SDL-Net pour la Localisation de Documents Structurés : un U-Net économe en ressources.

Anastasiia Kabeshova*, Guillaume Betmont*, Julien Lerouge*,
Evgeny Stepankevich*, Alexis Bergès*

*QuickSign, Paris, France
{anastasiia.kabeshova, guillaume.betmont,
julien.lerouge, evgeny.stepankevich, alexis.berges}@quicksign.com,
https://www.quicksign.com/

Résumé. L'analyse et la reconnaissance de documents structurés sont essentielles pour les processus modernes d'onboarding en ligne et la localisation de documents est une étape cruciale pour obtenir une extraction fiable des informations clés. Bien que l'apprentissage profond soit devenu la technique standard utilisée pour résoudre les problèmes d'analyse de documents, les applications réelles dans l'industrie sont encore confrontées à la disponibilité limitée de données étiquetées et de puissance de calcul informatique lors de l'entraînement de modèles profonds. Pour relever ces défis, nous proposons SDL-Net : une nouvelle architecture encodeur-décodeur, basée sur U-Net, pour la localisation de documents structurés. Notre approche permet de pré-entraîner l'encodeur de SDL-Net sur un ensemble de données générique contenant des échantillons de diverses classes de documents. Elle permet aussi un ajustement fin rapide, et frugal en termes de données étiquetées nécessaires, de décodeurs gérant la localisation de nouvelles classes de documents. Nous menons des expériences approfondies sur un ensemble de données propriétaire d'images de documents structurés pour démontrer l'efficacité et les capacités de généralisation de l'approche proposée.

1 Introduction

La vérification d'identité en ligne à partir de documents officiels est un service qui demande une automatisation rapide et précise. La capture de ces documents par smartphone peut altérer la qualité et la perspective de l'image (Burie et al., 2015). Localiser le document est une des solutions permettant de rectifier la perspective, comme le montre la Figure 1d.

Nous proposons ici une architecture encodeur-décodeur, de type U-net (Ronneberger et al., 2015), pour localiser des documents structurés, en considérant un problème industriel : le manque de données annotées pour assurer des performances suffisantes, pour des documents dont la collecte et l'annotation sont restreintes, pour des raisons de confidentialité, juridiques ou géographiques.

Cette architecture utilise un encodeur générique qui apprend des caractéristiques communes à plusieurs types de documents, et qui diminue le besoin en termes de quantité de

(a) Entrée (b) Prédictions (c) Quadrilatère (d) Rectification

FIG. 1 *: Localisation de documents par détection de coins. L'image d'entrée (a) est redimensionnée à 512x512 pixels. Un CNN prédit des cartes de chaleur pour les coins du document (b) (l'image montre la somme des quatre canaux, superposée à l'image d'entrée). Ces coins forment un quadrilatère (c) qui encadre le document. L'homographie entre ce quadrilatère et un rectangle estimé par le rapport hauteur/largeur du document permet de rectifier l'image (d).*

données annotées pour l'ajustement du décodeur, ce qui en fait une méthode frugale. Cela réduit le coût d'annotation et la puissance de calcul nécessaires pour entraîner des modèles performants et rapides. Nous testons notre travail sur un jeu de données interne et confidentiel d'images de documents, et nous démontrons son efficacité et sa capacité de généralisation.

2 État de l'art

2.1 Localisation de documents

La localisation de documents est un domaine de recherche qui existe depuis les débuts du traitement de documents capturés par des caméras. La littérature récente distingue quatre types d'approches : la détection de contours, la détection de points-clés, le template matching et la segmentation.

Méthodes basées sur la détection de contours : Ces méthodes, comme (Tropin et al., 2021), se basent sur les méthodes classiques de traitement d'image, qui détectent les bords des documents avec des détecteurs de contours, de la morphologie mathématique et des détecteurs de lignes ou de segments. Ces méthodes résistent à l'occlusion partielle du document (coin manquant), mais sont sensibles à l'éclairage et au contraste. Elles ne peuvent pas prédire l'orientation du document.

Méthodes basées sur la détection de coins : Ces méthodes, comme (Zhu et al., 2019), détectent les quatre coins des documents rectangulaires pour corriger la perspective. Ces méthodes sont sensibles à l'occlusion partielle, mais peuvent prédire l'orientation du document si les coins sont étiquetés. Notre approche est de ce type.

Méthodes basées sur du template matching : Ces méthodes, comme (Chiron et al., 2021), exigent une classification du document dans l'image, avant ou pendant le template matching. Un critère de similarité entre l'image et les modèles permet d'estimer une transformation entre les deux espaces. Ces critères se basent souvent sur des points-clés. L'orientation du document est donnée par le template matching.

Méthodes basées sur la segmentation : Ces méthodes, comme (Castelblanco et al., 2020), traitent la localisation de documents comme un problème de segmentation. Elles estiment un

masque du document, puis le transforment en quadrilatère pour obtenir la position. Ces méthodes demandent souvent un post-traitement, basé sur les contours, pour prédire l'orientation du document.

Des approches hybrides existent également, comme (Wu et al., 2023) qui rassemble prédiction de coins, de contours et classification dans un seul modèle CNN.

2.2 Les architectures encodeur-décodeur

Un encodeur est une partie d'un réseau de neurones qui extrait des caractéristiques haut niveau des images. Ces caractéristiques ont une dimension spatiale réduite, mais plus de canaux que l'image. Le décodeur est la partie entre l'encodeur et la sortie finale. Pour le traitement d'image, des CNN populaires comme ResNet (He et al., 2016) sont souvent utilisés comme encodeur et pré-entraînés sur de grands ensembles de données, comme ImageNet (Russakovsky et al., 2015). Des architectures récentes, comme SegNet, ont amélioré les performances sur les tâches de conversion image - image, comme la segmentation sémantique ou la détection de contours (Ji et al., 2021).

3 Méthodologie

3.1 Architecture du modèle

Notre modèle, Figure 2, est un U-Net (Ronneberger et al., 2015) adapté pour localiser des points-clés. Nous avons testé quatre scénarios de séparation entre l'encodeur et le décodeur, pour trouver le bon équilibre entre le gel de l'encodeur et la convergence du décodeur. Nous nous concentrons sur les documents rectangulaires, qui se localisent avec quatre coins.

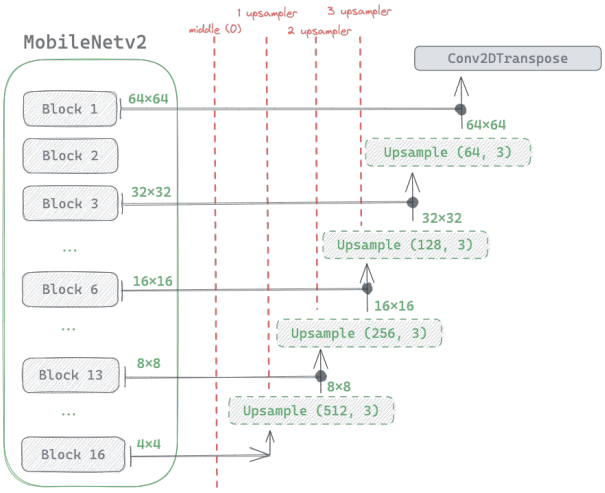


FIG. 2 *: Architecture du modèle. Les pointillés rouges représentent les séparations testées.*

Nous avons utilisé un MobileNetV2 (Sandler et al., 2018) comme encodeur de notre modèle, inspiré par LDRNet (Wu et al., 2023). Cela nous permet d'avoir un réseau plutôt léger

et pré-entraîné sur ImageNet. Nous avons ajouté des connexions résiduelles entre le MobileNetV2 et les couches de suréchantillonnage, comme dans un U-Net, pour reconstruire les sorties en pleine résolution.

3.2 Dataset

Nous avons pris au hasard 8942 images de documents dans un ensemble de données privées. Cet ensemble contient des données personnelles et ne peut pas être partagé. Ces images sont de cinq classes : Carte d'Identité (ID, 21%), Permis de Conduire (DL, 19%), Passeport (P, 25%), Titre de Séjour (RP, 14%) ou Certificat d'Immatriculation (VRC, 21%). La Figure 3 montre des spécimens représentatifs de chaque classe. Nous avons divisé chaque classe en entraînement (70%), validation (15%) et test (15%). Toutes les images ont été étiquetées manuellement, en déterminant la position des quatre coins.

(a) ID (b) DL (c) P (d) RP (e) VRC

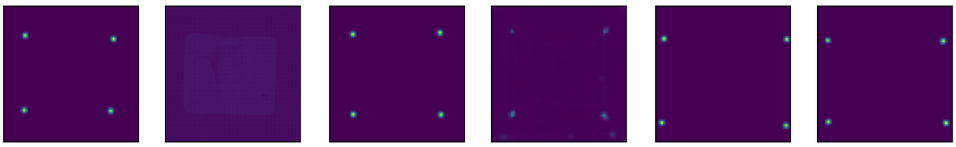
FIG. 3 *: Spécimens des cinq classes de document.*

3.3 Détails d'entraînement

Les coordonnées des coins sont converties en cartes de chaleur de 512x512. Les sorties du modèle sont des images noires avec une gaussienne pour chaque coin. Chaque canal correspond à un coin. Cela donne l'orientation des documents. Nous augmentons les données avec des opérations aléatoires : recadrage, redimensionnement, rotation, perspective, éclairage, bruit.

Nous implémentons les modèles avec TensorFlow. Nous utilisons Adam (Kingma et Ba, 2015) avec un learning rate de $2e^{-4}$ pour minimiser l'erreur quadratique moyenne. Nos modèles sont entrainés jusqu'à la stagnation de l'erreur sur l'ensemble de validation.

Nous utilisons pour nos entrainements des GPU Nvidia, TITAN X ou GTX 1080 Ti. La Figure 4 montre des sorties de nos modèles durant l'entraînement.



(a) Après 1 epoch (b) Après 10 epochs (c) Après 100 epochs

FIG. 4 *: Exemples de cartes de chaleur prédites par un de nos modèles pour trois documents différents. À gauche : la somme des annotations, à droite : la somme des prédictions.*

4 Expériences

Dans un premier temps, nous nous intéressons à la question de la séparation du modèle en un encodeur et un décodeur. Nous cherchons un bon compromis entre : avoir un encodeur partagé qui inclut beaucoup de paramètres permettant un ajustement rapide du décodeur et avoir un décodeur avec suffisamment de paramètres pour pouvoir être performant sur de nouvelles classes de documents.

4.1 Séparation encodeur décodeur

Nous avons étudié quatre possibilités de séparation, comme le montre la Figure 2, qui dépendent du nombre de blocs de suréchantillonnage présent dans l'encodeur : 0, 1, 2 ou 3.

Nous aurions pu aussi imaginer une séparation laissant tous les blocs de suréchantillonnage dans l'encodeur, laissant le décodeur avec uniquement la couche de convolution transposée, mais cette configuration est exclue de nos expériences, car cela laisse trop peu de paramètres dans le décodeur. Les différentes expériences sont comparées en utilisant l'Algorithme 1. Les poids de l'encodeur sont gelés durant l'étape d'ajustement du décodeur. Au total, 25 modèles ont été entrainés et les métriques suivantes ont été calculées :

- *IoU* (ou *indice de Jaccard*) : le rapport des aires de l'intersection sur l'union des quadrilatères. $\mathrm{IoU}(G, P) = \frac{\mathrm{A}(G \cap P)}{\mathrm{A}(G \cup P)}$. G et P étant les quadrilatères annoté et prédit.
- *Score* : L'intensité maximale de la carte de chaleur traitée comme un score de confiance.

Algorithme 1 Évaluation des séparations encodeur décodeur.

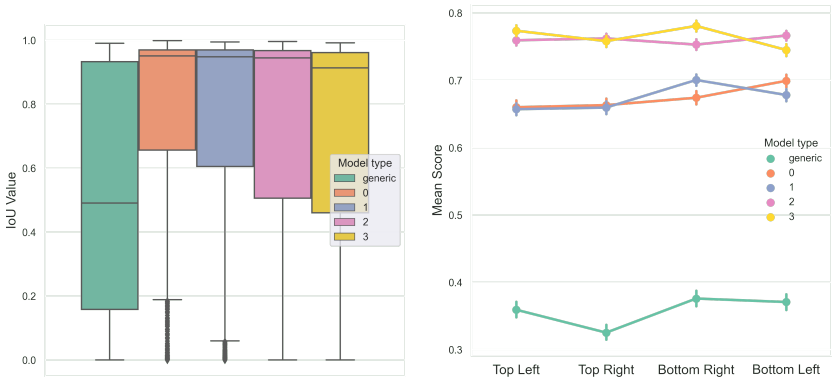
for *classe* in *5 classes de documents* **do**
  Entrainer un modèle M sur les données d'entraînement qui ne sont pas de la classe *classe*.
  Évaluer M sur la base de test de la classe *classe*.
  for *architecture* in *les 4 séparations encodeur décodeur* **do**
    Initialiser un modèle M' avec les poids de M.
    Entrainer M' sur les données d'entraînement de *classe* en gelant l'encodeur.
    Évaluer M' sur la base de test de la classe *classe*.
  end for
end for

Nous estimons les coordonnées des quatre coins en prenant le pixel le plus lumineux pour chaque carte de chaleur.

La Figure 5 montre les performances des modèles, pour les séparations décrites avant, sur leur test respectif. *Generic* est le modèle entraîné sur toutes les classes sauf celle du test. Les autres modèles sont ceux avec 0, 1, 2 ou 3 blocs de suréchantillonnage dans l'encodeur.

Les séparations avec 0 ou 1 bloc de suréchantillonnage dans l'encodeur semblent avoir des performances proches. Pour choisir l'architecture la mieux adaptée à nos besoins, nous considérons également le temps d'entraînement jusqu'à l'arrêt anticipé, présenté sur la Figure 6.

Le modèle avec 3 blocs de suréchantillonnage dans l'encodeur a moins de paramètres dans le décodeur, mais est plus lent à entraîner. Cela peut venir du nombre limité de paramètres



(a) IoU (b) Score de confiance

FIG. 5 *: Évaluation de nos modèles pour chaque séparation encodeur décodeur, entrainés sur quatre classes de documents et testés sur la cinquième.*

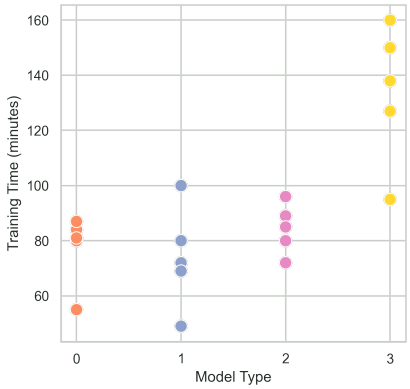


FIG. 6 *: Temps nécessaire à l'ajustement des décodeurs.*

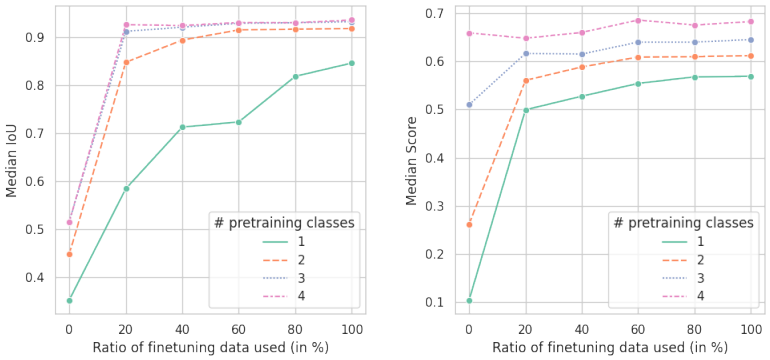
entraînables, qui demande plus d'epochs pour converger. Compte tenu de tous ces résultats, nous choisissons de continuer avec l'encodeur contenant 1 bloc de suréchantillonnage, un bon compromis entre performance et temps d'entraînement.

4.2 Capacité de généralisation

Nous évaluons comment l'architecture s'adapte au nombre de classes de documents pour l'entraînement de l'encodeur et à la quantité de données pour l'entraînement des décodeurs.

Nous pré-entraînons le modèle SDL-Net sur toutes combinaisons de classes, sauf le Permis de Conduire (DL), que nous utilisons pour l'ajustement d'un décodeur et l'évaluation. Nous faisons aussi deux autres expériences : entraîner un modèle sur DL seulement (M1) et entraîner un modèle sur toutes les classes, dont DL (M2).

Nous gelons les encodeurs des 17 modèles pré-entraînés, et nous ajoutons un décodeur pour DL. Nous varions la proportion (20, 40, 60, 80 et 100%) de données utilisées pour DL. La Figure 7 présente les résultats de l'évaluation. Par exemple, les métriques pour (1 classe, 20%) sont les tests des modèles pré-entraînés sur une classe (exceptée DL) et ajustés sur 20% de données DL.



(a) IoU (b) Score

FIG. 7 *: Résultats d'évaluation sur la classe DL. L'axe horizontal représente les pourcentages de documents d'entraînement de la classe DL utilisés lors de l'ajustement.*

Les résultats des deux expériences supplémentaires n'apparaissent pas dans la Figure 7. Pour (M1), l'IoU médian est de 0,59, et pour (M2), l'IoU médian est de 0,95.

Les résultats montrent que le modèle entrainé sur DL seulement (M1) est moins précis que les modèles pré-entraînés sur au moins une autre classe et ajustés sur DL avec 100% des données. Ils montrent aussi que les modèles pré-entraînés sur des classes variées sont meilleurs, même avec moins de données de la classe cible annotées. Le pré-entraînement sur des classes variées réduit le besoin marginal d'annotation et le temps de convergence.

5 Conclusion

Nous avons présenté une architecture encodeur-décodeur pour pré-entraîner un modèle de localisation de documents. Nous avons testé différentes séparations entre l'encodeur et le décodeur, et nous avons choisi la plus adaptée à notre cas, avec un compromis sur la performance et le temps d'entraînement. Nous avons évalué notre architecture sur plusieurs classes de documents, pour explorer la généralisation de notre modèle sur des documents inconnus. Les résultats ont montré que le pré-entraînement sur des données diversifiées et variées réduisait le besoin de données annotées pour la phase d'ajustement du décodeur. Nous avons montré que dans un contexte industriel, avec des données limitées sur des documents rares, nous pouvions améliorer la localisation en partageant un encodeur entraîné sur la détection de coins pour d'autres classes.

Références

Burie, J.-C., J. Chazalon, et M. Coustaty (2015). Icdar2015 competition on smartphone document capture and ocr (smartdoc). In *2015 13th ICDAR*, pp. 1161–1165. IEEE.

Castelblanco, A., J. Solano, C. Lopez, et E. Rivera (2020). Machine learning techniques for identity document verification in uncontrolled environments : A case study. In *Pattern Recognition : 12th Mexican Conference, MCPR 2020, Proceedings 12*, pp. 271–281. Springer.

Chiron, G., N. Ghanmi, et A. M. Awal (2021). Id documents matching and localization with multi-hypothesis constraints. In *25th ICPR*, pp. 3644–3651.

He, K., X. Zhang, S. Ren, et J. Sun (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778.

Ji, Y., H. Zhang, Z. Zhang, et M. Liu (2021). Cnn-based encoder-decoder networks for salient object detection : A comprehensive review and recent advances. *Information Sciences 546*.

Kingma, D. et J. Ba (2015). Adam : A method for stochastic optimization. In *ICLR*.

Ronneberger, O., P. Fischer, et T. Brox (2015). U-net : Convolutional networks for biomedical image segmentation. In *MICCAI*, pp. 234–241. Springer.

Russakovsky, O., J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision 115*(3), 211–252.

Sandler, M., A. Howard, M. Zhu, A. Zhmoginov, et L.-C. Chen (2018). Mobilenetv2 : Inverted residuals and linear bottlenecks. In *IEEE*, pp. 4510–4520.

Tropin, D. V., I. A. Konovalenko, N. S. Skoryukina, D. P. Nikolaev, et V. V. Arlazarov (2021). Improved algorithm of id card detection by a priori knowledge of the document aspect ratio. In *Thirteenth International Conference on Machine Vision*, Volume 11605, pp. 407–415.

Wu, H., H. Qian, H. Wu, et A. van Moorsel (2023). Ldrnet : Enabling real-time document localization on mobile devices. In *International Workshops of ECML 2022*, pp. 618–629.

Zhu, A., C. Zhang, Z. Li, et S. Xiong (2019). Coarse-to-fine document localization in natural scene image with regional attention and recursive corner refinement. *IJDAR 22*(3), 351–360.

Summary

Structured documents analysis and recognition are essential for modern online on-boarding processes, and document localization is a crucial step to achieve reliable key information extraction. While deep-learning has become the standard technique used to solve document analysis problems, real-world applications in industry still face the limited availability of labelled data and of computational resources when training or fine-tuning deep-learning models. To tackle these challenges, we propose SDL-Net: a novel U-Net like encoder-decoder architecture for the localization of structured documents. Our approach allows pre-training the encoder of SDL-Net on a generic dataset containing samples of various document classes, and enables fast and data-efficient fine-tuning of decoders to support the localization of new document classes. We conduct extensive experiments on a proprietary dataset of structured document images to demonstrate the effectiveness and the generalization capabilities of the proposed approach.