Réseaux de neurones convolutifs profonds pour la détection et la localisation de navires sur des images satellitaires

Jean-Jacques Szkolnik, Abdel-Ouahab Boudraa

IRENav (EA 3634), Ecole Navale/Arts-Métiers ParisTech, BCRM Brest, CC 600, 29240 Brest Cedex 9, France.
(jj.szkolnik, abdel.boudraa)@ecole-navale.fr

Résumé – La surveillance maritime est un enjeu stratégique à la fois économique, environnemental et sécuritaire. Elle fait appel à des données satellitaires. Les flux de données sont colossaux et l'automatisation de l'interprétation des données est devenue indispensable. Dans ce travail on s'intéresse à la surveillance maritime par la détection automatique des navires à partir d'images satellites. Nous proposons une solution basée sur les réseaux de neurones convolutifs profonds pour évaluer l'apport potentiel de ces techniques à la surveillance maritime. L'architecture U-Net du réseau mis en oeuvre a été initialement développée dans le milieu médical afin d'effectuer la segmentation sémantique d'images. Ce type de réseau est particulièrement adapté à une segmentation comportant deux classes, ce qui correspond à la problématique de détection de navires. Les résultats obtenus, à partir d'un réseau dimensionné en fonction de nos moyens matériels notamment nous encouragent à poursuivre dans cette voie.

Abstract – Maritime surveillance is a strategic economic, environmental and security issue, and satellite observations are used to ensure such task. Data flows are very large and automation of their interpretation has become indispensable. In this work we focus on the maritime awareness by the automated detection of ships using satellite images. Our aim is to evaluate the potential of deep convolutional neural networks as a solution to maritime surveillance problem. The implemented network U_Net architecture has initially been developed in the medical sector in order to segment images. This type of network is particularly well suited to a segmentation with two classes, which corresponds to the ships detection problem. Moreover, the results obtained, from a network dimensioned according to our material means encourage us to continue in this way.

1 Introduction

La surveillance maritime constitue un enjeu majeur à la fois écologique, économique et sécuritaire. On peut citer la surveillance des pollutions par hydrocarbures et l'identification des pollueurs, le trafic maritime, les activités de pêche illégale ou les actes de pirateries. Ainsi, dans ces domaines, les systèmes d'alerte doivent être réactifs et autonomes. La surveillance passe notamment par la détection automatique en temps réel des navires à partir d'images de satellites. Le gigantesque flot d'images rend la détection par les outils classiques compliquée [1],[2],[3]. De plus, la détection des navires est un grand défi en vision par ordinateur, principalement du fait de la grande variété des dimensions de ces derniers ou de leur environnement bruité qui peut conduire à de nombreuses fausses alarmes (Fig. 1). Une solution à ce problème est l'apprentissage profond, qui ces derniers temps, a connu un grand succès. C'est notamment le cas des réseaux de neurones convolutifs (CNN) qui ont été utilisés avec succès dans beaucoup d'applications relevant de la classification d'images. Les images utilisées sont issues de la base annotée de AIRBUS [5] dont un échantillon est montré figure 1.

2 Les CNN

La thématique centrale de ce travail est la segmentation sémantique compte tenu des évolutions futures prévisibles vers une plus grande précision dans la fouille d'images.

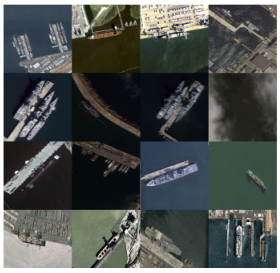

Figure 1 – Base de données AIRBUS.

Les CNN constituent une réponse à cette problématique car ils sont particulièrement adaptés à l'apprentissage automatique des caractéristiques discriminantes des images, au traitement des structures spatiales, et permettent d'atteindre des performances élevées tout en gérant la variabilité des objets. Les CNN sont assez similaires aux réseaux de neurones ordinaires de type perceptron multicouche. Ils procèdent cependant, d'une manière différente, le nombre de connexions d'une couche à une autre est limité en préservant toutefois la corrélation locale des images. Les couches constitutives des CNN sont les suivantes :

La couche d'entrée ou volume d'entrée

Cette couche d'entrée prend en compte tous les pixels de l'image, dans le cas d'images couleurs on parle alors de volume d'entrée.

La couche convolutive discrète (CONV)

La couche CONV est définie par trois hyper-paramètres qui déterminent le volume de cette dernière.
1) Le premier hyper-paramètre définit la profondeur de la couche et correspond au nombre de noyaux de convolution appliqués à chacune des couches du volume d'entrée.
2) Le second hyper-paramètre est le "pas", qui définit le taux de recouvrement des champs récepteurs et par la même le nombre de neurones des cartes d'activation.
3) Le troisième hyper-paramètre est la "marge". Pour que la surface à la sortie de la couche soit identique à celle du volume d'entrée il faut étendre la surface du volume d'entrée en ajoutant des zéros pour que la convolution en bord d'image puisse être réalisée. Le noyau de convolution représente une caractéristique recherchée dans l'image, plus une valeur de la carte d'activation est élevée et plus l'emplacement correspondant dans l'image ressemble à la caractéristique recherchée. Les caractéristiques de l'image sont apprises par le réseau qui positionne les poids des noyaux de convolution de manière appropriée au cours de la phase d'apprentissage à l'aide de l'algorithme de rétropropagation du gradient. Les CNN s'adaptent donc automatiquement au problème qui leur est posé, par identification des caractéristiques pertinentes de l'image.

La convolution transposée (CONV-TRANS) permet de reconstruire une image à partir d'une carte d'activation et d'un noyau de convolution.

La couche de correction

Elle revêt le plus souvent la forme de fonctions d'activation non-saturantes qui limitent certains artefacts lors de l'apprentissage. Les fonctions ReLU $f(x) = max(0,x)$ (*Rectified Linear Unit*) et leurs variantes sont parmi les plus utilisées.

La couche de *pooling*

Cette couche permet de sous-échantillonner l'image, elle est généralement située après une couche de correction. Son rôle est de limiter les effets de sur-apprentissage en remplaçant un groupe de pixels de l'image par un seul pixel dont la valeur peut être, la moyenne des valeurs des pixels du groupe, la valeur d'un pixel tiré aléatoirement ou la valeur maximale des pixels du groupe. C'est souvent cette dernière option (*maxpooling*) qui est privilégiée car elle présente l'avantage de favoriser les activations fortes.

La couche "totalement connectée" (FC)

La couche FC prend souvent place en fin de réseau lorsque toutes les étapes de *maxpooling* ont été effectuées. Elle réalise les fonctions de haut niveau en termes de raisonnement (classification) compte tenu des caractéristiques profondes contenues dans la dernière couche de convolution, elle réalise le lien entre les caractéristiques d'une image et une classe. La couche FC concentre l'essentiel des paramètres du réseau et est sensible au sur-apprentissage. Pour cette raison elle est parfois précédée d'une couche de *dropout* qui permet d'éteindre certains neurones choisis aléatoirement afin de limiter les risques de sur-apprentissage.

La couche de perte

C'est la dernière couche du réseau, elle en fixe la fonction, par exemple la prédiction d'une classe parmi n classes exclusives utilise la fonction "Softmax" qui permet de transformer le score de chacune des sorties en probabilité.

Les modèles de CNN

Les CNN sont constitués par une mise en cascade des différents éléments présentés ci-avant. Ces modèles ont été entrainés à partir de bases de données annotées elles aussi à disposition (par exemple ImageNet). Il est donc possible de développer par transfert d'apprentissage un réseau adapté au problème à traiter. Comme le réseau est déjà entraîné sur des caractéristiques communes il suffit de procéder à un affinement des poids du modèle (*fine tuning*) à l'aide d'une base de donnée annotée plus modeste.

3 Détection de navires

L'application visée ne consiste pas uniquement à détecter des navires, mais également à en déterminer la position à l'aide de la segmentation de l'image.

3.1 Les données et le contexte.

La base de données comprend les images, d'un volume approximatif de 30 Go, auxquelles vient s'ajouter le fichier des masques (annotation) indiquant la position des pixels considérés appartenir à l'objet navire [5].

FIGURE 2 – Image navire.

FIGURE 3 – Image navire avec boite.

3.1.1 Caractéristiques des données.

Les images sont de dimension 768×768×3 (RGB) et proviennent de la découpe d'images de plus grandes dimensions obtenues par les satellites Pléiades. La base 192555 images dont 149999 ne contiennent aucun navire. Le nombre total de navires est de 81723 répartis dans 42556 images avec un nombre d'images qui décroit avec la densité des navires qu'elles comprennent. La haute résolution des images rend possible de traiter les navires à quai et dans les ports. Avec plus de 75% des images qui ne comportent aucun na-

TABLE 1 – Caratéristiques

Base App.	Base Test	Nb CONV
31910 images	10639 images	14 couches

vire, la base de données apparaît déséquilibrée. De même, la présence de navires dans une image reste un évènement de faible intensité compte tenu de la disproportion entre le nombre de pixels qui leurs correspondent et le nombre total de pixels de l'image. Ces considérations nous amènent à envisager une base d'apprentissage remaniée constituée de 75% des images comportant au moins un navire auxquelles nous ajoutons un jeu d'images ne comportant aucun navire dans la proportion respective de 2/3, 1/3. Le reste des images constitue la base de test. Pour limiter les risques de sur-apprentissage nous procédons à une augmentation des données en appliquant aléatoirement à chacune des images de la base d'apprentissage une transformation (permutation des axes, rotation, bruit additif). Deux exemples d'images sont représentées en figures (4) et (5).

FIGURE 4 – Sans navire.

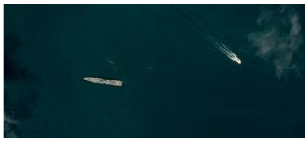

FIGURE 5 – Plusieurs navires.

3.1.2 Environnement matériel et logiciel.

Les CNN nécessitent de nombreuses opérations tensorielles, notamment lors de la phase d'apprentissage, ces dernières peuvent être parallélisées à l'aide de GPU. Nous disposons d'une carte GPU de type Tesla NVIDIA K40c à 2880 coeurs ce qui permet un gain significatif en termes de temps d'apprentissage. Pour les opérations séquentielles nous disposons de 40 coeurs CPU en *multithreading*. Notre choix s'est porté sur la librairie Tensorflow sous Python et la suite logicielle associée CUDA et cuDNN qui offrent un cadre adapté au développement de CNN sur GPU. Nous avons réalisé l'implémentation à partir de l'API Keras qui comprend la totalité des modèles disponibles au public.

3.2 Modèle et apprentissage.

Bien que YOLO soit un algorithme rapide et efficace pour la détection d'objets dans une image et constitue à ce titre l'état de l'art en la matière, la mise en oeuvre d'un algorithme de type U-Net se justifie lorsque la localisation précise des objets et la segmentation sémantique sont des priorités. Le modèle U-Net, est un réseau symétrique dit en U utilisé à l'origine pour la segmentation d'images médicales [6], avec une partie dite "encodeur" qui réduit la dimension de l'image tout en capturant les caractéristiques importantes, et une partie "decodeur" qui permet de reconstruire l'image segmentée à partir des caractéristiques extraites.

3.2.1 Architecture choisie

Le détail du réseau utilisé est donné par la figure 6. Les flèches indiquent des liaisons directes entre l'encodeur à gauche et le décodeur à droite. Les dernières cartes d'activation juste avant l'opération de *maxpooling* de l'encodeur sont directement réinjectées au niveau du décodeur et, concaténées avec une couche de CONV-TRANS juste avant de subir deux convolutions successives. Il n'y a pas en sortie du décodeur de couche FC afin de conserver la topologie de l'image. La couche FC est simplement remplacée par une couche de convolution avec un unique filtre de surface 1, si bien que les sorties correspondent à une combinaison linéaire de la dernière carte d'activation, activées par une fonction Sigmoïde.

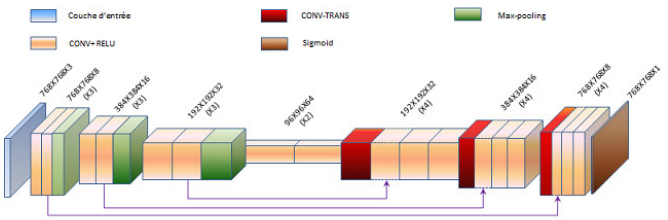

FIGURE 6 – Réseau U-Net

3.2.2 Apprentissage

L'algorithme de retro-propagation du gradient de l'erreur implémenté est l'algorithme Adam [7], c'est une évolution de l'algorithme stochastique de descente du gradient. Son principe repose sur une adaptation individuelle des taux d'apprentissage en fonction des paramètres à partir des deux premiers moments du gradient. La mise à jour des paramètres du réseau s'effectue par groupes d'images (*batch*), tirées aléatoi-rement dans la base d'apprentissage de manière à décorréler l'apprentissage entre *epoch*, et dont la taille résulte d'un compromis afin de limiter les risques de sur-apprentissage et d'instabilité de la descende du gradient. La métrique utilisée est basée sur la formule (1). Elle est adaptée au problème et permet de mesurer la pertinence de la prédiction par rapport à la vérité terrain en fournissant une mesure du taux de recouvrement.

$$IoU(A,B) = \frac{A \cap B}{A \cup B} \tag{1}$$

3.3 Résultats.

L'évaluation des résultats en segmentation d'images s'effectue classiquement à l'aide d'une métrique dédiée, ici le *F2 Score* qui fournit une mesure combinée de la précision et du rappel. On définit des seuils de détection qui conditionnent directement le taux de recouvrement défini par

(1) puis, pour chaque image on effectue la moyenne des *F2 Score* calculés pour chacun des seuils, on obtient alors un *F2 Score* moyen pour chaque image. Enfin, on réalise la moyenne des *F2 Score* moyens des images de la base de test. On justifie l'utilisation du *F2 Score* car on privilégie le rappel à la précision, c'est à dire que l'on désavantage la non détection par rapport au fait d'avoir un plus grand nombre de fausses détections (minimisation des faux négatifs). Le *F1 Score* est, par exemple, une mesure équilibrée entre rappel et précision à utiliser lorsque les classes sont d'égale importance. La figure 7 donne un exemple typique de résultat. L'image de gauche est l'image RGB d'entrée, celle du milieu représente la vérité terrain et celle de droite la prédiction obtenue. Le *F2 Score* moyen de cette image est de 0.52. Il est vrai qu'à la vue de cet exemple, notamment en termes de netteté de l'image, on peut se poser la question de la nécessité d'utiliser une technique qui nécessite une telle base d'apprentissage, un Support Vector Machine, par exemple, pourrait fournir de bons résultats. On opte pour l'utilisation des CNN en raison de leur aptitude à la prise en compte de la variabilités des images à traiter et par leur capacité de généralisation, deux caractéristiques prégnantes des CNN. Pour l'ensemble de

FIGURE 7 – Images d'entrée (à gauche). Vérité terrain (centre). Image prédite par le réseau (à droite).

la base de test nous obtenons un score proche de 0.65, qualitativement les résultats remplissent la fonction de détection visée. Ils sont néanmoins en retrait par rapport aux équipes de tête du défi AIRBUS (score au delà de 0.8). Cela s'explique par des bases d'entraînement et de test différentes (20000 navires de moins pour la base d'entraînement), par le fait que ne disposant que d'un seul GPU nous sommes contraint tant par la profondeur de notre modèle que par la taille des *batch* et, que pour en conserver la maîtrise, notre modèle ne résulte pas d'un transfert d'apprentissage (Tables 1 et 2). Pour finir, on réalise un test sur une vidéo tournée avec un drone au dessus de la marina de l'Ecole Navale figure 8. On cherche ici à évaluer qualitativement la capacité de généralisation de l'algorithme et à le comparer avec YOLO. L'algorithme YOLO a été entraîné ab initio avec notre base de données. Notre algorithme a été légèrement modifié, notamment en sortie avec l'adjonction d'une couche FC et "Softmax". Globalement les résultats sont comparables chacun avec ses particularités.

4 Conclusion et perspectives

Les librairies dédiées au *Deep Learning* fournissent des API de haut niveau qui permettent de développer et de

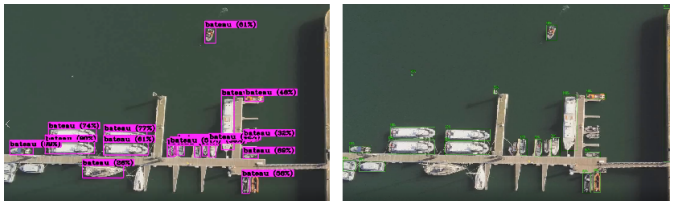

FIGURE 8 – Résultats obtenus avec YOLO (à gauche) et avec notre algorithme (à droite).

déployer des réseaux rapidement. Dans cette étude, qui adresse la problématique de la segmentation d'images et qui porte sur la détection automatisée de navires dans des images satellites haute résolution, nous implémentons un modèle convolutif profond dont l'architecture provient du milieu médical. Un compromis entre la complexité du modèle (nombre de paramètres à optimiser) et la taille du *batch* nous permet d'implémenter la phase d'entraînement sur un GPU. Les résultats sont encourageants, compte tenu des contraintes, et nous envisageons quelques évolutions sur la structure du modèle. Nous prévoyons également d'utiliser un modèle pré-entrainé et de procéder à un transfert d'apprentissage, le réglage fin étant réalisé à partir de notre base de données. Enfin, il est prévu que nous disposions d'un jeu complémentaire de données satellites. En classification et segmentation d'images les CNN profonds se sont incontestablement imposés. Cependant, ils nécessitent, pour l'apprentissage supervisé, d'importants volumes de données annotées vu le nombre parfois très importants de paramètres à optimiser. La constitution ces de bases de données figure le challenge des années à venir.

Références

[1] D.J. Crisp, The state-of-the-art in ship detection in SAR imagery, Technical report, Defense Science and Technology Organisation - Australia, 2004.

[2] M. Tello et al., A novel algorithm for ship detection in ENVISAT SAR imagery based on the wavelet transform. Proc. ERS Sym., vol. 572, pp. 1-6, 2005, Salzburg.

[3] J. Shi, Z. Jiang and H. Zhang, Few-shot ship classification in optical remote sensing images using nearest neighbor prototype representation, J. Sel. Top. Appl. Earth Obs. Remote Sens. vol. 14, pp. 3581-3590, 2021.

[4] Y. Le Cun et al., Handwritten digit recognition with a back-propagation network, Advances in NIPS 2, D.S. Touretzky, Ed., 1990, pp. 396–404.

[5] https ://www.kaggle.com/c/airbus-ship-detection

[6] O. Ronneberger and P. Fischer, T. Brox U-Net : Convolutional Networks for Biomedical Image Segmentation. MICCAI, 2015.

[7] D.P. Kingma and J.L. Ba Adam : A Method for Stochastic Optimization, Int. Conf. Learning Rep., 2015.