de [**Li et al., 2019b**] avec leurs modèles et ensuite d'entraîner nous-mêmes des modèles à l'aide de Detectron.

5.1.1. Reproduction des résultats

Tout d'abord, on utilise les modèles entraînés par [**Li et al., 2019b**] afin de reproduire les résultats du papier et réaliser une comparaison. Ces modèles entraînés sont disponibles sur leur GitHub [**Li et al., 2019a**]. Il y a deux types de modèles utilisés, ResNeXt-101 et ResNeXt-152, décrits à la section 3.1.4, qui proviennent de la librairie Detectron, décrit à la section 3.2. Pour chacun de ces types de modèles, les auteurs entraînent trois modèles: l'un sur le sous-ensemble de Word, un autre sur le sous-ensemble de Latex et un modèle sur l'union des deux sous-ensembles de Word et Latex. Chacun de ces modèles est ensuite testé sur ces trois sous-ensembles, soit Word, Latex et Word-Latex. Il est à rappeler que les auteurs utilisent pour ensemble de test un échantillon aléatoire de 1000 exemples pour chaque sous-ensemble. Ainsi, l'ensemble test de Word contient 1000 échantillons aléatoires, l'ensemble test de Latex contient 1000 échantillons aléatoires et l'ensemble test de Word-Latex combine 1000 échantillons aléatoires de l'ensemble Word et 1000 échantillons aléatoires de l'ensemble Latex.

Afin d'évaluer ces modèles, on échantillonne à notre tour 1000 exemples aléatoires de Word et Latex. Ces ensembles seront fixes au long des expériences. Rappelons aussi que les prédictions de modèles entraînés par Detectron retournent un seuil de confiance. Ainsi, on débute par reproduire les résultats avec ces ensembles de données, les modèles des auteurs et le même seuil de confiance pour les prédictions de Detectron que les auteurs, soit 90%. Les résultats sont présentés au tableau 5.2. Afin de comparer les résultats, on ajoute en dessous de chaque résultat la différence avec ceux de l'article.

Au tableau 5.2, on s'aperçoit rapidement que tous les résultats divergent selon différentes ampleurs, mais il semble y avoir un biais constant qui rend nos résultats plus élevés. De plus, une différence notable que l'on remarque est que le modèle le plus performant sur l'ensemble de test Word-Latex utilise ResNeXt-101 avec un avantage de 0.22% sur le modèle de ResNeXt-152. En effet, [**Li et al., 2019b**] avait découvert que le modèle plus profond ResNeXt-152 performait toujours mieux que celui de ResNeXt-101.

On se questionne à savoir si ces écarts sont causés par le hasard de l'échantillonnage des ensembles de tests ou par la métrique. Afin de trouver la source des divergences, on effectue 10 expériences dont chacune prélève 1000 échantillons aléatoires des ensembles de données de test. Ainsi, on peut tenter d'estimer si la différence des résultats est due

Modèles	Word			Latex			Word-Latex		
	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1
Word ResNeXt-101	93.45 % (-1.51%)	94.35 % (+10.47%)	93.90% (+4.82%)	99.56% (+0.54%)	59.14% (-0.34%)	74.21% (-0.11%)	94.73% (-1.21%)	83.46% (+7.39%)	88.74% (+3.88%)
Word ResNeXt-152	94.16% (-1.14%)	94.23% (+5.94%)	**94.20%** (+2.54%)	99.56% (+1.48%)	71.54% (+2.64%)	83.26% (+2.32%)	95.47% (-0.56%)	87.21% (+5.12%)	91.16% (+2.65%)
Latex ResNeXt-101	84.89% (+2.01%)	93.16% (-0.79%)	88.84% (+0.77%)	98.83% (+0.29%)	98.24% (+0.64%)	98.53% (+0.46%)	88.92% (+1.48%)	94.72% (-0.40%)	91.73% (+0.61%)
Latex ResNeXt-152	84.19% (+1.60%)	93.27% (-2.35%)	88.50% (-0.13%)	98.89% (+0.22%)	98.57% (+1.03%)	**98.73%** (+0.63%)	88.41% (+1.21%)	94.90% (-1.34%)	91.54% (+0.24%)
Word-Latex ResNeXt-101	91.93% (-3.64%)	93.51% (+9.48%)	92.72% (+3.29%)	98.90% (+0.04%)	98.16% (+1.22%)	98.53% (+0.64%)	94.05% (-2.65%)	94.95% (+6.78%)	**94.50%** (+2.26%)
Word-Latex ResNeXt-152	92.32% (-3.08%)	92.64% (+6.25%)	92.48% (+1.81%)	98.88% (+0.03%)	97.79% (+0.44%)	98.33% (+0.25%)	94.33% (-2.24%)	94.24% (+4.35%)	94.28% (+1.17%)

Tableau 5.2. Résultats des modèles entraînés par [**Li et al., 2019b**] en utilisant la métrique de TableBank développée par nos soins. Les modèles sont listés à la première colonne, tandis que les ensembles de tests sont représentés par les colonnes nommées Word, Latex, Word-Latex. On utilise des échantillons de 1000 images. On présente en gras le meilleur score-F1 obtenu par le meilleur modèle pour cet ensemble de test.

au tirage aléatoire de l'ensemble de test. Les résultats sont présentés au tableau 5.3. On souligne les cas où les résultats des auteurs se trouvent dans l'intervalle.

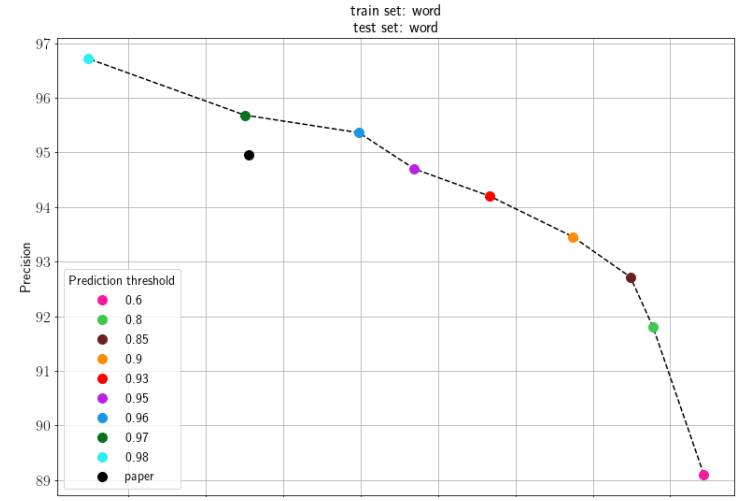
Modèles	Word			Latex			Word-Latex		
	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1
Word ResNeXt-101	[92.89, 94.30]%	[93.46, 94.39]%	[93.38, 94.17]%	[98.18, 99.65]%	[55.33, 62.24]%	[71.16, 76.57]%	[93.98, 95.33]%	[81.06, 84.28]%	[87.41, 89.25]%
Word ResNeXt-152	[92.86, 94.89]%	[93.81, 95.38]%	[93.76, 94.72]%	[98.36, 99.85]%	[66.64, 72.22]%	[79.69, 83.47]%	[94.76, 96.15]%	[84.89, 87.08]%	[89.55, 91.25]%
Latex ResNeXt-101	[82.87, 84.93]%	[92.47, 94.38]%	[87.49, 89.41]%	[97.38, 98.73]%	[97.76, 98.31]%	[97.60, 98.43]%	[87.39, 88.97]%	[94.50, 94.96]%	[90.90, 91.79]%
Latex ResNeXt-152	[83.34, 86.75]%	[91.25, 93.01]%	[87.37, 89.22]%	[97.48, 98.57]%	[97.12, 98.21]%	[97.38, 98.30]%	[87.59, 89.49]%	[93.47, 94.73]%	[90.81, 92.02]%
Word-Latex ResNeXt-101	[90.72, 92.70]%	[92.94, 94.71]%	[92.09, 93.26]%	[97.25, 98.89]%	[97.22, 98.22]%	[97.49, 98.39]%	[93.15, 94.71]%	[94.20, 95.26]%	[93.89, 94.97]%
Word-Latex ResNeXt-152	[91.28, 93.59]%	[91.42, 93.38]%	[91.35, 93.34]%	[97.72, 99.07]%	[96.74, 98.06]%	[97.51, 98.37]%	[93.72, 94.84]%	[93.38, 94.90]%	[93.72, 94.51]%

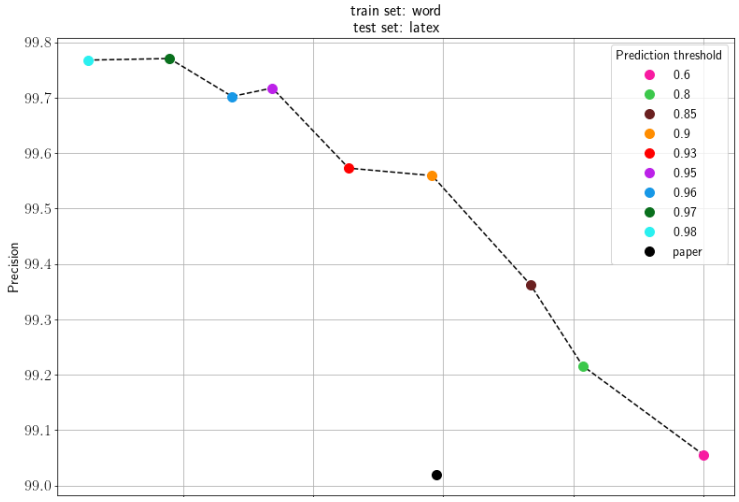
Tableau 5.3. Minimum et maximum de la précision, rappel et score-F1 de 10 expériences où on prélève 1000 échantillons aléatoires des ensembles de données de test de Word, Latex et Word-Latex. On surligne les intervalles dont les résultats des auteurs se trouvent à l'intérieur.

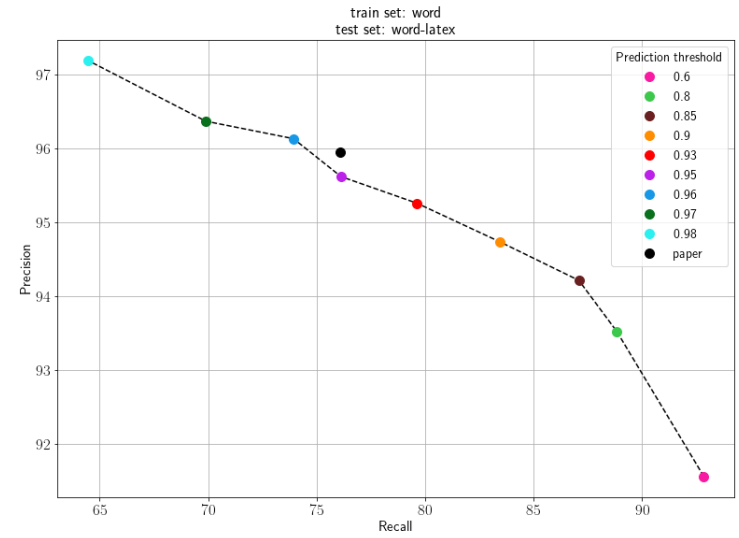
Tout d'abord, au tableau 5.3 on peut voir que 10/18 des intervalles sur le score-F1 contiennent les résultats du papier. Il est curieux de voir que l'ensemble de test Latex semblent être celui qui se rapproche le plus des résultats du papier. Au tableau 5.2, on pouvait voir que l'ensemble de test Latex est celui qui contenait les plus petites différences avec les résultats des auteurs. Une cause de ce phénomène pourrait être les erreurs d'annotations qui sont contenues dans le sous-ensemble Word. En effet, celui-ci contient beaucoup de tableaux mal annotés. Cette problématique est discutée plus en détail dans la section 5.2. Ces erreurs font alors que l'échantillonnage a beaucoup plus d'importance étant donné que le sous-ensemble est moins stable.

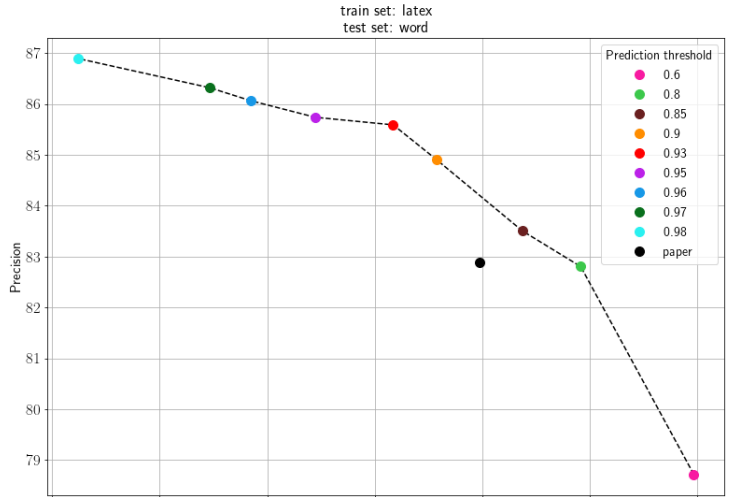
Il est aussi intéressant de noter que les modèles qui performent le mieux pour un certain ensemble test sont les plus stables pour différents tests. Par exemple, le modèle entraîné sur Word et testé sur l'ensemble Latex varie beaucoup, car c'est le modèle qui engendre les plus faibles performances pour cet ensemble test.

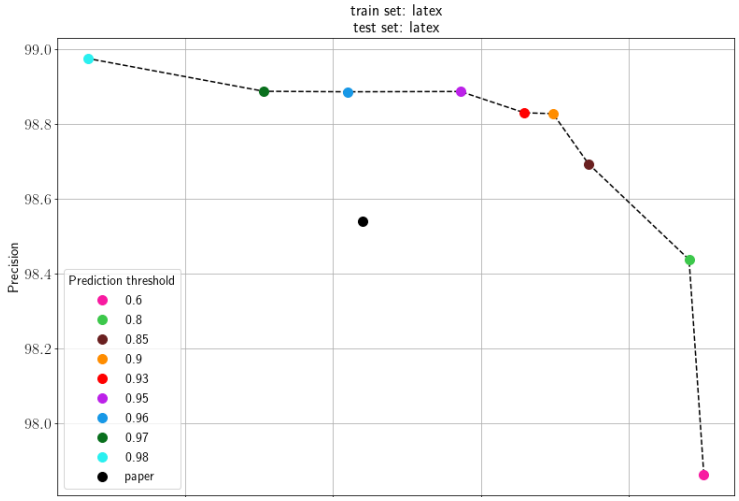
Étant donné que plusieurs résultats des auteurs ne sont pas dans l'intervalle, on semble alors observer une réelle différence dans les métriques. On peut se questionner à savoir s'il y aurait un moyen de se rapprocher de la métrique des auteurs. Pour s'y faire, on produit la courbe de précision et rappel avec les différents seuils de confiance pour chaque cas d'ensemble d'entraînement et de test avec le modèle utilisant le squelette ResNeXt-101 à la figure 5.1.

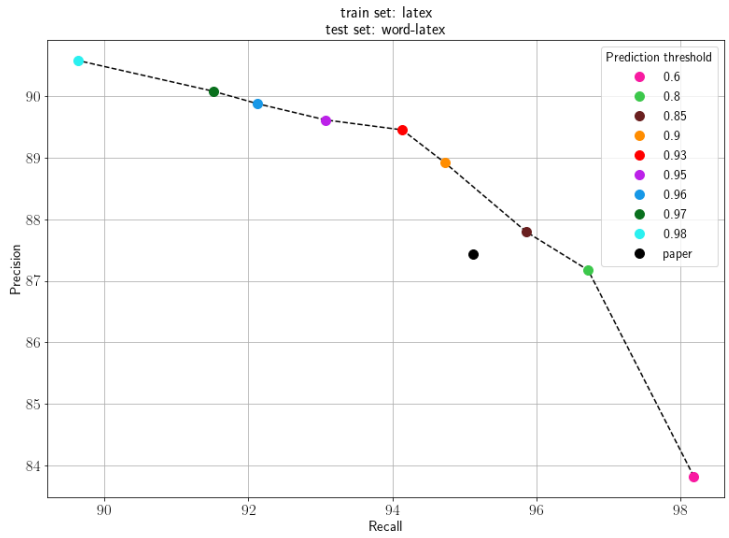

train set: word
test set: word
Precision
Prediction threshold
0.6
0.8
0.85
0.9
0.93
0.95
0.96
0.97
0.98
paper


train set: word
test set: latex
Precision
Prediction threshold
0.6
0.8
0.85
0.9
0.93
0.95
0.96
0.97
0.98
paper


train set: word
test set: word-latex
Precision
Recall
Prediction threshold
0.6
0.8
0.85
0.9
0.93
0.95
0.96
0.97
0.98
paper


train set: latex
test set: word
Precision
Prediction threshold
0.6
0.8
0.85
0.9
0.93
0.95
0.96
0.97
0.98
paper


train set: latex
test set: latex
Precision
Prediction threshold
0.6
0.8
0.85
0.9
0.93
0.95
0.96
0.97
0.98
paper


train set: latex
test set: word-latex
Precision
Recall
Prediction threshold
0.6
0.8
0.85
0.9
0.93
0.95
0.96
0.97
0.98
paper

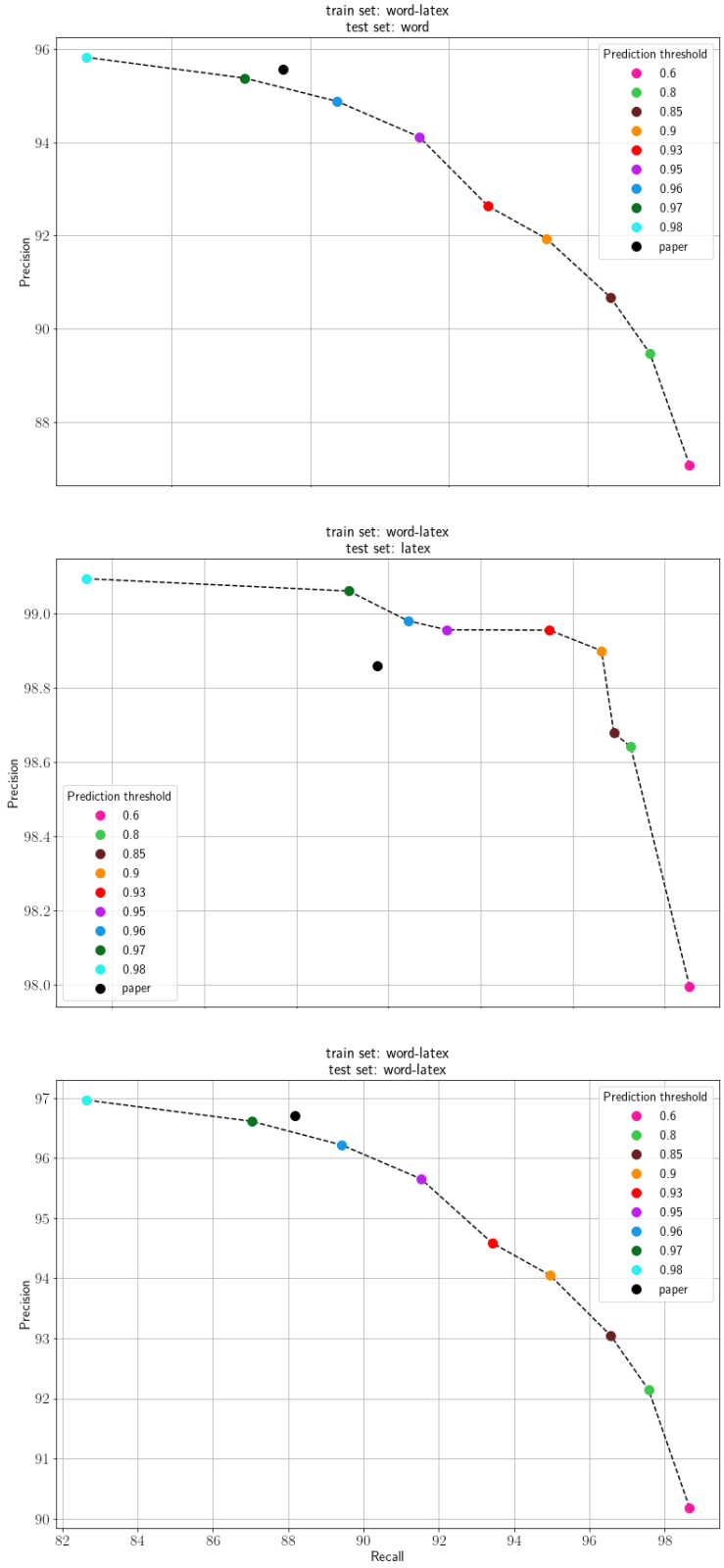


Fig. 5.1. Courbe de précision et rappel avec différents seuils de confiance pour chaque cas d'entraînement et de test. La précision et le rappel produits par les auteurs sont indiqués par un point noir, tandis que chaque couleur représente différents seuils de confiance. Le titre de chaque graphique précise l'ensemble d'entraînement et de test.

À la figure 5.1, on relève que les résultats des auteurs ne sont jamais sur la courbe de précision-rappel. En effet, 6/9 des graphiques semblent indiquer qu'un taux de confiance entre 95% et 97% permet de se rapprocher le plus des auteurs, tandis que les autres graphiques montrent qu'un taux de confiance autour de 80% en serait plus proche. Il semble alors que, pour certains cas, augmenter le seuil de confiance permet de se rapprocher des résultats des auteurs. On tente alors encore une fois de produire les résultats avec les mêmes ensembles de test, mais cette fois avec un taux de confiance de 95%. Les résultats sont présentés au tableau 5.4.

Modèles	**Word**			**Latex**			**Word-Latex**		
	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1
Word ResNeXt-101	94.70 % (-0.26%)	89.24 % (-0.26%)	91.89% (+2.81%)	99.71% (+0.69%)	46.85% (-12.63%)	63.75% (-10.57%)	95.62% (-0.32%)	76.13% (+0.06%)	84.76% (-0.1%)
Word ResNeXt-152	94.83% (-0.47%)	90.41% (+2.12%)	**92.57%** (+0.91%)	99.67% (+1.59%)	60.86% (-8.04%)	75.57% (-5.37%)	95.91% (-0.12%)	81.27% (-0.82%)	87.98% (-0.53%)
Latex ResNeXt-101	85.74% (+2.86%)	90.90% (-3.05%)	88.24% (+0.17%)	98.89% (+0.35%)	97.93% (+0.33%)	**98.41%** (+0.34%)	89.62% (+2.18%)	93.07% (-2.05%)	91.31% (+0.19%)
Latex ResNeXt-152	85.44% (+2.85%)	90.27% (-5.35%)	87.79% (-0.84%)	98.93% (+0.26%)	97.75% (+0.21%)	98.34% (+0.24%)	89.42% (+2.22%)	92.58% (-3.66%)	90.97% (-0.52%)
Word-Latex ResNeXt-101	94.12% (-1.45%)	88.94% (+4.91%)	91.45% (+2.02%)	98.96% (+0.10%)	97.32% (+0.38%)	98.13% (+0.24%)	95.65% (-1.05%)	91.53% (+3.36%)	**93.55%** (+1.31%)
Word-Latex ResNeXt-152	94.52% (-0.88%)	86.89% (+0.5%)	90.55% (-0.12%)	98.94% (+0.09%)	96.92% (-0.40%)	97.92% (-0.16%)	95.95% (-0.62%)	90.00% (+0.11%)	92.88% (-0.23%)

Tableau 5.4. Résultats des modèles entraînés par les auteurs en utilisant la métrique de TableBank développer par nos soins. On utilise les mêmes échantillons de 1000 images, mais avec un seuil de confiance de 95%. On présente en gras le meilleur score-F1 obtenu par le meilleur modèle pour cet ensemble de test.

Afin de comparer les résultats du tableau 5.2 avec le seuil de confiance de 90% et ceux du tableau 5.4 avec le seuil de confiance de 95%, on produit une moyenne des différences absolues des deux tableaux. On réalise cette moyenne sur les scores-F1 de chaque modèle sur chaque ensemble test. Avec le seuil de confiance de 90%, on obtient une moyenne de 1.59, tandis qu'avec le seuil de confiance de 95%, on obtient une moyenne de 1.48. On semble alors avoir une plus petite différence absolue avec le seuil de 95%.

De plus, encore une fois, le modèle le plus performant sur l'ensemble de test Word-Latex utilise ResNeXt-101 avec un avantage de 0.67% sur le modèle de ResNeXt-152. On rappelle que précédemment, cet avantage était de seulement 0.22%. Ainsi, augmenter le taux de confiance donne un avantage plus proéminent au modèle gagnant incorrect. Cette problématique est aussi soulignée avec le modèle entraîné sur Latex et testé sur

Latex. Le modèle ResNeXt-101 performe mieux que ResNeXt-152. Ainsi, malgré que les différences absolues sont en moyenne plus petites, augmenter le seuil de confiance ne permet pas de se rapprocher des résultats des auteurs, car les modèles gagnants, soit le modèle ayant les meilleures performances, diffèrent davantage.

Au final, ces performances laissent penser qu'une réelle différence dans les métriques existe. Cette différence semble assez mineure pour qu'on puisse simplement reconnaître cette différence sans chercher la cause plus profondément. On continuera alors d'utiliser un seuil de confiance de 90%.

5.1.2. Ajout de PubLayNet

Étant donné que dans ce chapitre on s'intéresse à l'impact de l'ensemble d'entraînement, on ajoute à l'étude l'ensemble de données PubLayNet. On commence par mesurer les résultats du modèle des auteurs sur l'ensemble de données PubLayNet. Encore une fois, on produit un ensemble de test de 1000 images aléatoires qui sont fixées au long de ce chapitre. Les résultats sont présentés au tableau 5.5.

Modèles	**PubLayNet**		
	Précision	Rappel	Score-F1
Word ResNeXt-101	98.08%	60.85%	75.11%
Word ResNeXt-152	98.15%	67.79%	80.19%
Latex ResNeXt-101	96.16%	94.84%	**95.50%**
Latex ResNeXt-152	95.60%	95.36%	95.48%
Word-Latex ResNeXt-101	96.78%	94.10%	95.42%
Word-Latex ResNeXt-152	96.56%	93.92%	95.22%

Tableau 5.5. Résultats des modèles des auteurs sur l'ensemble de test PubLayNet.

Au tableau 5.5, on note que les modèles de [**Li et al., 2019b**] performent bien sur ces nouvelles données et que le comportement des résultats est semblable à l'ensemble de test Latex. En effet, le modèle le plus efficace est le modèle entraîné sur l'ensemble Latex. Comme il a été montré précédemment, un modèle entraîné avec des données semblables à l'ensemble de test est important pour les performances.

Afin de mesurer l'impact de l'ajout d'un ensemble de données, il est nécessaire d'avoir la liberté d'entraîner un modèle avec les données désirées. Pour s'y faire, il

est nécessaire de procéder aux mêmes étapes que les auteurs, soit utiliser la librairie Detectron afin d'affiner des modèles préentraînés de Detectron. Les configurations utilisées par les auteurs de TableBank nécessitent trop de mémoire pour l'ordinateur à notre disposition. Notre ordinateur utilise le GPU *Nvidia Titan V* avec 12GB de mémoire et 640 Tensor Cores. Ainsi, on emploie Detectron, mais en utilisant un modèle plus léger. On utilise le modèle Faster R-CNN avec une profondeur de 50 qui est préentraîné sur un ensemble de 118 000 images avec environs 80 catégories comme 'ballon', 'chien', 'livre', etc. On réutilise les mêmes ensembles tests et on produit de nouvelles combinaisons d'ensembles d'entraînement et de test. Les résultats sont présentés au tableau 5.6.

Modèles	Word	Latex	PubLayNet	Word-Latex	Word-PubLayNet	Latex-PubLayNet	Word-Latex-PubLayNet
Word	**89.74%**	85.46%	85.97%	87.72%	88.10%	86.91%	88.21%
Latex	88.69%	**95.90%**	91.73%	88.46%	88.28%	93.11%	89.66%
PubLay-Net	82.51%	93.10%	**96.77%**	85.73%	89.25%	95.56%	89.97%
Word-Latex	89.44%	88.46%	89.82%	89.52%	88.65%	91.22%	89.63%
Word-PubLayNet	89.32%	93.70%	95.95%	**92.55%**	**92.30%**	95.21%	**92.56%**
Latex-PubLayNet	84.60%	95.41%	95.65%	87.82%	89.65%	**95.57%**	90.73%
Word-Latex-PubLayNet	88.03%	94.78%	93.82%	90.00%	90.64%	94.14%	91.40%

Tableau 5.6. Score-F1 des modèles entraînés en utilisant la métrique de TableBank développée par nos soins. On utilise les échantillons de 1000 images prédéfinies avec un taux de confiance de 90%. On présente en gras le meilleur score-F1 obtenu par le meilleur modèle pour cet ensemble de test. Les résultats avec les précisions et rappel sont présentés en annexe au tableau A.1.

Au tableau 5.6, on note que les nouveaux modèles plus légers produisent des score-F1 avec un désavantage d'autour 5% sur ceux des auteurs de TableBank pour les ensembles test de Word, Latex et Word-Latex. Pour les ensembles de test individuels de Word, Latex et PubLayNet, les modèles entraînés sur ce même ensemble d'entraînement performent le mieux. Pour tous les autres cas, l'ajout de PubLayNet à l'ensemble d'entraînement améliore nettement les performances. En pratique, ce désavantage de 5% n'est pas très notable.

Par contre, on remarque que le modèle entraîné sur la combinaison de Word, Latex et PubLayNet ne produit jamais les meilleurs résultats. Les auteurs de TableBank avaient un résultat semblable. Le modèle entraîné sur Word et Latex performait le mieux sur

l'ensemble de test de Word et Latex. En effet, on remarquait qu'avoir cet ensemble d'entraînement diversifié n'améliorait pas les performances sur les tests contenant uniquement Word et uniquement Latex. Par contre, on s'attendait à ce que le modèle entraîné sur Word, Latex et PubLayNet génère au moins les meilleures performances sur l'ensemble de test provenant de ces trois ensembles. Cette faiblesse peut être expliquée par le fait que le modèle n'est pas assez complexe pour tirer profit de ce gros ensemble d'entraînement. Il est alors logique que le modèle entraîné sur Word et PubLayNet soit le plus efficace pour l'ensemble de test diversifié, parce que Latex et PubLayNet sont deux ensembles semblables, et en sachant que le modèle léger excelle avec un plus petit ensemble d'entraînement, soit PubLayNet.

Toutes ces expériences démontrent qu'avoir un ensemble d'entraînement semblable à l'ensemble de test est un facteur considérable afin d'assurer une réussite.

5.2. Erreurs de TableBank

Il est intéressant de comprendre d'où proviennent les erreurs du modèle afin de découvrir s'il y a un moyen d'améliorer les performances. Les images résultant des modèles sont étudiées à cette fin et des exemples sont présentés à la figure 5.2. Les erreurs les plus communes sont lorsque le modèle considère une figure comme un tableau, manque complètement un tableau, ou encadre incorrectement le tableau.

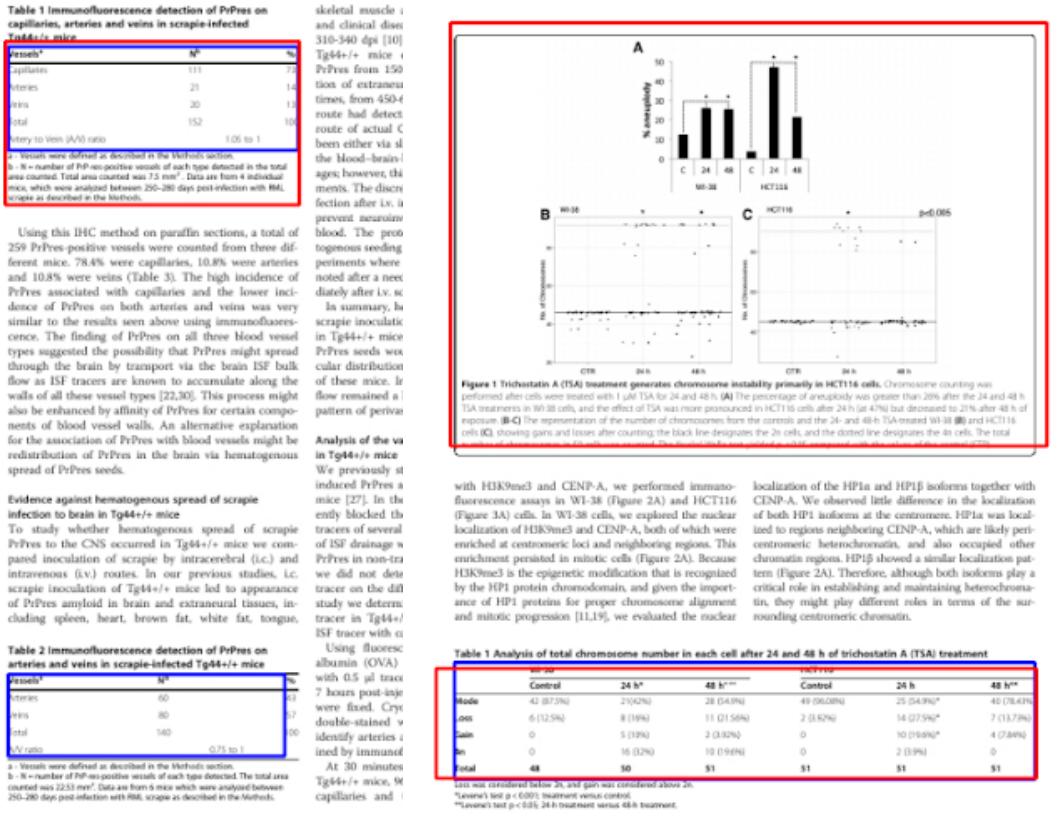


Fig. 5.2. Exemples d'erreurs que les modèles peuvent causer. Les prédictions sont présentées en rouge et les labels sont présentés en bleu.

Toutefois, un autre type d'erreur apparait, soit des erreurs de labels. Il arrive parfois que la mauvaise performance sur une image soit due à une erreur de label, et non à une erreur du modèle. Des exemples sont présentés à la figure 5.3. Ces cas ont des précisions très faibles, même si le modèle a bien prédit les tableaux. Il arrive souvent que des tableaux soient mal annotés ou simplement complètement manqués. Il semble y avoir une tendance à manquer des tableaux quand l'image en contient plus d'un. De plus, les erreurs de label paraissent plus présentes dans l'ensemble de données Word.

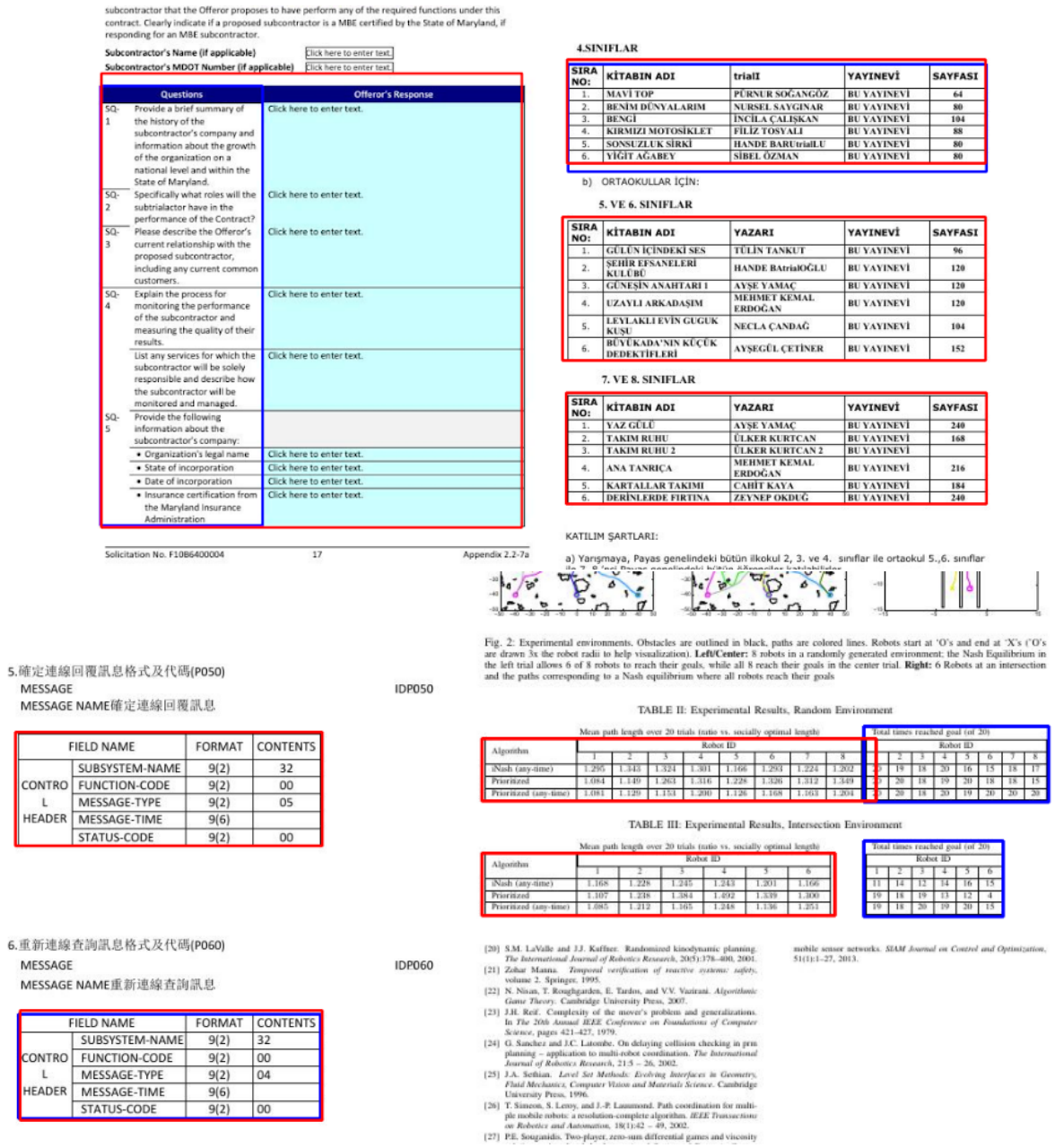


Fig. 5.3. Exemples d'erreurs de labels contenues dans l'ensemble test Word. Les prédictions sont présentées en rouge et les labels sont présentés en bleu.

Afin d'obtenir une approximation sur les erreurs, on extrait les images de Word avec un score-F1 de moins de 5% sur le total de 2281 images de l'ensemble de test. On extrait 428

images pour ensuite les analyser à la main. On trouve que 240/428 = 56.07% des images ont des erreurs de label. Sur l'ensemble de test total, ça devient 240/2281 = 10.52%. Ainsi, au moins 10% de l'ensemble de test de Word contient des erreurs de label et ce taux pourrait être encore plus élevé. La même étude est effectuée sur l'ensemble de test de Latex. Sur les 5719 images de test, 366 images ont un score-F1 de moins de 5%. Cette fois, seulement 12/366 = 3.28% ont des erreurs de label, soit un estimé d'au moins 16/5719 = 0.21% au total. Pour PubLayNet, sur les 4988 images de test, 122 images ont un score-F1 de moins de 5%. Seulement 13/122 = 10% ont des erreurs de labels, soit un estimé d'au moins 13/4988 = 0.26% au total. Les erreurs de label semblent alors particulièrement problématiques pour l'ensemble Word. Les auteurs utilisent les *tags* *<w:tbl>* du fichier xml source des documents Word. Par contre, les auteurs eux-mêmes précisent que c'est une annotation qui représente habituellement un tableau, mais que ce n'est pas une marque immuable. Ainsi, il semblerait que des tableaux Word peuvent être créé sans ce *tag* et que ce n'est pas un encadrement rigide.

Ces erreurs ont certainement un impact sur l'évaluation de l'ensemble de test de Word, en sachant qu'au moins 56% des images qui engendrent un score-F1 de moins de 5% contient des erreurs de label. L'effort nécessaire pour étudier l'ensemble d'entraînement complet et corriger les erreurs d'annotation est trop grand. Il reste important de noter que l'annotation automatique de TableBank génère des imperfections qui ajoutent un biais sur les résultats. Bref, ces résultats jettent un doute sur le taux des auteurs qui affirmaient que sur 1000 images de l'ensemble de données total, seulement 5 contenaient des erreurs de label, soit 0.5%. Cette problématique a aussi été pointée par les auteurs de CascadeTabNet [**Prasad et al., 2020b**] qui ont retiré les images avec des annotations erronées sur leurs ensemble de test Word de 1000 exemples. Par contre, ils ne précisent pas le nombre d'images retiré.

Bref, étant donné que la principale force de TableBank est d'être un des plus grands ensembles de données pour la détection et reconnaissance de tableaux, il aurait été nécessaire que les auteurs effectuent une étude plus importante sur les erreurs de labels qui sont non négligeables dans l'ensemble de test Word.

5.3. CascadeTabNet

CascadeTabNet [**Prasad et al., 2020b**] (section 3.4) est un modèle entraîné spécifiquement pour la reconnaissance et la détection de tableaux. Le modèle utilise un réseau de convolution. On reprend alors le tableau 5.6 en ajoutant le modèle CascadeTabNet afin de mettre en contraste les résultats représentés au tableau 5.7.

Au tableau 5.7, on voit que les performances de CascadeTabNet avoisinent ceux de TableBank. C'est un résultat intéressant sachant que l'ensemble d'entraînement est

Modèles	Word	Latex	PubLayNet	Word-Latex	Word-PubLayNet	Latex-PubLayNet	Word-Latex-PubLayNet
Word	**89.74%**	85.46%	85.97%	87.72%	88.10%	86.91%	88.21%
Latex	88.69%	**95.90%**	91.73%	88.46%	88.28%	93.11%	89.66%
PubLay-Net	82.51%	93.10%	**96.77%**	85.73%	89.25%	95.56%	89.97%
Word-PubLayNet	89.32%	93.70%	95.95%	**92.55%**	**92.30%**	95.21%	**92.56%**
Latex-PubLayNet	84.60%	95.41%	95.65%	87.82%	89.65%	**95.57%**	90.73%
Cascade-TabNet	72.04%	82.15%	91.51%	75.20%	81.62%	88.51%	81.72%

Tableau 5.7. Score-F1 des meilleurs modèles entraînés et du modèle CascadeTabNet en utilisant la métrique de TableBank développer par nos soins. On utilise les échantillons de 1000 images prédéfinies avec un taux de confiance de 90%. On présente en gras le meilleur score-F1 obtenu par le meilleur modèle pour cet ensemble de test. Les résultats avec les précisions et rappel sont présentés en annexe au tableau A.1.

beaucoup plus petit que les modèles utilisant Detectron, et qu'il n'est pas composé d'un ensemble ressemblant au test. Le modèle est le plus efficace sur l'ensemble PubLayNet. Il est à noter que [**Prasad et al., 2020b**] utilise la métrique de TableBank afin de comparer leurs résultats avec ceux de TableBank. Par contre, les résultats produits sur l'ensemble test de TableBank sont autour de 5% plus bas que ceux produits par les auteurs de TableBank. Cette différence est due au fait que les auteurs utilisent un ensemble d'entraînement beaucoup plus petit, soit de 1500 exemplaires pour Word, 1500 exemplaires pour Latex et la combinaison des deux ensemble précédent pour Word-Latex. Ainsi, les auteurs de CascadeTabNet affirment que leur modèle est plus performant, mais c'est seulement parce qu'ils utilisent un plus petit ensemble d'entraînement pour comparer le modèle de CascadeTabNet et TableBank.

De plus, les résultats que l'on génère avec le modèle CascadeTabNet sont beaucoup plus faibles que ceux présentés par [**Prasad et al., 2020b**]. En effet, les auteurs obtenaient un score-F1 de 94.92% sur Word, 96.60% sur Latex et 94.33% sur Word-Latex. Cette différence ne pourrait pas être expliquée par la différence dans l'implémentation de la métrique ni par l'échantillonnage des 1000 exemplaires. Cette divergence est curieuse et l'origine n'est pas connue. Par contre, on peut se questionner à savoir si la métrique de TableBank est réellement en mesure de correctement comparer les deux modèles. On s'intéresse alors à une nouvelle information: la performance selon le nombre de tableaux dans l'image. Ce nouvel élément est étudié à la prochaine section 5.4.

5.4. Tableaux multiples

À la suite de quelques observations, on découvre que les modèles semblent éprouver plus de difficultés à détecter les tableaux lorsqu'un document en possède plusieurs, comme on peut le voir à la figure 5.3. On commence par dénombrer le nombre d'exemples contenant une certaine quantité de tableaux par image pour chaque ensemble de test et d'entraînement au tableau 5.8.

#Tab	Test			Train		
	Word	**Latex**	**PubLayNet**	**Word**	**Latex**	**PubLayNet**
1	1788 78.39%	4563 79.78%	3911 78.22%	56 819 77.43%	146 376 79.80%	71 562 84.03%
2	387 16.97%	888 15.53%	916 18.32%	12 676 17.27%	29 363 15.69%	11 834 13.90%
3	70 3.07%	201 3.51%	135 2.70%	2 719 3.80%	5 992 3.20%	1 395 1.64%
4	27 1.18%	43 0.75%	32 0.64%	780 1.06%	1 708 0.91%	307 0.36%
5	5 0.22%	15 0.26%	4 0.08%	216 0.29%	405 0.22%	45 0.05%
6	3 0.13%	7 0.12%	2 0.04%	67 0.09%	207 0.11%	9 0.01%

Tableau 5.8. Distribution du nombre de tableaux par image pour chaque ensemble ainsi que la proportion que ce nombre représente en dessous.

Le tableau 5.8 indique que la grande majorité des exemples ne contiennent qu'un seul tableau. Aussi, il est à noter qu'un bon ensemble de données contient une distribution semblable sur ses caractéristiques pour l'ensemble d'entraînement et l'ensemble de test. C'est ce qu'on observe pour les ensembles de PubLayNet et TableBank, les proportions de tableaux par image sont relativement les mêmes sur l'entraînement et le test. Malgré que TableBank semble être légèrement mieux distribué que PubLayNet. Il est aussi intéressant d'observer que PubLayNet possède essentiellement la même distribution que l'ensemble de TableBank de Word et Latex.

Étant donné que la majorité des exemples de l'ensemble d'entraînement contiennent un seul tableau, on mesure la performance des modèles en divisant les ensembles de test selon le nombre de tableaux par image. Les résultats sont présentés à la figure 5.4 pour les modèles de TableBank et CascadeTabNet. On utilise le modèle de TableBank entraîné sur l'ensemble de Word et Latex.

À la figure 5.4, on remarque que le modèle de CascadeTabNet est beaucoup plus stable au travers les ensembles que le modèle de TableBank qui lui voit son score-F1

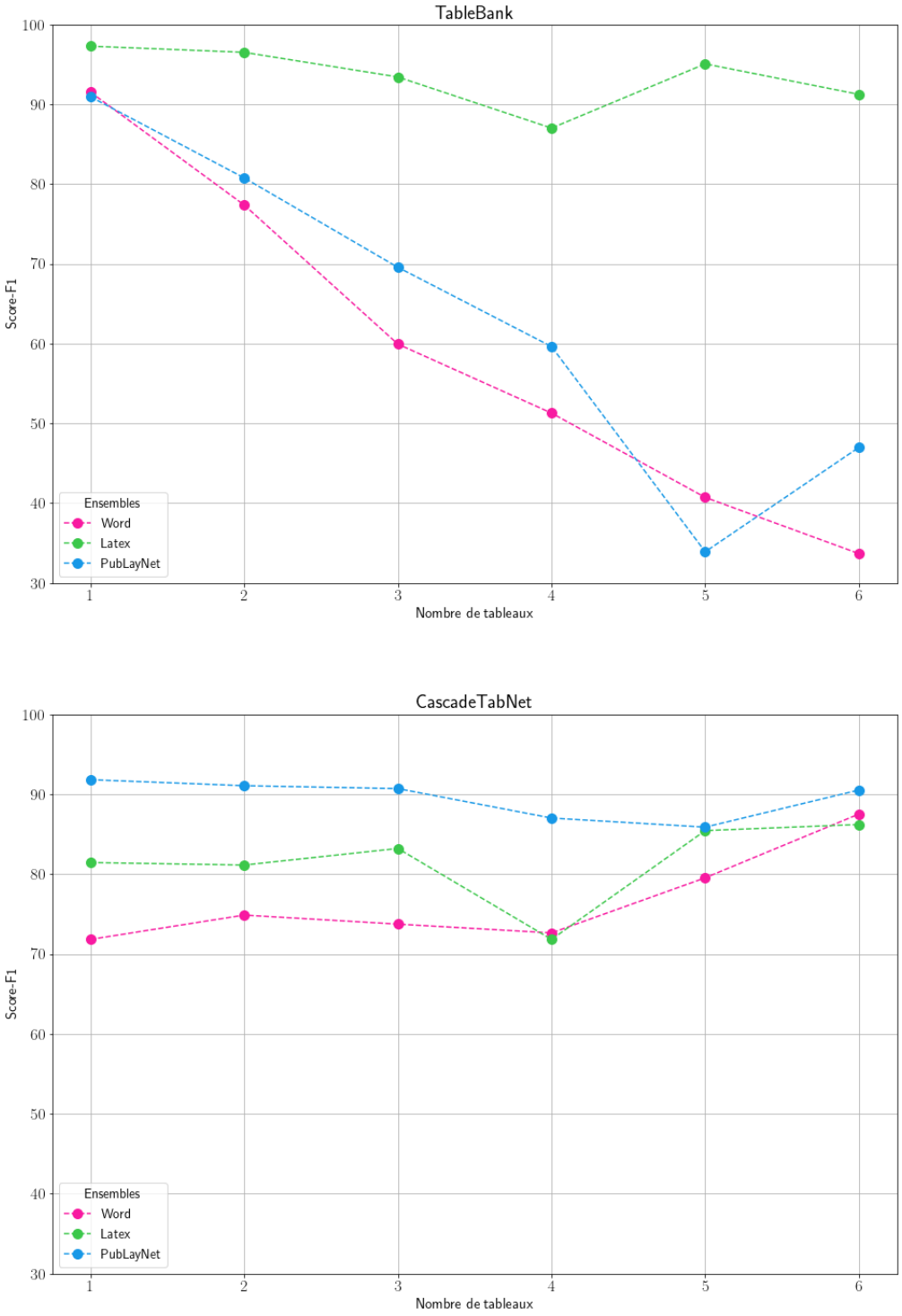


Fig. 5.4. Comparaison du score-F1 en fonction du nombre de tableaux par images sur les ensembles de test avec les modèles de TableBank et CascadeTabNet.

fortement diminué si le nombre de tableaux dans l'image est trop élevé. Ainsi, malgré que TableBank offrent de meilleures performances avec la métrique de TableBank proposée dans [**Gilani et al., 2017**], la métrique ne prend pas en compte l'avantage de CascadeTabNet sur la quantité de tableaux.

Il n'en reste pas moins que cet atout de CascadeTabNet est étonnant. Il pourrait être dû à une meilleure technique d'entraînement, même si le modèle utilise un ensemble d'entraînement plus petit que TableBank. En effet, CascadeTabNet emploie différentes méthodes d'augmentation de données afin d'enrichir l'ensemble d'entraînement. Ces transformations sont alors testées à la prochaine section 5.5 pour les modèles de Detectron entraîné sur TableBank.

5.5. Transformations

Comme mentionné précédemment, CascadeTabNet utilise des méthodes d'augmentation de données sur l'ensemble d'entraînement. Les auteurs utilisent une transformation par élargissement (*dilation*) et la transformation par bavure (*smudge*). La première consiste à grossir les régions contenant des pixels noirs en convertissant les images en noir et blanc. Tandis que pour la transformation par bavure, l'image est convertie en noir et blanc puis un effet de flou sur les pixels noirs est appliqué. L'effet de flou est créé en appliquant diverses transformations de distance comme *Euclidean Distance Transform*, *Linear Distance Transform* et *Max Distance Transform*. Des exemples des deux transformations sont présentés à la figure 5.5.

Fig. 5.5. Exemple de transformation par élargissement et par bavure de [**Prasad et al., 2020b**].

On applique ces transformations aux ensembles d'entraînement de Word, Latex et PubLayNet à l'aide du GitHub des auteurs [**Prasad et al., 2020a**] contenant le code pour appliquer les deux transformations. Comme il avait été vu précédemment, on avait l'hypothèse que le modèle Detectron entraîné par nos soins n'était pas assez complexe pour maximiser le potentiel d'un grand ensemble d'entraînement. Ainsi, on teste en

appliquant les transformations à seulement l'ensemble Word et seulement l'ensemble Latex. Les résultats sont présentés au tableau 5.9.

Modèles	Word	Latex	PubLayNet	Word-Latex
Word	89.74%	85.46%	85.97%	87.72%
Word + B	88.88%	69.02%	39.46%	82.84%
Word + E	89.01%	84.11%	72.78%	87.65%
Word + B + E	**90.24%**	78.78%	59.20%	87.13%
Latex	88.69%	**95.90%**	**91.73%**	88.46%
Latex + B	83.24%	91.44%	81.47%	85.68%
Latex + E	86.34%	93.93%	88.70%	85.53%
Latex + B + E	85.87%	94.91%	87.95%	**88.47%**

Tableau 5.9. Scores-F1 des modèles entraînés avec une augmentation de données en utilisant la métrique de TableBank développée par nos soins. On utilise les échantillons de 1000 images prédéfinis avec un taux de confiance de 90%. Les modèles sont définis avec un B pour l'ajout des images avec bavure et E pour l'ajout des images avec élargissement. Les résultats avec les précisions et rappel sont présentés en annexe au tableau A.2.

Au tableau 5.9, on remarque que la transformation par élargissement semble être plus avantageuse que la transformation par bavure sur l'ensemble Word et Latex. En revanche, l'addition des transformations par élargissement et bavure améliore les performances sur l'ensemble de Word. Ce qui est logique puisque le modèle est entraîné sur un grand nombre d'images de Word, tout en gardant une taille raisonnable. Par contre, sur tous les autres ensemble de test, l'ajout de transformation diminue la qualité. Un phénomène semblable se produit sur l'ensemble de données Latex. L'ajout de transformation n'améliore pas les résultats, sans non plus fortement les détériorer. Néanmoins, l'ajout des deux transformations sur l'ensemble d'entraînement Latex augmente le score-F1 de 0.01% sur l'ensemble test de Word-Latex. Bref, il est difficile de mesurer si la faible influence de l'augmentation de données est due à son impertinence ou a un besoin d'un modèle plus complexe.

5.6. Conclusion

Tout d'abord, on a reproduit les résultats des auteurs de TableBank avec notre propre modèle et en utilisant leurs métriques. Étant donné que le code de la métrique n'est pas disponible, celle-ci a été développée selon notre compréhension. On semblait remarquer

une différence plutôt minime dans les métriques. De plus, il a été démontré qu'ajouter l'ensemble de données PubLayNet à l'ensemble d'entraînement avait un impact positif et qu'avoir un modèle plus complexe, permettant de mettre en valeur un plus gros ensemble d'entraînement, serait bénéfique pour avoir un modèle plus général. En effet, le modèle performe toujours le mieux lorsqu'il a été entraîné avec un ensemble d'entraînement qui est semblable à l'ensemble de test. Ainsi, puisque l'objectif est typiquement d'avoir un modèle qui est robuste à des ensembles de test variés, avoir un ensemble d'entraînement varié semble important.

De surcroît, un problème important de l'ensemble de données TableBank a été pointé, soit les erreurs d'annotations du sous-ensemble Word. Il serait donc pertinent d'améliorer l'outil produisant les annotations automatiques, les performances d'un modèle seraient certainement mieux évaluées.

Le modèle CascadeTabNet a été comparé au modèle de Detectron. Ce dernier performait toujours mieux que CascadeTabNet. L'augmentation de données utilisée par CascadeTabNet ne semblait pas être bénéfique pour les ensembles de TableBank puisque ce sont déjà de grands ensembles diversifiés. Toutefois, malgré les résultats moins avantageux de CascadeTabNet, on remarque que le modèle est beaucoup plus stable avec des cas de documents contenant plusieurs tableaux. Cette découverte amorce le prochain chapitre 6 qui examine l'importance des métriques sur l'étude des modèles.

Chapitre 6

Impact des métriques

Dans ce chapitre, on explore les différentes métriques utilisées pour la détection d'objets et leur influence sur la détermination du meilleur modèle. En effet, le domaine de détection d'objets connaît un manque d'uniformité dans les formats de données et des métriques. On rappelle que ce problème est soulevé par [**Padilla et al., 2021a**] qui ont créé un outil contenant une grande quantité de métriques et de formats de données afin de centraliser l'évaluation. Par exemple, la majorité des ensembles de données présentés à la section 4 sont testés sur différentes métriques. On utilise alors l'outil de [**Padilla et al., 2021a**] qui présente les métriques mAP, $AP[0.50 : 0.95]$, $AP50$, $AP75$, AP_{medium}, AP_{large}, AR_{medium} et AR_{large}. Celles-ci sont décrites au chapitre 2. Il est à noter que les métriques de AP_{small} et AR_{small} existent, mais ne sont pas applicables ici, car les ensembles n'ont aucun tableau de la taille *small*, soit plus petit que 32^2 pixels. De plus, étant donné qu'une seule classe est présente pour la détection de tableaux, la métrique de mAP est en fait la métrique $PascalAP50$. En effet, on rappelle que la métrique de AP (section 2.5) peut être interpolée selon différents nombres de points. La compétition COCO [**COCO, 2019**] utilise $N = 101$ et la compétition Pascal Visual Object Classes [**Everingham et al., 2015**] utilisait $N = 11$ pour ensuite basculer avec la méthode d'interpolation de tous les points. Tandis, que les métriques $AP[0.50 : 0.95]$, $AP50$, $AP75$, AP_{medium}, AP_{large}, AR_{medium} et AR_{large} sont originaires de la compétition COCO et utilisent l'interpolation avec $N = 101$. On spécifie aussi des recommandations sur les métriques les plus appropriés pour notre tâche.

6.1. Outil de métriques

Les résultats produits à l'aide de l'outil de [**Padilla et al., 2021a**] sont présentés aux tableaux de 6.1. Chaque sous-tableau présente les résultats d'un modèle spécifique et chaque colonne représente les ensembles de tests qui sont chacun associés à une couleur. Les métriques sont listées à la première colonne afin que chaque ligne représente les résultats de cette métrique. Ainsi, pour chaque métrique et chaque ensemble test, on souligne le modèle le plus performant avec cette couleur. L'agglomération des colonnes de chaque sous-tableau contient alors 10 résultats soulignés par une couleur, pour les 10 métriques. La métrique nommée scores-F1 est celle utilisée au chapitre précédant, soit la métrique utilisée par les auteurs de TableBank [**Li et al., 2019b**].

Word

Métriques	Word	Latex	PubLayNet	Word-Latex	Word-PubLayNet	Latex-PubLayNet	Word-Latex-PubLayNet
Score-F1	89.74%	85.46%	85.97%	87.72%	88.10%	86.91%	88.21%
Pascal AP50	90.58%	95.07%	91.20%	92.40%	88.92%	92.93%	91.12%
Pascal AP85	86.61%	81.64%	77.75%	84.01%	80.57%	78.88%	80.96%
AP[0.50:0.95]	85.70%	80.38%	78.11%	82.22%	80.41%	78.55%	79.96%
AP50	90.23%	94.72%	90.96%	92.03%	88.42%	92.45%	90.50%
AP75	88.52%	91.97%	83.94%	89.83%	84.38%	87.93%	87.00%
APmedium	64.17%	65.31%	47.53%	62.39%	46.81%	64.36%	61.47%
APlarge	85.87%	81.45%	78.36%	82.94%	80.71%	79.25%	80.55%
ARmedium	90.00%	87.07%	81.67%	80.91%	85.45%	80.45%	80.96%
ARlarge	95.87%	87.54%	87.33%	91.88%	91.67%	87.44%	90.35%

Latex

Métriques	Word	Latex	PubLayNet	Word-Latex	Word-PubLayNet	Latex-PubLayNet	Word-Latex-PubLayNet
Score-F1	88.69%	95.90%	91.73%	88.46%	88.28%	93.11%	89.66%
Pascal AP50	87.18%	98.95%	97.46%	93.50%	91.58%	98.31%	94.45%
Pascal AP85	76.99%	94.87%	88.02%	86.10%	81.69%	91.57%	86.40%
AP[0.50:0.95]	75.63%	90.90%	84.79%	83.41%	79.57%	87.93%	83.57%
AP50	86.73%	98.53%	96.68%	92.77%	91.04%	98.07%	93.71%
AP75	81.88%	97.19%	93.92%	89.89%	87.04%	95.70%	90.84%
APmedium	34.55%	84.08%	72.48%	80.74%	50.37%	83.01%	79.82%
APlarge	75.86%	91.40%	85.03%	83.55%	79.76%	88.26%	83.78%
ARmedium	34.00%	87.71%	78.33%	84.66%	58.18%	87.08%	84.26%
ARlarge	88.80%	95.01%	89.61%	91.77%	89.20%	92.24%	91.05%

Word-Latex

Métriques	Word	Latex	PubLayNet	Word-Latex	Word-PubLayNet	Latex-PubLayNet	Word-Latex-PubLayNet
Score-F1	89.44%	88.46%	89.82%	89.52%	88.65%	91.22%	89.63%
Pascal AP50	84.96%	98.91%	95.69%	93.20%	89.99%	97.43%	93.69%
Pascal AP85	80.33%	93.53%	83.70%	88.04%	81.95%	88.79%	86.40%
AP[0.50:0.95]	77.03%	89.21%	81.60%	84.31%	79.05%	85.63%	83.13%
AP50	84.75%	98.10%	95.21%	92.76%	89.43%	97.05%	93.36%
AP75	82.68%	96.85%	90.57%	91.17%	86.23%	93.93%	90.47%
APmedium	53.40%	87.42%	80.18%	85.46%	68.35%	86.48%	84.74%
APlarge	77.14%	89.40%	81.75%	84.27%	79.23%	85.71%	83.13%
ARmedium	74.00%	92.41%	83.33%	91.36%	79.10%	91.80%	90.85%
ARlarge	93.06%	93.68%	87.52%	93.36%	90.33%	90.51%	91.39%

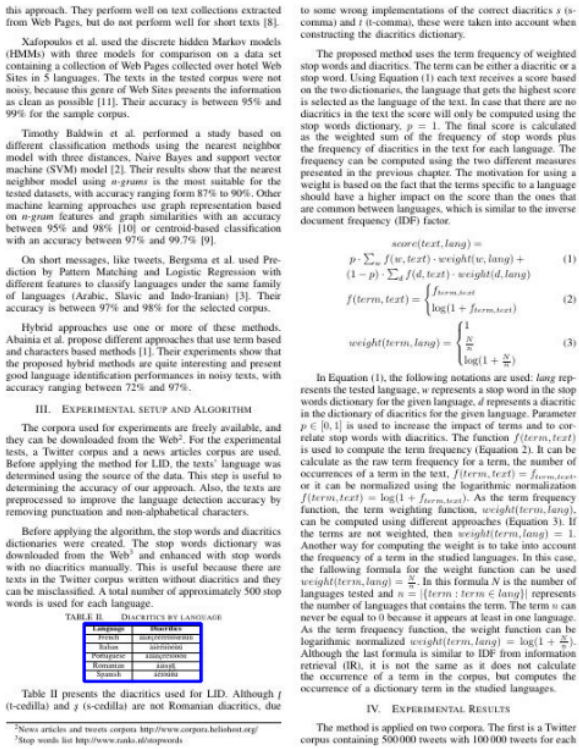
Word-PubLayNetx							
Métriques	Word	Latex	PubLayNet	Word-Latex	Word-PubLayNet	Latex-PubLayNet	Word-Latex-PubLayNet
Score-F1	89.32%	93.70%	95.95%	92.55%	92.30%	95.21%	92.56%
Pascal AP50	89.12%	98.25%	98.29%	94.08%	95.38%	98.21%	96.10%
Pascal AP85	84.61%	82.69%	93.66%	83.70%	90.59%	88.71%	87.97%
AP[0.50:0.95]	82.43%	83.66%	91.58%	82.98%	88.55%	87.82%	86.66%
AP50	88.75%	97.78%	97.84%	93.66%	94.97%	97.73%	95.66%
AP75	86.71%	94.60%	96.37%	91.03%	92.96%	95.32%	93.33%
APmedium	47.67%	72.81%	71.23%	71.28%	61.57%	72.53%	71.13%
APlarge	82.58%	84.39%	91.86%	83.50%	88.70%	88.29%	87.08%
ARmedium	86.00%	81.93%	78.33%	82.16%	81.82%	81.69%	81.91%
ARlarge	94.33%	89.74%	95.16%	92.13%	94.74%	92.52%	93.15%

Word-Latex-PubLayNet							
Métriques	Word	Latex	PubLayNet	Word-Latex	Word-PubLayNet	Latex-PubLayNet	Word-Latex-PubLayNet
Score-F1	88.03%	94.78%	93.82%	90.00%	90.64%	94.14%	91.40%
Pascal AP50	85.63%	98.43%	97.80%	93.78%	92.25%	98.09%	94.99%
Pascal AP85	79.85%	92.76%	90.97%	87.80%	85.91%	91.73%	88.71%
AP[0.50:0.95]	75.89%	87.58%	87.43%	83.13%	82.08%	87.34%	84.43%
AP50	85.52%	97.78%	97.41%	93.34%	91.93%	97.60%	94.58%
AP75	82.47%	96.72%	94.99%	91.34%	89.42%	96.11%	92.46%
APmedium	43.55%	82.58%	77.65%	81.19%	60.03%	82.09%	80.53%
APlarge	76.08%	88.09%	87.60%	83.28%	82.39%	87.83%	84.60%
ARmedium	76.00%	89.52%	81.67%	88.75%	79.09%	88.99%	88.30%
ARlarge	90.98%	92.69%	92.12%	91.80%	91.54%	92.40%	91.91%

CasecadeTabNet							
Métriques	Word	Latex	PubLayNet	Word-Latex	Word-PubLayNet	Latex-PubLayNet	Word-Latex-PubLayNet
Score-F1	72.04%	82.15%	91.51%	75.20%	81.62%	88.51%	81.72%
Pascal AP50	48.07%	74.13%	83.84%	59.65%	63.75%	78.53%	67.00%
Pascal AP85	41.09%	61.20%	74.88%	49.67%	55.54%	67.60%	57.32%
AP[0.50:0.95]	41.40%	62.23%	74.39%	50.53%	55.58%	67.77%	57.62%
AP50	47.56%	74.25%	83.10%	59.48%	63.41%	78.25%	66.95%
AP75	43.61%	71.06%	79.14%	55.23%	59.53%	74.85%	62.58%
APmedium	37.57%	49.16%	34.04%	39.26%	27.26%	49.32%	39.49%
APlarge	41.75%	63.90%	75.30%	51.17%	56.22%	69.09%	58.45%
ARmedium	38.00%	64.34%	73.33%	62.84%	57.27%	64.94%	63.51%
ARlarge	61.58%	76.45%	87.34%	68.71%	74.29%	82.05%	74.97%

Tableau 6.1. Les différentes métriques testées sur les ensembles tests définies par chaque colonne où chaque sous-tableau représente un modèle indiqué en gras au-dessus des colonnes.

Au tableau 6.1 on remarque que les métriques de AP_{medium} et AR_{medium} semblent généralement exceller sur un même modèle, soit celui entraîné sur Word-Latex, même lorsque les autres métriques pointent sur un modèle plus performant. On rappelle que les tableaux moyens sont entre 32^2 pixels et 96^2 pixels, un exemple de cette taille est présenté à la figure 6.1. Les tableaux moyens sont une minorité, on compte 94 tableaux moyens pour la combinaison de l'ensemble test Word, Latex et PubLayNet et 3705 tableaux de grande taille. Le fait d'avoir un modèle particulier qui se démarque pour cette métrique est étonnant puisque le modèle de Word-PubLayNet performe le mieux sur la plupart des ensembles de tests et métriques. Ce résultat prouve qu'il est important d'employer plus d'une métrique pour mesurer des informations spécifiques. Par exemple, si on chercherait un modèle performant bien sur des tableaux de petites tailles, on pourrait prioriser les métriques AP_{small}/AR_{small} et AP_{medium}/AR_{medium}.



this approach. They perform well on text collections extracted from Web Pages, but do not perform well for short texts [8].

Xafopoulos et al. used the discrete hidden Markov models (HMMs) with three models for comparison on a data set containing a collection of Web Pages collected over hotel Web Sites in 5 languages. The texts in the tested corpus were not noisy, because this genre of Web Sites presents the information as clean as possible [11]. Their accuracy is between 95% and 99% for the sample corpus.

Timothy Baldwin et al. performed a study based on different classification methods using the nearest neighbor model with three distances, Naive Bayes and support vector machine (SVM) model [2]. Their results show that the nearest neighbor model using *n-grams* is the most suitable for the tested datasets, with accuracy ranging form 87% to 90%. Other machine learning approaches use graph representation based on *n-gram* features and graph similarities with an accuracy between 95% and 98% [10] or centroid-based classification with an accuracy between 97% and 99.7% [9].

On short messages, like tweets, Bergsma et al. used Prediction by Pattern Matching and Logistic Regression with different features to classify languages under the same family of languages (Arabic, Slavic and Indo-Iranian) [3]. Their accuracy is between 97% and 98% for the selected corpus.

Hybrid approaches use one or more of these methods. Abainia et al. propose different approaches that use term based and characters based methods [1]. Their experiments show that the proposed hybrid methods are quite interesting and present good language identification performances in noisy texts, with accuracy ranging between 72% and 97%.

III. Experimental setup and Algorithm

The corpora used for experiments are freely available, and they can be downloaded from the Web[2]. For the experimental tests, a Twitter corpus and a news articles corpus are used. Before applying the method for LID, the texts' language was determined using the source of the data. This step is useful to determining the accuracy of our approach. Also, the texts are preprocessed to improve the language detection accuracy by removing punctuation and non-alphabetical characters.

Before applying the algorithm, the stop words and diacritics dictionaries were created. The stop words dictionary was downloaded from the Web[3] and enhanced with stop words with no diacritics manually. This is useful because there are texts in the Twitter corpus written without diacritics and they can be misclassified. A total number of approximately 500 stop words is used for each language.

TABLE II. Diacritics by language

Language	Diacritics
French	àâçéèêëîïôùûüÿœæ
Italian	àèéìíîòóùú
Portuguese	áâàãçéêíóôõúü
Romanian	ăâșțîĂÂȘȚÎ
Spanish	áéíóúüñ

Table II presents the diacritics used for LID. Although *ţ* (t-cedilla) and *ş* (s-cedilla) are not Romanian diacritics, due to some wrong implementations of the correct diacritics *s* (s-comma) and *t* (t-comma), these were taken into account when constructing the diacritics dictionary.

The proposed method uses the term frequency of weighted stop words and diacritics. The term can be either a diacritic or a stop word. Using Equation (1) each text receives a score based on the two dictionaries, the language that gets the highest score is selected as the language of the text. In case that there are no diacritics in the text the score will only be computed using the stop words dictionary, $p = 1$. The final score is calculated as the weighted sum of the frequency of stop words plus the frequency of diacritics in the text for each language. The frequency can be computed using the two different measures presented in the previous chapter. The motivation for using a weight is based on the fact that the terms specific to a language should have a higher impact on the score than the ones that are common between languages, which is similar to the inverse document frequency (IDF) factor.

$$score(text, lang) = p \cdot \sum_w f(w, text) \cdot weight(w, lang) + (1-p) \cdot \sum_d f(d, text) \cdot weight(d, lang) \quad (1)$$

$$f(term, text) = \begin{cases} f_{term,text} \\ \log(1 + f_{term,text}) \end{cases} \quad (2)$$

$$weight(term, lang) = \begin{cases} 1 \\ \frac{N}{n} \\ \log(1 + \frac{N}{n}) \end{cases} \quad (3)$$

In Equation (1), the following notations are used: *lang* represents the tested language, *w* represents a stop word in the stop words dictionary for the given language, *d* represents a diacritic in the dictionary of diacritics for the given language. Parameter $p \in [0, 1]$ is used to increase the impact of terms and to correlate stop words with diacritics. The function $f(term, text)$ is used to compute the term frequency (Equation 2). It can be calculate as the raw term frequency for a term, the number of occurrences of a term in the text, $f(term, text) = f_{term,text}$, or it can be normalized using the logarithmic normalization $f(term, text) = \log(1 + f_{term,text})$. As the term frequency function, the term weighting function, $weight(term, lang)$, can be computed using different approaches (Equation 3). If the terms are not weighted, then $weight(term, lang) = 1$. Another way for computing the weight is to take into account the frequency of a term in the studied languages. In this case, the fallowing formula for the weight function can be used $weight(term, lang) = \frac{N}{n}$. In this formula N is the number of languages tested and $n = |\{term : term \in lang\}|$ represents the number of languages that contains the term. The term n can never be equal to 0 because it appears at least in one language. As the term frequency function, the weight function can be logarithmic normalized $weight(term, lang) = \log(1 + \frac{N}{n})$. Although the last formula is similar to IDF from information retrieval (IR), it is not the same as it does not calculate the occurrence of a term in the corpus, but computes the occurrence of a dictionary term in the studied languages.

IV. Experimental Results

The method is applied on two corpora. The first is a Twitter corpus containing 500 000 tweets with 100 000 tweets for each

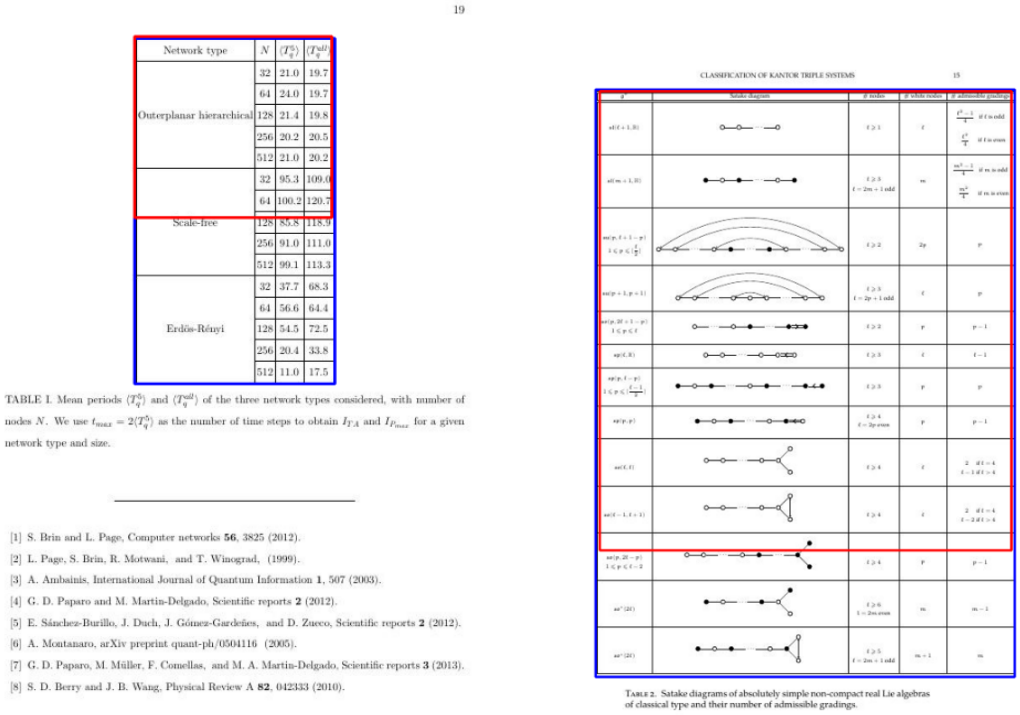
[2]News articles and tweets corpora http://www.corpora.heliohost.org/
[3]Stop words list http://www.ranks.nl/stopwords

Fig. 6.1. Exemple d'un tableau de taille moyenne.

On rappelle que la métrique de AP est le *average precision* qui est associé à un seuil IoU pour associer une détection à un label et déterminer si on considère la région comme détectée correctement. Afin de bien concevoir les différentes métriques de AP, on présente à la figure 6.2 une prédiction avec un IoU de 0.51 et 0.76. On juge que la métrique de AP50 n'est pas assez précise pour être utilisée seule, comme on le voit à la figure 6.2, un IoU de 0.5 est un seuil plutôt bas et n'encadre pas justement le tableau. La détection de tableaux ne peut pas être jugée comme n'importe quel détection d'objets. Plusieurs types de détections d'objets ne nécessitent pas d'avoir une grande

aire de chevauchement, c'est la raison pour laquelle le standard pour les métriques de détection est un mAP avec un IoU de 0.5. Par contre, pour la détection de tableaux dans les documents, le but est généralement d'appliquer un outil d'OCR par la suite afin d'extraire les informations des tableaux. Ainsi, il est ainsi important que la détection soit de bonne qualité. Par contre, si la tâche serait simplement de grossièrement localiser des tableaux, alors utiliser une métrique utilisant un IoU de 0.5 pourrait être pertinente. C'est la raison pour laquelle on décide d'ajouter la métrique de $PascalAP85$, qui n'est pas standard, afin de façonner la métrique pour nos besoins. Sur les 7 ensembles tests, il y a 3 ensembles pour lesquels $PascalAP50$ et $PascalAP85$ n'engendrent pas les mêmes gagnants. À noter, qu'évidemment, utiliser un IoU plus tolérant, produit un score plus haut. Bref, on voit que les métriques $PascalAP50$ et $PascalAP85$ ne seront pas toujours d'accord. On juge alors qu'il est plus important de prioriser la métrique $PascalAP85$ avec un IoU plus sévère pour la détection de tableaux.

Cette même logique s'applique aux métriques COCO $AP50$ et $AP75$. Pour la détection de tableaux, il serait plus pertinent d'utiliser un IoU au moins plus grand que 0.8. La métrique de $AP[0.50 : 0.95]$, ou une métrique semblable, est aussi utilisée par quelques scientifiques pour la détection de tableaux, comme il sera discuté à la section 6.2 suivante. Par contre, on peut encore une fois se questionner sur la pertinence de prendre en compte des IoU bas dans la moyenne.



19

Network type	N	$\langle T_q^{\diamond} \rangle$	$\langle T_q^{\text{add}} \rangle$
Outerplanar hierarchical	32	21.0	19.7
	64	24.0	19.7
	128	21.4	19.8
	256	20.2	20.5
	512	21.0	20.2
Scale-free	32	95.3	109.0
	64	100.2	120.7
	128	85.8	118.9
	256	91.0	111.0
	512	99.1	113.3
Erdős-Rényi	32	37.7	68.3
	64	56.6	64.4
	128	54.5	72.5
	256	20.4	33.8
	512	11.0	17.5

TABLE I. Mean periods $\langle T_q^{\diamond} \rangle$ and $\langle T_q^{\text{add}} \rangle$ of the three network types considered, with number of nodes N. We use $t_{max} = 2\langle T_q^{\diamond} \rangle$ as the number of time steps to obtain I_{TA} and $I_{P_{max}}$ for a given network type and size.

[1] S. Brin and L. Page, Computer networks **56**, 3825 (2012).
[2] L. Page, S. Brin, R. Motwani, and T. Winograd, (1999).
[3] A. Ambainis, International Journal of Quantum Information **1**, 507 (2003).
[4] G. D. Paparo and M. Martin-Delgado, Scientific reports **2** (2012).
[5] E. Sánchez-Burillo, J. Duch, J. Gómez-Gardeñes, and D. Zueco, Scientific reports **2** (2012).
[6] A. Montanaro, arXiv preprint quant-ph/0504116 (2005).
[7] G. D. Paparo, M. Müller, F. Comellas, and M. A. Martin-Delgado, Scientific reports **3** (2013).
[8] S. D. Berry and J. B. Wang, Physical Review A **82**, 042333 (2010).

CLASSIFICATION OF KANTOR TRIPLE SYSTEMS 15

TABLE 2. Satake diagrams of absolutely simple non-compact real Lie algebras of classical type and their number of admissible gradings.

Fig. 6.2. Exemple de détections avec un IoU de 0.51 à gauche et 0.76 à droite où les détections sont en rouge et les labels sont en bleu.

Afin de comparer la conformité des métriques, on liste les modèles gagnants, soit le modèle ayant les meilleures performances, selon les différentes métriques au tableau 6.2.

Ens. Test	**Modèle gagnant**	
	TableBank Score-F1	Autres métriques
Word	Word	Word (9/9)
Latex	Latex	Latex (7/9)
PubLayNet	Word-PubLayNet	Word-PubLayNet (7/9)
Word-Latex	Word-PubLayNet	Word-Latex (6/9)
Word-PubLayNet	Word-PubLayNet	Word-PubLayNet (8/9)
Latex-PubLayNet	Word-PubLayNet	Latex (3/9)
Word-Latex-PubLayNet	Word-PubLayNet	Word-PubLayNet (6/9)

Tableau 6.2. Modèles les plus performants pour chaque ensemble test selon les différentes métriques. Pour la colonne "Autres métriques" on précise la proportion des métriques avec le même avis.

Au tableau 6.2, on remarque rapidement que les métriques sont rarement en accord et génèrent différents modèles gagnants. De plus, même lorsqu'une majorité de métriques pointe sur le même modèle, il arrive rarement qu'elles soient toutes en accord. Ce résultat démontre que les métriques utilisées sont décisives pour choisir le modèle le plus performant.

De surcroît, on note que la métrique de Score-F1 utilisée par TableBank n'est pas toujours en accord avec les autres métriques. En effet, la métrique de TableBank prend en compte chaque différence entre le label et la prédiction, tandis qu'avec une métrique de AP, selon un certain seuil, on associe la prédiction à 100%. Ce choix semble plus logique, particulièrement en sachant que l'ensemble de TableBank contient des erreurs de label et que d'autres ensembles créés automatiquement risquent d'en contenir aussi. Ainsi, ignorer une petite différence peut alors permettre d'excuser certaines erreurs de label.

6.2. Métriques d'articles associés

Comme mentionné précédemment, les différents articles importants dans le domaine de détection de tableaux et détection d'objets dans des documents utilisent majoritairement tous des métriques différentes. Les auteurs de TableBank [**Li et al., 2019b**]

utilisent leurs métriques de scores-F1 mesurant l'air de chevauchement. Les auteurs de PubLayNet [**Zhong et al., 2019**] utilisent la métrique de $mAP[0.50 : 0.95]$, car l'ensemble contient plusieurs classes. Tandis que DeepFigure [**Siegel et al., 2018**], un ensemble de données tel PubLayNet pour la détection de graphiques, diagrammes et tableaux, utilise un simple calcule de score-F1 avec une condition de IoU de 0.8 pour considérer la prédiction comme valide. Les auteurs de CascadeTabNet [**Prasad et al., 2020b**] utilisent le *weighted-average F1* sur des IoU de 0.6, 0.7, 0.8, 0.9.

Il existe aussi quelques autres modèles pour la détection de tableaux, que nous n'avons pas étudiés. Le modèle HybridTabNet [**Nazir et al., 2021**] a été créé pour la détection de tableaux et utilise aussi la métrique *weighted-average F1*, mais cette fois avec des IoU de 0.5, 0.6, 0.7, 0.8, 0.9. Le modèle TableNet [**Paliwal et al., 2019**] permet la détection de tableaux et reconnaissance de la structure de tableaux, par contre, les auteurs sont plutôt abstraits par rapport aux métriques. Ils précisent seulement qu'ils utilisent la précision, rappel et score-F1. Un autre modèle, DeepDeSRT [**Schreiber et al., 2017**], pour la reconnaissance de la structure d'un tableau, utilise la précision, rappel et score-F1 qui sont utilisés dans la compétition ICDAR2013, mais ils ne précisent pas non plus quelle condition ils utilisent pour les calculer. La seule explication mentionnée est de mesurer la précision, rappel et score-F1 pour chaque document séparément pour ensuite appliquer une moyenne sur l'ensemble. Les auteurs utilisent aussi les métriques de AP et AR, mais ne précisent pas avec quel IoU. Connaissant le grand nombre de métriques utilisé pour cette problématique, il est important que les scientifiques détaillent leurs métriques.

Par exemple, [**Prasad et al., 2020b**] utilise la métrique de TableBank afin de comparer leurs résultats avec ceux de TableBank en reproduisant ceux-ci. Par contre, les résultats produits sont autour de 5% plus bas à ceux produits par les auteurs de TableBank. Les auteurs de CascadeTabNet affirment alors que leur modèle est plus performant, mais seulement parce qu'ils entraînent les modèles de TableBank avec un ensemble d'entraînement beaucoup plus petit.

Bref, comme on peut le voir, la tâche de détection de tableaux n'est pas préservée de la problématique de détection d'objets, soit un trop grand nombre de formats de données et de métriques. Les ensembles de données et les modèles utilisent tous des métriques différentes. Cette variété rend la comparaison de modèles et de résultats très difficile. Par exemple, tous les modèles définis au paragraphe précédent affirment être plus performants que leurs prédécesseurs, mais démontrent seulement cet avantage sur une métrique précise. Par contre, comme il a été montré à la section 6.1, les métriques vont rarement s'accorder sur un même modèle gagnant. Ainsi, comparer les modèles sur

des métriques pertinentes et semblables est essentiel.

6.3. Conclusion

Le choix d'une métrique particulière peut amener à un classement de modèles différent. Il n'en reste pas moins que les différences observées selon les métriques ne sont pas drastiques. Cette particularité est majoritairement due au fait que les modèles sont tous entraînés par le même modèle, excepté CascadeTabNet. Ainsi, dans notre cas, sélectionner un modèle selon une métrique particulière n'engendre pas de différences notables dans des cas réels, mais représente plutôt un enjeu significatif dans un contexte académique. Ce qu'on sous-entend par «cas réels» est, par exemple, qu'un logiciel pour détecter les tableaux dans des documents ne provoquera pas de différences significatives pour un usager régulier s'il utilise un modèle avec une précision 2% plus haute. Par contre, il est important de connaître les besoins d'une tâche afin de choisir les bonnes métriques, même pour des cas réels, comme par exemple, la quantité de tableaux, la quantité de figures accompagnant les tableaux ou le nombre de tableaux par documents.

D'un autre côté, on voit que même pour un seul modèle avec plusieurs ensembles d'entraînement, les métriques génèrent différents gagnants, ce qui montre l'importance d'observer plus d'une métrique. On souligne aussi l'importance pour les scientifiques de s'accorder sur l'utilisation de métriques afin de faciliter et clarifier la comparaison de modèles. Dans ce but, on propose des métriques pertinentes pour la détection de tableaux, soit les métriques AP_{medium}/AP_{medium}, $PascalAP85$ ou $COCOAP85$ et la métrique de TableBank. La métrique AP_{medium}/AP_{medium} permet de mesurer la qualité pour différentes tailles de tableaux, $PascalAP85$ ou $COCOAP85$ utilise un seuil d'IoU plus haut que la norme, tandis que la métrique de TableBank prend en compte chaque différence entre le label et la prédiction, ce qui peut être un avantage pour notre tâche. En effet, comme mentionné, les détections de tableaux nécessitent d'être mesurées avec des seuils plutôt stricts. Il faut un bon encadrement pour être utile en pratique, particulièrement en sachant que le but est généralement d'appliquer un outil d'OCR afin d'extraire les informations des tableaux.

Chapitre 7

Conclusion

Dans ce mémoire, plusieurs aspects de la détection de tableaux dans des documents ont été explorés. Tout d'abord, on rassemble toutes les informations essentielles sur la tâche: les métriques, les modèles et les ensembles de données associés à la détection de tableaux, ce qui représente un rassemblement intéressant des éléments fondamentaux. Ensuite, une étude des corpus d'entraînement a été effectuée. On débute par reproduire les résultats des auteurs de TableBank [**Li et al., 2019b**]. Étant donné que les détails de la métrique utilisée par les auteurs n'est pas disponible, on implémente celle-ci selon notre compréhension et on remarque une différence dans nos résultats et ceux des auteurs. Cette divergence renforce notre insistance sur la nécessité de transparence de la part des auteurs afin de permettre des analyses et des comparaisons aisées.

De plus, pour étudier l'impact des ensembles d'entraînement, on emploie un modèle moins complexe Faster R-CNN que ceux des auteurs de TableBank qui utilisait ResNeXt-101 et ResNeXt-152 afin de pouvoir jouer avec les ensembles de données selon les conditions de notre ordinateur. On rappelle que TableBank utilise deux ensembles distincts de Latex et Word, et créer différentes combinaisons afin de mesurer l'influence de ceux-ci. On a ajouté à ces combinaisons un sous-ensemble de PubLayNet. Cet ajout semblait être bénéfique, ce qui est particulièrement pertinent en sachant que PubLayNet contient moins d'erreurs d'annotations que TableBank.

On compare aussi notre modèle entrainé sur les différentes combinaisons de TableBank et PubLayNet avec le modèle CascadeTabNet. Notre modèle générait systématiquement des meilleurs résultats que ceux de CascadeTabNet, contrairement à ce que les auteurs avançaient. Afin de comparer les modèles sous une nouvelle information, on sépare les ensembles de données selon leurs nombres de tableaux, et étonnamment CascadeTabNet est un modèle stable qui ne semble pas être influencé par le nombre de tableaux sur un document. C'est un résultat particulièrement intéressant en sachant que notre modèle TableBank voit son score-F1 diminuer fortement si le nombre de tableaux dans l'image est trop élevé. On évalue aussi des méthodes d'augmentation de données qui sont utilisées dans l'entraînement du modèle de CascadeTabNet et on mesure une amélioration aux

performances dans certain cas.

Ces études ont aussi apporté la conclusion que si l'on désire un modèle qui peut performer sur des documents diversifiés, avoir un ensemble d'entraînement diversifié est essentiel. Par contre, il a été intéressant de noter que si l'on travaille dans un cas où l'on connait l'ensemble à évaluer, avoir un ensemble d'entraînement ressemblant à l'ensemble test est à prioriser au lieu de prioriser la diversité.

Par la suite, on étudie l'impact des métriques. On compare les résultats des métriques Pascal AP50, Pascal AP85, AP[0.5:0.95], AP50, AP75, APmedium, APlarge, ARmedium et ARlarge avec différents modèles. On remarque que les métriques ne vont généralement pas produire les mêmes modèles gagnants. Ce résultat marque un problème plutôt important sachant que tous les modèles de détections de tableaux que nous avons listés à la section 6.2 utilisent différentes métriques. Ces auteurs se comparent et posent l'hypothèse qu'ils performent mieux qu'un modèle précédent. On pointe alors l'importance pour les scientifiques de s'accorder sur l'utilisation de métriques afin de faciliter et clarifier la comparaison de modèles. Dans ce but, on propose des métriques pertinentes pour la détection de tableaux, soit les métriques AP_{medium}/AP_{medium}, Pascal AP85 ou COCO AP85 et la métrique de TableBank.

Travaux futurs

Il serait bénéfique d'être en mesure de produire les résultats du modèle complexe ResNeXt-152 utilisé par les auteurs de TableBank avec les différentes combinaisons d'ensembles d'entraînement. Ainsi, on pourrait poser des hypothèses solides sur l'utilisation de grands ensembles d'entraînement combinant TableBank et PubLayNet, et l'application d'augmentation de données sur ceux-ci. En effet, il était parfois difficile de se prononcer précisément sur les avantages d'élargir les ensembles d'entraînement compte tenu de l'utilisation d'un modèle moins complexe qui ne pouvaient pas exploiter le plein potentiel d'un ensemble volumineux.

De plus, étant donné les erreurs contenues dans l'ensemble de données Word de TableBank, il serait intéressant d'améliorer la méthode d'annotation pour cet ensemble. De cette façon, on pourrait mesurer l'impact d'erreurs d'annotation de modèles d'apprentissage profonds pour notre tâche. Les performances pourraient se voir améliorées.

Aussi, il serait intéressant de produire les résultats de tous les modèles existants pour la détection de tableaux en utilisant les métriques que l'on recommande afin d'obtenir un regroupement structuré et une comparaison équitable. On serait ainsi en mesure de clairement juger la façon dont l'utilisation de certaines métriques peuvent biaiser les performances, et, sous un nouvel angle, définir les meilleurs modèles pour la tâche.

Résumé du code

Modèle

Pour remettre en contexte, nous avons étudié les ensembles de données TableBank et PubLayNet. Comme mentionné, les auteurs de TableBank utilisent Detectron (section 3.2), une librairie qui offre des modèles de différentes tailles préentraînés sur ImageNet. Les auteurs de TableBank ont affiné ces modèles préentraînés avec leurs ensembles de données TableBank et ont rendu disponibles leurs modèles entraînés sur leur GitHub [**Li et al., 2019a**] que j'ai utilisés.

Étant donné que nous voulions tenter d'améliorer les résultats des auteurs de TableBank, il était nécessaire de procéder aux mêmes étapes que les auteurs, soit utiliser la librairie Detectron afin d'affiner des modèles préentraînés de Detectron. Nous avons alors conçu des méthodes pour utiliser la librairie Detectron et entraîner les modèles préentraînés, ce qui représente environs 30 lignes de code

Il est aussi important de mentionner que Detectron utilise un format spécifique de données, soit une façon particulière de placer les données et définir les attributs. Ceci nous a amené à produire des méthodes pour transformer les ensembles de données TableBank et PubLayNet dans le format requis pour entraîner les modèles de Detectron, ce qui représente environs 50 lignes de codes.

De surcroît, nous avons produit un code pour utiliser le modèle préentraîné de CascadeTabNet avec les différents ensembles de données de TableBank et PubLayNet. Les modèles préentraînés sont disponibles sur le GitHub des auteurs [**Prasad et al., 2020a**]. Ils ne nécessitaient pas d'entraînement.

Les auteurs de CascadeTabNet utilisent des méthodes d'augmentation de données à l'aide de deux types de transformations. Les méthodes de transformations sont définies dans le GitHub des auteurs [**Prasad et al., 2020a**]. Nous avons alors créé des fonctions pour appliquer ces transformations sur nos ensembles de données afin d'entraîner les modèles préentraînés de Detectron et mesurer les effets sur les performances, ce qui représente environs 50 lignes de code.

Évaluation

La métrique utilisée par les auteurs de TableBank n'était pas disponible, ainsi, celle-ci a été développée selon notre compréhension des explications des auteurs. Le code de la métrique est disponible sur mon GitHub [**Yockell, 2022**] et représente environs 125 lignes de code.

Une implémentation de méthodes pour évaluer la qualité à l'aide de la métrique développée a été faite. Nous avons aussi créé des fonctions pour générer les différentes combinaisons d'ensembles de données possibles avec les deux ensembles TableBank et PubLayNet, où on rappelle que TableBank contient deux sous-ensembles, ce qui représente environs 20 lignes de code.

D'autre part, nous avons découvert qu'il est pertinent de mesurer la qualité d'un modèle en mesurant la performance sur des images contenant différents nombres de tableaux. En effet, un modèle tente à avoir plus de difficulté à détecter les tableaux dans une image lorsqu'ils sont nombreux. Ce phénomène nous a mené à développer une méthode pour séparer un ensemble de données selon le nombre de tableaux dans l'image et ensuite mesurer les performances de chaque sous-ensemble, ce qui représente environs 40 lignes de code.

Dans le chapitre 6, on étudie différentes métriques. Les nouvelles métriques étudiées dans ce chapitre sont définies dans un outil disponible sur le Github des auteurs [**Padilla et al., 2021b**] qui présente les métriques mAP, $AP[0.50 : 0.95]$, $AP50$, $AP75$, AP_{medium}, AP_{large}, AR_{medium} et AR_{large}. Encore une fois, pour utiliser l'outil, il était nécessaire de définir des méthodes pour transformer les données dans un format compatible avec l'outil, ce qui représente environs 40 lignes de code.

La contribution de code repose aussi dans la centralisation de *scripts* et *Jupyter notebooks* qui permettent d'utiliser facilement les différentes méthodes, modèles et métriques reliés à la détection de tableaux. Nous avons construit une ressource accessible par mon GitHub [**Yockell, 2022**] visant à simplifier la navigation de ce domaine pour des chercheurs ou étudiants. Ce mémoire sert de référence essentielle pour quiconque s'engage dans des travaux de recherche dans ce domaine spécifique.

Références bibliographiques

[Borchmann et al., 2021] Borchmann, Ł., Pietruszka, M., Stanislawek, T., Jurkiewicz, D., Turski, M., Szyndler, K., and Graliński, F. (2021). DUE: End-to-end document understanding benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

[Cai and Vasconcelos, 2018] Cai, Z. and Vasconcelos, N. (2018). Cascade r-cnn: Delving into high quality object detection. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6154–6162.

[Chen et al., 2019] Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C. C., and Lin, D. (2019). MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*.

[COCO, 2019] COCO (2019). Coco detection challenge (bounding box). `https://competitions.codalab.org/competitions/20794`. Accessed: 2023-03-07.

[Costa e Silva, 2011] Costa e Silva, A. (2011). Metrics for evaluating performance in document analysis: Application to tables. *Document Analysis and Recognition*, 14:101–109.

[Everingham et al., 2015] Everingham, M., Eslami, S. M. A., Gool, L. V., Williams, C. K. I., Winn, J., and Zisserman, A. (2015). The pascal visual object classes challenge: A retrospective.

[Gao et al., 2019] Gao, L., Huang, Y., Déjean, H., Meunier, J.-L., Yan, Q., Fang, Y., Kleber, F., and Lang, E. (2019). Icdar 2019 competition on table detection and recognition (ctdar). In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1510–1515.

[Gilani et al., 2017] Gilani, A., Qasim, S. R., Mali, I., and Shafait, F. (2017). Table detection using deep learning.

[Girshick, 2015] Girshick, R. (2015). Fast r-cnn.

[Girshick et al., 2013] Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2013). Rich feature hierarchies for accurate object detection and semantic segmentation.

[Göbel et al., 2013] Göbel, M., Hassan, T., Oro, E., and Orsi, G. (2013). Icdar 2013 table competition. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1449–1453.

[He et al., 2017] He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn.

[He et al., 2015] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition.

[Institute of Computer Science and Technology of Peking University, 2011] Institute of Computer Science and Technology of Peking University (2011). Marmot dataset. `https://www.icst.pku.edu.cn/cpdp/sjzy/`.

[Khandelwal, 2021] Khandelwal, R. (2021). Coco and pascal voc data format for object detection.

[Lab et al., 2011] Lab, S. V., University, S., and University, P. (2011). Imagenet. `https://www.image-net.org/`.

[Li et al., 2019a] Li, M., Cui, L., Huang, S., Wei, F., Zhou, M., and Li, Z. (2019a). Tablebank. `https://github.com/doc-analysis/TableBank`.

[Li et al., 2019b] Li, M., Cui, L., Huang, S., Wei, F., Zhou, M., and Li, Z. (2019b). Tablebank: A benchmark dataset for table detection and recognition.

[Mandal, 2021] Mandal, M. (2021). Introduction to convolutional neural networks (cnn).

[Nazir et al., 2021] Nazir, D., Hashmi, K. A., Pagani, A., Liwicki, M., Stricker, D., and Afzal, M. Z. (2021). Hybridtabnet: Towards better table detection in scanned document images. *Applied Sciences*, 11(18).

[Padilla et al., 2021a] Padilla, R., Passos, W. L., Dias, T. L. B., Netto, S. L., and da Silva, E. A. B. (2021a). A comparative analysis of object detection metrics with a companion open-source toolkit. *Electronics*, 10(3).

[Padilla et al., 2021b] Padilla, R., Passos, W. L., Dias, T. L. B., Netto, S. L., and da Silva, E. A. B. (2021b). Open-source visual interface for object detection metrics. `https://github.com/rafaelpadilla/review_object_detection_metrics`.

[Paliwal et al., 2019] Paliwal, S. S., D, V., Rahul, R., Sharma, M., and Vig, L. (2019). Tablenet: Deep learning model for end-to-end table detection and tabular data extraction from scanned document images. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 128–133.

[Panchal, 2019] Panchal, S. (2019). Table detection dataset. `https://github.com/sgrpanchal31/table-detection-dataset`.

[Prasad et al., 2020a] Prasad, D., Gadpal, A., Kapadni, K., Visave, M., and Sultanpure, K. (2020a). Cascadetabnet. `https://github.com/DevashishPrasad/CascadeTabNet`.

[Prasad et al., 2020b] Prasad, D., Gadpal, A., Kapadni, K., Visave, M., and Sultanpure, K. (2020b). Cascadetabnet: An approach for end to end table detection and structure recognition from image-based documents.

[Rančić et al., 2023] Rančić, K., Blagojević, B., Bezdan, A., Ivosevic, B., Tubić, B., Vranešević, M., Pejak, B., Crnojevic, V., and Marko, O. (2023). Animal detection and counting from uav images using convolutional neural networks. *Drones*, 7:179.

[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.

[Schreiber et al., 2017] Schreiber, S., Agne, S., Wolf, I., Dengel, A., and Ahmed, S. (2017). Deepdesrt: Deep learning for detection and structure recognition of tables in document images. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 1162–1167.

[Siegel et al., 2018] Siegel, N., Lourie, N., Power, R., and Ammar, W. (2018). Extracting scientific figures with distantly supervised neural networks.

[Simonyan and Zisserman, 2014] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition.

[U.S. National Institutes of Health's National Library of Medicine, 2019] U.S. National Institutes of Health's National Library of Medicine (2019). Pubmed national library of medicine. `https://pubmed.ncbi.nlm.nih.gov/`.

[Wang et al., 2019] Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., Liu, W., and Xiao, B. (2019). Deep high-resolution representation learning for visual recognition.

[Wu et al., 2019] Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., and Girshick, R. (2019). Detectron2. `https://github.com/facebookresearch/detectron2`.

[Xie et al., 2016] Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. (2016). Aggregated residual transformations for deep neural networks.

[Yockell, 2022] Yockell, E. (2022). table_detection. `https://github.com/eugeniey/table_detection`.

[Zhong et al., 2019] Zhong, X., Tang, J., and Yepes, A. J. (2019). Publaynet: largest dataset ever for document layout analysis.

Annexe A

Informations additionnelles

A.1. TableBank - Reproduction des résultats

À la section 5.1.2, on présente les résultats du modèle Faster R-CNN entraînés avec les ensembles de données TableBank et PubLayNet, ainsi que le modèle CascadeTabNet. Ces modèles sont ensuite testés sur différents ensembles test. Au tableau 5.7, on présente seulement les scores-F1, tandis qu'au tableau A.1 on présente les résultats avec la précision, rappel et score-F1.

À la section 5.5, on présente l'impact d'utiliser des méthodes d'augmentation de données sur les ensembles d'entraînement, comme [**Prasad et al., 2020b**] ont utilisé pour entraîner le modèle de CascadeTabNet. Au tableau 5.9, on présente seulement les scores-F1, tandis qu'au tableau A.2 on présente les résultats avec la précision, rappel et score-F1.

Modèles	Word			Latex			PubLayNet		
	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1
Word	83.45%	97.05%	**89.74%**	77.96%	94.56%	85.46%	91.53%	81.04%	85.97%
Latex	90.11%	87.31%	88.69%	96.23%	95.56%	**95.90%**	89.60%	93.96%	91.73%
PubLayNet	84.50%	80.61%	82.51%	97.70%	88.91%	93.10%	97.48%	96.06%	**96.77%**
Word-Latex	85.26%	94.04%	89.44%	82.86%	94.87%	88.46%	91.21%	88.47%	89.82%
Word-PubLayNet	82.94%	96.75%	89.32%	95.94%	91.57%	93.70%	95.80%	96.10%	95.95%
Latex-PubLayNet	81.84%	87.55%	84.60%	97.29%	93.60%	95.41%	97.11%	94.24%	95.65%
Word-Latex-PubLayNet	80.82%	96.65%	88.03%	94.03%	95.56%	94.78%	91.97%	95.75%	93.82%
CascadeTabNet	80.05%	65.49%	72.04%	89.32%	76.04%	82.15%	91.78%	91.24%	91.51%

Modèles	Word-Latex			Word-PubLayNet			Latex-PubLayNet			Word-Latex-PubLayNet		
	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1
Word	80.97%	95.69%	87.72%	86.70%	89.54%	88.10%	91.02%	83.15%	86.91%	87.32%	89.12%	88.21%
Latex	82.86%	94.87%	88.46%	83.00%	94.28%	88.28%	91.75%	94.50%	93.11%	85.28%	94.53%	89.66%
PubLayNet	88.45%	83.18%	85.73%	90.70%	87.85%	89.25%	97.55%	93.65%	95.56%	91.98%	88.06%	89.97%
Word-Latex	84.56%	95.11%	89.52%	85.27%	92.31%	88.65%	92.19%	90.26%	91.22%	86.85%	92.59%	89.63%
Word-PubLayNet	89.78%	95.51%	**92.55%**	88.50%	96.45%	**92.30%**	95.85%	94.58%	95.21%	89.78%	95.51%	**92.56%**
Latex-PubLayNet	86.28%	89.42%	87.82%	88.63%	90.68%	89.65%	97.17%	94.02%	**95.57%**	90.22%	91.24%	90.73%
Word-Latex-PubLayNet	84.46%	96.31%	90.00%	85.67%	96.23%	90.64%	92.65%	95.69%	94.14%	87.15%	96.10%	91.40%
CascadeTabNet	83.00%	68.75%	75.20%	86.12%	77.56%	81.62%	91.03%	86.13%	88.51%	86.71%	77.27%	81.72%

Tableau A.1. Résultats du modèle Faster R-CNN entraîné sur différents ensembles et du modèle CascadeTabNet en utilisant la métrique de TableBank développer par nos soins. On utilise les échantillons de 1000 images prédéfinies avec un taux de confiance de 90%. On présente en gras le meilleur score-F1 obtenu par le meilleur modèle pour cet ensemble de test.

Modèles	Word			Latex			PubLayNet			Word-Latex		
	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1	Précision	Rappel	Score-F1
Word	83.45%	97.05%	89.74%	77.96%	94.56%	85.46%	91.53%	81.04%	85.97%	80.97%	95.69%	87.72%
Word + B	84.14%	94.19%	88.88%	66.78%	71.41%	69.02%	40.28%	38.67%	39.46%	78.94%	87.15%	82.84%
Word + E	84.65%	93.84%	89.01%	93.26%	76.59%	84.11%	90.19%	61.01%	72.78%	86.80%	88.51%	87.65%
Word + B + E	85.37%	95.69%	**90.24%**	91.15%	69.36%	78.78%	88.49%	44.48%	59.20%	86.72%	87.55%	87.13%
Latex	90.11%	87.31%	88.69%	96.23%	95.56%	**95.90%**	89.60%	93.96%	**91.73%**	82.86%	94.87%	88.46%
Latex + B	82.39%	84.10%	83.24%	95.63%	87.60%	91.44%	94.78%	71.43%	81.47%	86.18%	85.19%	85.68%
Latex + E	79.22%	94.86%	86.34%	94.03%	93.82%	93.93%	87.29%	90.16%	88.70%	83.25%	94.54%	85.53%
Latex + B + E	77.87%	95.71%	85.87%	94.67%	95.15%	94.91%	88.75%	87.17%	87.95%	82.37%	95.54%	**88.47%**

Tableau A.2. Résultats des modèles entraînés avec une augmentation de données en utilisant la métrique de TableBank développée par nos soins. On utilise les échantillons de 1000 images prédéfinis avec un taux de confiance de 90%. Les modèles sont définis avec un B pour l'ajout des images avec bavure et E pour l'ajout des images avec élargissement.