Département de géomatique appliquée

Faculté des lettres et sciences humaines

Université de Sherbrooke

Adaptation des architectures ADDA et semi-ADDA pour la détection d'objets par apprentissage profond sur les images satellites THR

par

Étienne Lauzier-Hudon

Mémoire présenté pour l'obtention du grade de Maître ès sciences géographiques (M.Sc.)
cheminement Télédétection

Avril 2022

IDENTIFICATION DU JURY

Le 18 avril 2022 : le jury a accepté le mémoire de Étienne Lauzier-Hudon dans sa version finale.

Directeur de recherche : Yacine Bouroubi
Codirecteur de recherche : Nouri Sabo (Centre Canadien de cartographie et d'observation de la terre)
Codirecteur de recherche : Samuel Foucher (Centre de Recherche Informatique de Montréal)

Membres du jury :
Mickaël Germain (Département de géomatique appliquée, Université de Sherbrooke)
Pierre-Luc St-Charles (Institut québécois d'intelligence artificielle - MILA)

RÉSUMÉ

La détection d'objets est une tâche fondamentale en vision par ordinateur ayant pour but d'identifier et de localiser différentes catégories d'objets dans une image selon une taxonomie prédéfinie. Entraînées par l'émergence des réseaux de neurones convolutifs (CNNs, *convolutional neural networks*), de nombreuses approches de détection d'objets basées sur les CNNs ont été proposées dans les dernières années améliorant considérablement les performances obtenues par les algorithmes traditionnels. Or, bien que les CNNs obtiennent généralement de très bonnes performances sur cette tâche, de récentes publications soulignent leurs lacunes de généralisation lorsqu'appliqués à de nouvelles régions géographiques jamais ou très peu vues lors de l'entraînement. En pratique, il serait intéressant que les modèles puissent être étendus à de nouvelles régions géographiques où il n'existe pas nécessairement de données d'entraînement.

Le champ de recherche essayant de résoudre ce problème est celui de l'adaptation de domaine (DA, *domain adaptation*). Une nouvelle méthode de DA basée sur l'apprentissage adverserielle nommée *Adversarial Discriminative Domain Adaptation* (ADDA) s'est récemment démarquée en surpassant les autres méthodes de DA sur des tâches d'adaptation de domaine entre les jeux de données de classification de caractères écrits MNIST, USPS et SVHN. Peu longtemps après, une légère modification fut proposée (semi-ADDA) pour rendre le processus semi-supervisé. Cependant, bien que les performances atteintes par ADDA et semi-ADDA soient supérieures à celles des autres méthodes de DA, la tâche visée par ces architectures en est une de classification. Or, en télédétection, on retrouve généralement beaucoup plus d'un objet par image. Il devient donc nécessaire d'adapter les approches de DA afin de permettre l'extraction de plusieurs objets au sein d'une image, soit par détection d'objets ou par segmentation. Lors de la revue de littérature effectuée dans le cadre de ce projet (automne 2018), très peu de travaux s'étaient concentrés sur l'adaptation de domaine appliquée aux autres tâches de vision par ordinateur. Dans le cadre du présent projet, une modification des architectures ADDA et semi-ADDA est proposée afin de passer d'une tâche de classification vers une tâche de détection d'objets, et ce en remplaçant l'architecture de classification de ADDA et semi-ADDA par une architecture de détection d'objets, à savoir Faster R-CNN. L'adaptation de domaine est effectuée pour 2 classes en concordance entre les taxonomies des jeux de données xView et GeoImageNet, où xView est le domaine source et GeoImageNet le domaine cible. Les images de ces jeux de données ne proviennent ni du même territoire, ni du même capteur, ce qui constitue une tâche

complexe de DA. À titre comparatif, 4 approches différentes sont considérées, soit aucune adaptation de domaine, ADDA, semi-ADDA et un *fine-tuning* conventionnel. Les résultats sont comparés sur un ensemble de test composé de 60 images annotées provenant du domaine cible, GeoImageNet. Les résultats montrent l'efficacité de ADDA et semi-ADDA même lorsqu'étendu à la détection d'objets, et ce avec un passage de 64,2 % de *mean Average Precision* (mAP) sans adaptation de domaine à 87,2 % après ADDA et 95,5 % après semi-ADDA. Le *fine-tuning* obtient pour sa part un mAP de 89,5 %.

Citation : Lauzier-Hudon, É., 2022, Adaptation des architectures ADDA et semi-ADDA pour la détection d'objets sur les images satellites THR, Mémoire de maîtrise, Département de géomatique appliquée, Université de Sherbrooke, 63 p.

Mots-clés : apprentissage profond, adaptation de domaine, détection d'objets, ADDA, semi-ADDA, télédétection, xView, GeoImageNet

TABLES DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ANNEXES

LISTE DES ANGLICISMES ET TRADUCTIONS

Puisque l'entièreté (ou presque) des papiers sur le *deep learning* sont en anglais et qu'il existe autour de ce domaine un jargon très spécifique, beaucoup d'anglicismes n'ont pas été traduit dans le texte. Plusieurs de ces termes ne possèdent pas de réelle traduction en français et essayer de les traduire aurait seulement rendu le texte plus lourd et aride à lire pour le lecteur averti. Pour assurer la compréhension du lecteur, la liste suivante a pour but de dresser les anglicismes utilisés dans le cadre de ce rapport, ainsi que leur traduction dans le texte (lorsqu'applicable).

Terme anglais	**Traduction utilisée dans le texte**
Adversarial	Adverseriel
Anchor	(Aucune)
Batch size	(Aucune)
Bounding box	(Aucune)
Bounding box regressor	Régresseur de *bounding box*
Convolutional Neural Network	Réseau de neurones convolutif
Deep learning	Apprentissage profond
Domain adaptation	Adaptation de domaine
Domain shift	Décalage de domaine
Dropout	(Aucune)
Feature map	(Aucune)
Fine-tuning	(Aucune)
Fully convolutional network	Réseau entièrement convolutif
Ground truth	Vérité terrain
Intersection over union	(Aucune)
Iverson bracket	Crochet d'Iverson
Learning rate	(Aucune)
Maximum mean discrepancy	Différence moyenne maximale
Multi layer perceptron	(Aucune)

Objectness	(Aucune)
Optimizer	Optimiseur
Oversampling	(Aucune)
Pansharpening	(Aucune)
Precision x Recall curve	Courbe de Précision x Rappel
Region of interest	Région d'intérêt
Representation/Embedding	Représentation
Support Vector Machine	(Aucune)
Transfer learning	Apprentissage par transfert

LISTE DES SIGLES ET ACRONYMES

ADDA	*Adversarial Discriminative Domain Adaptation*
AP	*Average precision*
BB	*Bounding box*
CNN	*Convolutional neural network*
CRIM	Centre de Recherche en Informatique de Montréal
D	Discriminateur
DA	*Domain adaptation*
E_c	Encodeur cible (CNN cible)
E_s	Encodeur source (CNN source)
FPN	*Feature Pyramid Network*
G	Générateur
GAN	*Generative adversarial network*
GIN	GeoImageNet
IoU	*Intersection over Union*
mAP	*Mean average precision*
MILA	*Montreal Institute of Learning Algorithms*
MLP	*Multi layer perceptron*
RGB	*Red, green, blue*
RoI	*Region of Interest*
RPN	*Region proposal network*
Semi-ADDA	*Semisupervised Adversarial Discriminative Domain Adaptation*
THR	Très haute résolution

REMERCIEMENTS

Ce mémoire est le fruit d'un travail de deux années qui n'aurait pu être réalisé seul. Nombreuses sont les personnes m'ayant guidé, accompagné et supporté dans ce processus.

Cette maîtrise s'est déroulée au sein du centre d'applications et de recherches en télédétection (Cartel) de l'Université de Sherbrooke. Je salue donc tous les membres de ce laboratoire, en espérant que l'application de l'apprentissage profond à la télédétection devienne une orientation de recherche de plus en plus poursuivie au département. Lorsqu'entraînés avec des données de qualité, les modèles d'apprentissage profond sont capables d'exploits extraordinaires. Bientôt, je pense que ces modèles surpasseront les performances du photo-interprète sur des tâches complexes comme la détection d'objets et la segmentation.

Je souhaite remercier particulièrement mon directeur de recherche, Yacine Bouroubi, pour son support technique, émotionnel et financier tout au long de ce projet. Je n'aurais pu demander un directeur plus compréhensif et humain. Je tiens également à adresser un remerciement particulier à Pierre-Luc St-Charles, nouvellement chercheur au MILA, à qui je dois une majeure partie de mes apprentissages en apprentissage profond. Sans les nombreux échanges de courriels, les rencontres virtuelles et les passages dans son bureau, ce projet serait complètement différent. Je remercie également l'entièreté de l'équipe du CRIM pour leur accueil sympathique et chaleureux dans leur équipe lors de mon stage de maîtrise.

Finalement, mes remerciements vont également à toutes les personnes qui ont participé de près ou de loin au succès de mon projet, notamment ma famille et mes amis. Une pensée particulière pour mon père, décédé durant ma maîtrise, qui aurait certainement été très fier de son fils. Ma dernière pensée est pour ma fabuleuse copine qui a dû supporter mes moments de désespoir et de découragement et qui, à chaque fois, m'a aidé à les surmonter.

1. Introduction

1.1. Cadre général

Ce projet de maîtrise s'effectue dans la veine du projet GeoImageNet. GeoImageNet (https://geoimagenet.ca/) est une plateforme de recherche collaborative qui vise à développer des applications de l'apprentissage profond pour la cartographie de l'occupation du sol et la détection des objets, à partir des images satellites très haute résolution (THR). Elle intègre plus de 10 000 km^2 d'images Pléiades 50 cm couvrant la majorité des grandes villes canadiennes ainsi que divers écosystèmes naturels. GeoImageNet contribue à l'effort grandissant d'exploitation des images satellites à l'aide des diverses architectures d'apprentissage profond.

La plateforme comporte une taxonomie de classes d'occupation du sol (environ 50) et une autre pour les classes de géo-objets (près de 200). L'annotation de ces classes est réalisée manuellement et parfois automatiquement à partir des données vectorielles de CanVec et OpenStreetMap. Les images annotées peuvent être récupérées par les utilisateurs afin de mettre sur pied des ensembles d'entrainement pour leurs algorithmes. Des architectures CNNs peuvent être téléchargés ou téléversée par les utilisateurs. La plateforme est ouverte aux organismes canadiens (universités, centre de recherche, industrie, etc.) qui œuvrent dans les domaines de la télédétection et de l'intelligence artificielle.

1.2. Problématique

En raison de sa vaste superficie, le Canada utilise l'imagerie satellitaire dans un grand nombre d'applications : la gestion du territoire, l'estimation des ressources naturelles, le suivi environnemental, la planification urbaine, la surveillance des infrastructures, le développement du Grand Nord et bien d'autres (Turgeon-Pelchat, 2019). Les images satellites THR en particulier comportent une mine considérable d'information en raison du niveau de détail important qu'elles contiennent. Cependant, l'extraction manuelle de cette information par photo-interprétation est une tâche fastidieuse et chronophage. Pour cette raison, beaucoup d'efforts ont été déployés dans la littérature au cours des dernières années pour proposer des méthodes d'analyse automatique de ces images (Voulodimos et coll., 2018)

Parmi ces méthodes, les réseaux de neurones convolutifs (CNNs, *convolutional neural networks*) ont obtenu des performances remarquables comme modèle pour l'extraction automatique d'information

au sein des images satellites (p. ex., bâtiments, routes, voitures, panneaux solaires) (Zhu et coll., 2017). Toutefois, les modèles obtenant d'excellentes performances dans la littérature sont ceux où les données d'entraînement et de test proviennent du même domaine, soit de la même région géographique, du même capteur et, idéalement, des mêmes conditions d'acquisition. Or, en pratique, on souhaite ultimement pouvoir appliquer les modèles sur de grands étendus géographiques avec des conditions d'acquisition changeantes et il est peu probable que des données d'entraînement soient disponibles pour chaque emplacement. Ce serait également souhaitable de pouvoir appliquer le modèle sur des images provenant d'un capteur légèrement différent (p. ex., Pléiades plutôt que Worldview).

Ceci est problématique puisque de récentes publications (Maggiori et coll., 2017; Wang et coll., 2017) ont montré les lacunes de généralisation des CNNs lorsqu'appliqués à de nouvelles régions géographiques jamais ou très peu vues lors de l'entraînement. Il en va de même pour des images provenant d'un nouveau capteur. Ces problèmes de généralisation sont dus, en partie, aux changements d'apparence (et donc de statistiques) des images en fonction du type de capteur et des conditions d'acquisition. Il existe également une variabilité statistique des caractéristiques des objets (couleur, forme, texture) d'une région géographique à l'autre.

Pour atténuer l'impact du changement de domaine, une solution est d'annoter davantage de données dans le nouveau domaine. Cette solution est toutefois loin d'être optimale parce que, comme mentionné précédemment, le processus d'annotation est long et fastidieux. Afin de pouvoir utiliser les CNNs dans de façon opérationnelle en télédétection, il devient nécessaire de développer des méthodes permettant de pallier les changement de domaine fréquemment observés en télédétection sans avoir à passer par l'annotation manuelle de données d'entraînement dans le nouveau domaine.

Le champ de recherche s'intéressant à ce problème est celui de l'adaptation de domaine (DA, *domain adaptation*). Le terme « domaine » fait référence à la distribution sous-jacente d'un jeu de données (Wang et Deng, 2018). En DA, on suppose qu'il existe un domaine « source » dans lequel une quantité abondante de données d'entraînement est disponible et un domaine « cible », quelque peu différent, dans lequel les données d'entraînement sont rares ou absentes. Le DA cherche à fournir des méthodes permettant de minimiser les pertes de performances occasionnées par le changement de domaine, soit de façon non supervisé (sans annotation dans le domaine cible) ou de façon semi-supervisé (avec quelques annotations dans le domaine cible). En télédétection, un changement de région

géographique, de capteur ou même de conditions d'acquisition peut être considéré comme un changement de domaine (Wang et coll., 2018).

Plusieurs méthodes d'adaptation de domaine ont été proposées au cours des dernières années (Wang et Deng, 2018). Parmi celles-ci, les méthodes basées sur un processus d'apprentissage adverseriel similaire à celui des GANs (Goodfellow et coll., 2014) ont démontré des résultats très prometteurs. La méthode *Adversarial Discriminative Domain Adaptation* (ADDA) (Tzeng et coll., 2017) s'est récemment démarquée en surpassant les autres méthodes de DA sur des tâches d'adaptation de domaine entre les jeux de données de classification de caractères MNIST (LeCun et coll., 1998a), USPS (Hull, 1994) et SVHN (Netzer et coll., 2011). L'approche initiale ADDA est toutefois basée sur un processus d'apprentissage non supervisé, où aucune donnée d'entraînement n'est disponible dans le domaine cible. Or, en télédétection, il est souvent possible d'obtenir de petites quantités de données étiquetées dans la région géographique cible. Dans cette optique, Wang et coll. (2018) proposent une modification à l'architecture initiale ADDA afin de rendre le processus semi-supervisé et de permettre l'utilisation des annotations disponibles dans le domaine cible. Ils nomment cette modification semi-ADDA (*semisupervised* ADDA).

ADDA et semi-ADDA, bien que présentant d'excellentes performances, se concentrent toutefois sur une tâche de classification (Voir Annexe 1). Lors de la revue de littérature effectuée dans le cadre de ce projet (automne 2018), très peu de travaux s'étaient concentrés sur l'adaptation de domaine appliquée aux autres tâches de vision par ordinateur. Or, puisque les images satellites contiennent généralement beaucoup plus d'un objet par image, la classification n'est définitivement pas la tâche la plus adaptée en télédétection. Il serait plutôt intéressant de pouvoir adapter ADDA et semi-ADDA afin de permettre l'extraction de plusieurs types d'objets (et leurs instances) au sein d'une image, soit par détection d'objets ou par segmentation. La détection d'objets, pour prendre cette tâche en exemple, est beaucoup plus complexe que la classification puisqu'il s'agit en fait d'une combinaison de classification et de localisation d'objets par régression. Cette tâche est donc moins « contrôlée » puisqu'il n'y a pas un seul objet par image. De plus, les objets ne sont pas centrés sur l'image et possèdent différentes échelles rendant cette tâche nettement plus complexe.

L'amélioration des capacités de généralisation des CNNs signifie également l'accélération de la création de nouveaux jeu de données. En effet, si les CNNs généralisaient bien, il serait possible de mettre à profit les jeux de données d'apprentissage profond de télédétection existants afin de réduire le fardeau d'annotation, et ce en entraînant un modèle sur un jeu de données existant et en effectuant

ensuite l'inférence à l'aide de ce modèle sur les nouvelles images que l'on souhaite annoter. Les détections pourraient ensuite être envoyées en validation par un humain, accélérant ainsi grandement le processus d'annotations.

Récemment, dans le but de mettre sur pied un jeu de données spécifique au territoire canadien, le projet GeoImageNet (https://geoimagenet.ca/) a récemment vu le jour (Bouroubi et coll., 2019). Ce projet a permis la création d'une plateforme de recherche permettant l'annotation collaborative d'images satellites pour les chercheurs et organismes du secteur géospatial canadien. Les annotations recueillies sur cette plateforme serviront à la création d'un jeu de données pancanadien d'apprentissage profond. Ce dernier permettra la réalisation d'applications de l'apprentissage profond à la télédétection sur le territoire canadien. La taxonomie actuelle de la plateforme contient 48 classes d'occupation du sol et 178 classes d'objets.

Or, l'annotation manuelle d'un jeu de données d'une telle envergure est une tâche colossale. C'est dans cette optique que le projet actuel s'oriente. En améliorant les méthodes d'adaptation de domaine ADDA et semi-ADDA, il devient possible de mettre à profit les jeux de données d'apprentissage profond de télédétection existants pour réduire substantiellement le fardeau d'annotation inhérent à la création d'un jeu de données aussi colossale.

1.3. Objectifs et hypothèse de recherche

L'objectif principal du projet vise à adapter les architectures d'adaptation de domaine ADDA et semi-ADDA de la classification vers la détection d'objets, et ce en remplaçant le modèle de classification de ces architectures par un modèle de détection d'objets. L'adaptation de domaine est effectuée sur de l'imagerie satellitaire THR où les images sources et cibles proviennent de régions et de capteurs différents, ce qui constitue une tâche complexe de DA. Les performances de ADDA et semi-ADDA sont comparées à un *fine-tuning* conventionnel et à un modèle non-adapté et non réentraîné. Il en découle les objectifs spécifiques suivants :

1) Entraîner le modèle Faster R-CNN sur le domaine source.
2) Évaluer directement les performances du modèle pré-entraîné sur le domaine source sur l'ensemble de test du domaine cible sans appliquer d'adaptation domaine.
3) Évaluer les performances du modèle pré-entraîné sur le domaine source sur l'ensemble de test du domaine cible après un *fine-tuning* conventionnel sur le domaine cible.

4) Évaluer les performances du modèle pré-entraîné sur le domaine source sur l'ensemble de test du domaine cible après une adaptation de domaine avec ADDA et semi-ADDA.

L'hypothèse posée est que l'application de ADDA permettra d'obtenir de meilleures performances sur le domaine cible que si aucune adaptation de domaine n'est effectuée. De plus, semi-ADDA devrait surpasser les autres méthodes puisqu'un processus d'entraînement semi-supervisée est mis en place contrairement à un processus purement non-supervisé pour ADDA.

2. Cadre théorique

2.1. Détection d'objets sur l'imagerie aérienne et satellitaire

La détection d'objets est une tâche fondamentale en vision par ordinateur ayant pour but d'identifier et de localiser, via une *bounding box* (BB), différentes catégories d'objets (et leurs instances) dans une image selon une taxonomie prédéfinie (Russakovsky et coll., 2015). En télédétection, la détection d'objets joue un rôle important dans plusieurs applications, telles que la surveillance environnementale, la détection de danger géologique, la cartographie de l'occupation du sol, la mise à jour des systèmes d'information géographique, l'agriculture de précision, la planification urbaine, la surveillance des infrastructures, la gestion des catastrophes majeures et autres (Cheng et Han, 2016). La Figure 1 présente quelques classes d'objets pouvant être détectées sur l'imagerie aérienne et satellitaire. Sur ces images, les instances d'objets sont localisées via une BB verte.

Figure 1 : Exemples d'objets d'intérêt sur les images aériennes et satellitaires. (Tiré de Li et coll., 2020)

L'extraction de l'information sur les images de télédétection est effectuée traditionnellement par photo-interprétation. Au cours des dernières années, beaucoup d'efforts ont été déployés pour tenter d'automatiser ce processus et les performances se rapprochent de plus en plus de celles de l'humain depuis l'avènement de l'apprentissage profond.

Selon une revue sur les méthodes de détection d'objets sur l'imagerie aérienne et satellitaire effectuée par Cheng et Han (2016), ces méthodes peuvent être séparées en quatre catégories : 1) les méthodes basées sur l'appariement; 2) les méthodes basées sur la connaissance; 3) les approches par classification orientée-objet; et 4) les méthodes basées sur l'apprentissage machine. Récemment, les CNNs ont complètement éclipsé les performances des autres méthodes et ont conduit à un changement de paradigme dans le domaine de la détection d'objets : l'extraction des propriétés des images se fait désormais par apprentissage automatique (à l'aide de bancs de filtres) plutôt que par réingénierie humaine de ces propriétés (couleurs, formes, textures, patrons, etc.). Les méthodes basées sur les

CNNs sont désormais incontournables (Li et coll., 2020). La section 2.2 présente les principales architectures de détections d'objets basées sur l'apprentissage profond.

2.2. Détection d'objets par apprentissage profond

2.2.1. Architectures

Dans leur revue sur l'utilisation de l'apprentissage profond pour la détection d'objets, Liu et coll. (2018) séparent les architectures de détection d'objets en deux catégories : 1) les architectures en deux stades, incluant une étape de prétraitement pour la proposition de régions d'intérêt (RoIs, *Regions of Interest*); et 2) les architectures en un stade, où les pixels de l'image brute sont directement convertis en BBs et en probabilités de classe. Les architectures en un stade n'atteignent pas des performances aussi élevées que celles en deux stades, mais elles sont plus rapides à exécuter et donc plus appropriées pour la détection en temps réel (Huang et coll., 2017). Dans le cadre du présent projet, la précision du modèle est plus importante que la rapidité. Pour cette raison, seule une revue des architectures en deux stades est effectuée.

Dans les architectures de détection d'objets en deux stades, le processus de détection d'objets peut être divisé en deux étapes : 1) la proposition de RoIs; et 2) la classification et la régression de ces RoIs vers des BBs réelles. Le but de la première étape est de proposer un ensemble de boites rectangulaires (RoIs) indépendantes de toutes classes pouvant potentiellement contenir une instance d'objets appartenant à une des classes d'intérêt. Un CNN extrait ensuite les caractéristiques de ces RoIs pour finalement les envoyer à un classificateur qui tente de leur attribuer une classe d'appartenance et d'affiner leurs délimitations. Parmi le grand nombre d'architectures de détection d'objets proposées au cours des dernières années (Sermanet et coll., 2013; Girshick, 2015; Ren et coll., 2015; Gidaris et Komodakis, 2015; Dai et coll., 2016; Liu et coll., 2016; Redmon et coll., 2016), les détecteurs en deux stades ont reçu une attention particulière en raison de leur précision. La suite de cette section présente une brève revue des principales architectures en deux stades.

R-CNN : Inspirés par les résultats retentissants obtenus par les CNNs sur des tâches de classification et par le succès de l'algorithme de proposition de RoIs *selective search* (Uijlings et coll., 2013), Girshick et coll. (2014) ont été parmi les premiers à utiliser les CNNs pour la détection d'objets avec leur architecture R-CNN. Leur méthode extrait d'abord des RoIs de l'image d'entrée à l'aide de l'algorithme *selective search* (Uijlings et coll., 2013). Ces RoIs sont ensuite déformées et recadrées à une résolution fixe pour être envoyées dans un réseau AlexNet (Kryzevsky et coll., 2012) servant à

l'extraction de caractéristiques profondes. Les caractéristiques extraites sont ensuite envoyées dans des *Support Vector Machines* (SVMs) pré-entraînés spécifiques à chaque classe pour effectuer la classification. Finalement, un régresseur de BB (*Bounding box regressor*, BBR) est entraîné pour chaque classe d'objets afin d'affiner les coordonnées des BBs finales. En 2014, R-CNN obtient une précision moyenne de 53,3 % dans le compétition PASCAL VOC 2012 (Everingham et coll., 2010), ce qui correspond à une amélioration de plus de 30 % par rapport au meilleur résultat précédent, DPM HSC (Ren et Ramanan, 2013). Malgré cette amélioration significative par rapport au meilleur résultat précédent, R-CNN présente tout de même d'importants inconvénients (Liu et coll., 2018). L'apprentissage s'effectue en plusieurs étapes, ce qui rend le réseau long et difficile à entraîner puisque chaque partie doit être optimisée séparément. De plus, l'entrainement des SVMs et des BBRs est exigeant en termes de stockage sur le disque dur et de temps. Finalement, le réseau est très lent durant les tests, et ce en raison de l'extraction de caractéristiques par un CNN pour chaque RoI proposée.

SPP-Net : Le principal goulot d'étranglement de R-CNN est l'extraction des caractéristiques profondes, qui doit se faire pour chaque région proposée par l'algorithme de proposition de RoIs. Pour pallier cet inconvénient, He et coll. (2015) introduisent la couche de *spatial pyramid pooling* (SPP). Leur réseau, SPP-Net, utilise cette nouvelle couche pour projeter des régions de différentes tailles vers des vecteurs de caractéristiques de tailles fixes en utilisant les caractéristiques profondes extraites par la 5^{e} couche de convolution de leur encodeur. Cette nouvelle couche permet de réutiliser les caractéristiques préalablement extraites et rend SPP-Net nettement plus rapide que R-CNN, sans sacrifier la précision.

Fast R-CNN : Bien que SPP-Net soit considérablement plus efficace que R-CNN, il reste plusieurs inconvénients notables (Zhao et coll, 2019). Les couches de convolution précédant la couche SPP ne peuvent être mises à jour avec l'algorithme de *fine-tuning* (He et coll, 2015). Il en résulte une chute de précision lors de l'utilisation de réseaux très profonds. Afin de régler ce problème, Girshick (2015) propose une nouvelle architecture qu'il nomme Fast R-CNN. Dans Fast R-CNN, l'entraînement de toutes les couches du réseau peut être effectué en une seule étape (outre la proposition de RoIs), et ce grâce à la nouvelle fonction de perte multitâche proposée qui entraîne simultanément un classificateur et un BBR. Cela évite d'entraîner un classificateur, des SVMs et des BBRs en trois étapes distinctes comme dans R-CNN et SPP-Net. Cette solution réduit non seulement l'espace de stockage, mais améliore aussi la précision et l'efficacité à l'aide de schémas d'entraînement plus raisonnables (Zhao

et coll., 2019). Comparé à R-CNN et SPP-Net, Fast R-CNN est généralement 3 fois plus rapide à l'entraînement et 10 fois plus rapide en inférence (Liu et coll., 2018).

Faster R-CNN : Fast R-CNN a considérablement accéléré le processus de détection et d'apprentissage des architectures en deux stades, mais l'architecture repose toujours sur un algorithme de proposition de RoIs externes, tels que *selective search* (Uijlings et coll., 2012) ou *Edgebox* (Zitnick et Dollár, 2014). Le coût de calcul associé à cette étape devient le nouveau goulot d'étranglement. Pendant ce temps, des travaux soulignent les capacités remarquables des CNNs pour localiser des objets à l'aide des *feature maps* (Zhou et coll., 2014; Oquab et coll., 2015; Zhou et coll., 2016). Basé sur ces observations, Ren et coll. (2015) propose le *Region Proposal Network* (RPN), un générateur de RoIs efficace et précis basé sur les capacités des CNNs. Leur architecture, Faster R-CNN, est constituée de leur RPN pour la proposition de RoIs et de Fast R-CNN pour la classification de celles-ci, le tout compris dans un seul réseau. Le RPN et Fast R-CNN partagent plusieurs couches de convolution, permettant ainsi de réduire le coût de calcul. Puisque le RPN est en fait un réseau entièrement convolutif (FCN, *Fully Convolutional Network*), Faster R-CNN devient la première architecture de détection d'objets entièrement basé sur des CNNs. Faster R-CNN et son RPN est à la base des approches ayant remporté la première place aux compétitions de détection d'objets *ImageNet Large Scale Visual Recognition Competition* (ILSVRC) 2015 (He et coll., 2016), *Common Objects in Context* (COCO) 2015 (He et coll., 2016) et COCO 2017 (Peng et coll., 2018). L'architecture sera ensuite utilisée comme fondation dans de nombreux travaux ultérieurs (Gidaris et Komodakis, 2015; Liu et coll., 2016; Dai et coll., 2016). Faster R-CNN a également montré sa flexibilité en étant étendu à d'autres tâches comme la segmentation d'instances (He et coll., 2017). D'autres travaux, comme RFCN (Dai et coll., 2016) et RetinaNet (Lin et coll., 2017), ont essayé de bonifier l'architecture de Faster R-CNN, mais ces architectures atteignent des précisions comparables à Faster R-CNN, à des temps d'exécution parfois plus rapides (Liu et coll., 2018).

Les architectures présentées se sont toutefois concentrées sur le problème conventionnel de détection d'objets sans considérer la tâche de DA qui est souvent nécessaire lors de la mise en application réelle de ces architectures. Dans le cadre de ce projet, Faster R-CNN est choisi comme détecteur de base, mais la méthode proposée a également pour but d'améliorer les capacités de généralisation de l'architecture dans un domaine cible à l'aide d'approches de DA. La section 2.3 fait la revue des approches existantes dans ce domaine. Nous allons d'abord présenter (Section 2.2.2) les métriques utilisées dans le domaine de la détection d'objets.

2.2.2. Métriques

Il existe trois grandes compétitions de détection d'objets sur les images génériques : PASCAL VOC, MSCOCO et Open Images (Everingham et coll., 2010; Lin et coll., 2014; Kuznetsova et coll., 2018). Dans chacune de celles-ci, la métrique utilisée pour évaluer les différents modèles est le *mean Average Precision* (mAP), soit la moyenne du *Average Precision* (AP) sur l'ensemble des classes. Il existe toutefois différentes variantes de cette métrique dépendamment du seuil d'*Intersection over Union* utilisé. Le calcul de cette métrique nécessite la compréhension de plusieurs concepts importants. Ces concepts seront passés en revue succinctement dans les prochaines sections avant de montrer un exemple de calcul de l'AP.

Intersection over Union (IoU)

L'*Intersection over Union* (IoU) est une métrique visant à évaluer le niveau de superposition entre deux BBs. Son calcul nécessite une boîte correspondant à la vérité terrain (ce que l'on appelle également une annotation) et une boîte prédite par le modèle (ce que l'on appelle également une détection). Le IoU est donné par l'aire de superposition des deux boîtes divisée par l'aire d'union, tel qu'illustré sur la Figure 13.

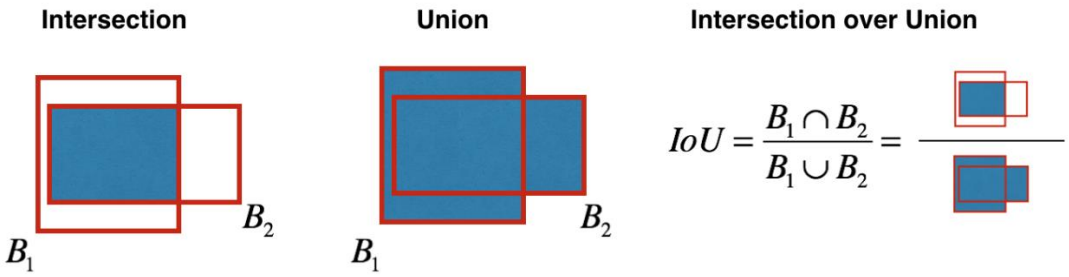


Figure 2 : Illustration graphique et mathématique du calcul de l'I*ntersection over Union* (Tiré de https://www.oreilly.com/library/view/hands-on-convolutional-neural/9781789130331/assets/8a083f5a-2925-4206-abc5-7cdfc4a3ba0b.png)

Vrai Positif, Faux Positif, Faux Négatif et Vrai Négatif

Le deuxième concept est celui de Vrai Positif (VP ou TP, *true positive*), Faux Positif (FP), Faux Négatif (FN) et Vrai Négatif (VN ou TN, *true negative*).

Vrai Positif (VP) : une bonne détection, soit une détection avec IoU supérieur au seuil.

Faux Positif (FP) : une mauvaise détection, soit une détection avec IoU inférieur au seuil.

Faux Négatif (FN) : une vérité terrain non détectée.

Vrai Négatif (VN) : ne s'applique pas. Cela représenterait une boîte où il n'y a pas d'objet et qui n'est pas détecté par le modèle (ce que l'on veut qu'il arrive).

Le seuil de IoU utilisé dans les compétitions est généralement de 50 %, 75 % ou 95 %. Il existe également d'autres variantes plus complexes.

Précision

La précision est une métrique exprimant la capacité d'un modèle à trouver seulement les objets d'intérêt. Le calcul est le suivant :

$$Précision = \frac{VP}{VP + FP} = \frac{VP}{toutes\ les\ détections}$$

Rappel

Le rappel est une métrique exprimant la capacité d'un modèle à trouver l'entièreté des objets d'intérêt. Le calcul est le suivant :

$$Rappel = \frac{VP}{VP + FN} = \frac{VP}{toutes\ les\ vérités\ terrain}$$

***Average Precision* (AP)**

Le AP correspond à l'aire sous la courbe de la courbe de Précision x Rappel (*Precision x Recall curve*). La meilleure façon de comprendre cette courbe et le calcul de l'AP est à l'aide d'un exemple. L'exemple illustré suivant est tiré de la page GitHub de Rafael Padilla[1].

Si l'on considère les détections suivantes :

[1] https://github.com/rafaelpadilla/Object-Detection-Metrics

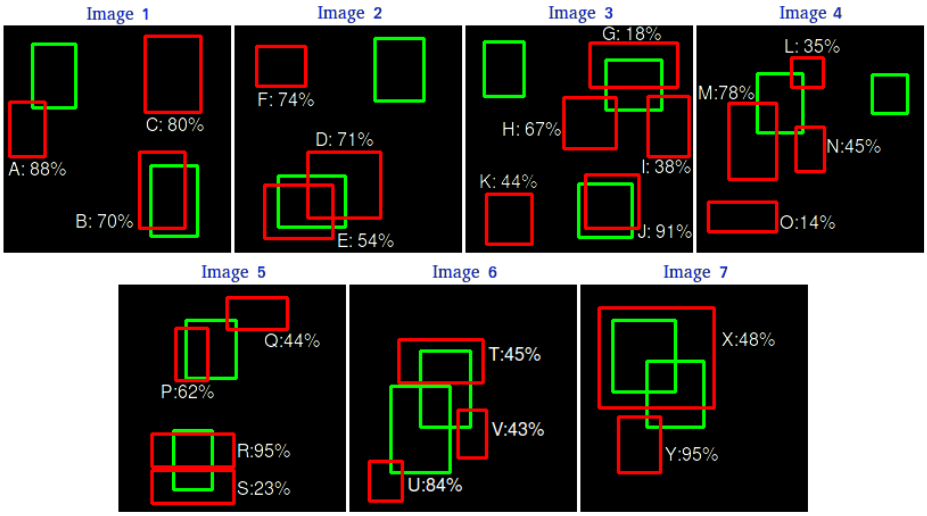


Figure 3 : Exemples d'images avec des vérités terrains (boîtes vertes) et des détections provenant d'un modèle (boîtes rouges) (Tiré de https://github.com/rafaelpadilla/Object-Detection-Metrics)

On retrouve 7 images contenant 15 vérités terrains (boîtes vertes) et 24 détections provenant d'un modèle (boîtes rouges). Chaque détection possède un niveau de confiance et est identifiée par une lettre (A, B,…, Y). En choisissant un seuil de IoU, on peut déterminer si une détection est un VP ou un FP. Dans cet exemple, le seuil de IoU choisi pour considérer une détection comme un VP est de 30 %. Lorsque plusieurs détections superposent la même vérité terrain (comme c'est le cas dans plusieurs images), seule la détection avec le plus grand IoU est considérée comme un VP. Ainsi, 5 VP d'un même objet comptent comme 1 VP et 4 FP.

Pour faire la courbe de Précision x Rappel, il faut d'abord calculer la précision et le rappel de manière cumulative sur l'ensemble des détections. Pour ce faire, il faut commencer par ordonner les détections par leur niveau de confiance. La précision et le rappel cumulatif sont ensuite calculés pour l'entièreté des détections comme le montre le tableau suivant.

Tableau 1 : Détections ordonnées par niveau de confiance (*confidences*) avec calcul de la précision cumulative et du rappel cumulatif (Tiré de https://github.com/rafaelpadilla/Object-Detection-Metrics)

Images	Detections	Confidences	TP	FP	Acc TP	Acc FP	Precision	Recall
Image 5	R	95%	1	0	1	0	1	0.0666
Image 7	Y	95%	0	1	1	1	0.5	0.0666
Image 3	J	91%	1	0	2	1	0.6666	0.1333
Image 1	A	88%	0	1	2	2	0.5	0.1333
Image 6	U	84%	0	1	2	3	0.4	0.1333
Image 1	C	80%	0	1	2	4	0.3333	0.1333
Image 4	M	78%	0	1	2	5	0.2857	0.1333
Image 2	F	74%	0	1	2	6	0.25	0.1333
Image 2	D	71%	0	1	2	7	0.2222	0.1333
Image 1	B	70%	1	0	3	7	0.3	0.2
Image 3	H	67%	0	1	3	8	0.2727	0.2
Image 5	P	62%	1	0	4	8	0.3333	0.2666
Image 2	E	54%	1	0	5	8	0.3846	0.3333
Image 7	X	48%	1	0	6	8	0.4285	0.4
Image 4	N	45%	0	1	6	9	0.4	0.4
Image 6	T	45%	0	1	6	10	0.375	0.4
Image 3	K	44%	0	1	6	11	0.3529	0.4
Image 5	Q	44%	0	1	6	12	0.3333	0.4
Image 6	V	43%	0	1	6	13	0.3157	0.4
Image 3	I	38%	0	1	6	14	0.3	0.4
Image 4	L	35%	0	1	6	15	0.2857	0.4
Image 5	S	23%	0	1	6	16	0.2727	0.4
Image 3	G	18%	1	0	7	16	0.3043	0.4666
Image 4	O	14%	0	1	7	17	0.2916	0.4666

En faisant le graphique de la précision cumulative en fonction du rappel cumulatif, un trouve la courbe de Précision x Rappel suivante :

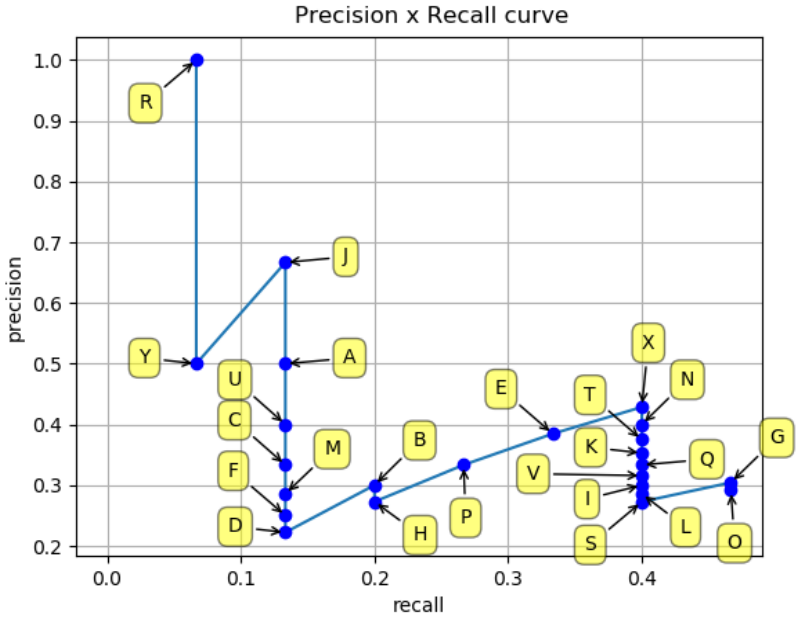


Figure 4 : Courbe de Précision x Rappel. (Tiré de https://github.com/rafaelpadilla/Object-Detection-Metrics)

À partir de cette courbe, il existe deux méthodes pour calculer l'AP : l'interpolation à l'aide de 11 points équidistants ou l'interpolation à l'aide de tous les points. Depuis 2010, la méthode utilisée par les compétitions de détection d'objets est l'interpolation de tous les points, alors cette méthode a été utilisée dans le présent projet. En interpolant tous les points, on trouve la courbe suivante :

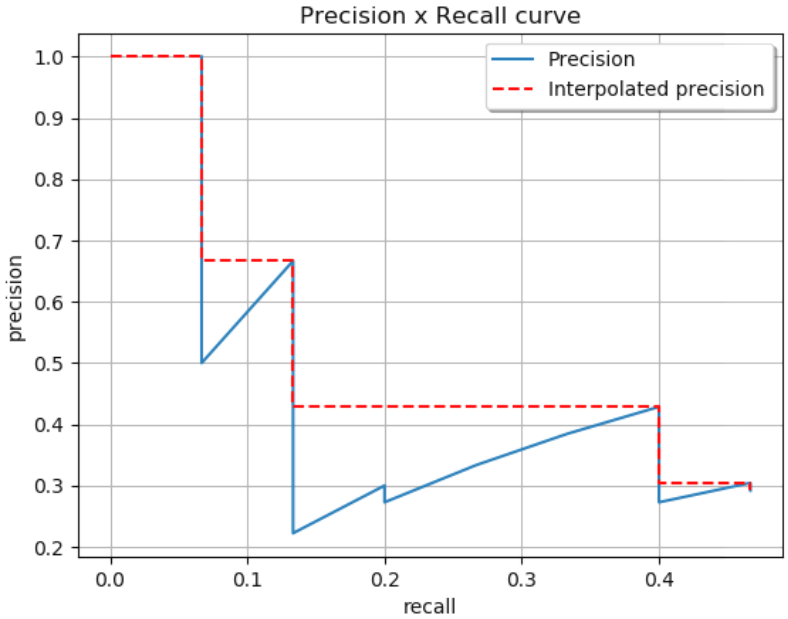


Figure 5 : Courbe de Précision x Rappel avec précision interpolée. (Tiré de https://github.com/rafaelpadilla/Object-Detection-Metrics)

À partir de ce graphique, il est possible de séparer l'aire sous la courbe en 4 aires (A1, A2, A3 et A4).

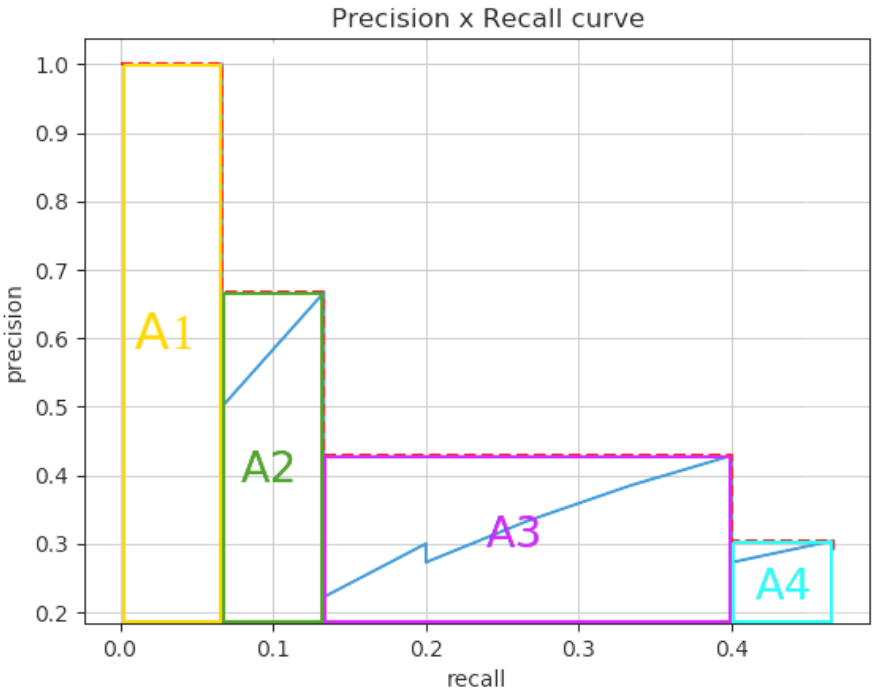


Figure 6 : Courbe de Précision x Rappel avec l'aire sous la courbe séparée en 4 polygones. (Tiré de https://github.com/rafaelpadilla/Object-Detection-Metrics)

L'AP correspond à l'aire totale sous la courbe de précision interpolée. Ainsi, pour calculer l'AP, on trouve :

$$AP = A1 + A2 + A3 + A4$$

Avec :

$A1 = (0{,}0666 - 0) \cdot 1 = \mathbf{0,0666}$

$A2 = (0{,}1333 - 0{,}0666) \cdot 0.6666 = \mathbf{0,04446222}$

$A3 = (0{,}4 - 0{,}1333) \cdot 0{,}4285 = \mathbf{0,11428095}$

$A4 = (0{,}4666 - 0{,}4) \cdot 0{,}3043 = \mathbf{0,02026638}$

On trouve :

$AP = \ 0{,}0666 + \ 0{,}04446222 + 0{,}11428095 + 0{,}02026638$

$AP = 0{,}24560955$

$AP = \mathbf{24,56}\ \%$

2.3. Adaptation de domaine

Des volumes importants de données non annotées sont disponibles dans de nombreux domaines, mais l'annotation de ces données demeure une tâche exigeante en termes de temps et d'argent. Pour s'éviter

le processus fastidieux d'annotation, des solutions alternatives ont été proposées dans la littérature afin d'exploiter les données non étiquetées disponibles (apprentissage non supervisé) ou les données et/ou modèles disponibles dans des domaines similaires (apprentissage par transfert) (Csurka, 2017). L'adaptation de domaine (DA, *domain adaptation*) est un cas particulier d'apprentissage par transfert (*transfer learning*) visant à exploiter les données étiquetées dans un ou plusieurs domaines sources pour apprendre à un modèle à performer dans un domaine cible connexe (Wang et Deng, 2018). Le(s) domaine(s) source(s) sont typiquement différents du domaine ciblé par l'application, mais la tâche reste similaire (Par ex. classification de caractères). Si les domaines sont identiques, on parle d'un problème d'apprentissage machine standard où les données d'entraînement et de test proviennent de la même distribution. Lorsque les distributions (caractéristiques) des ensembles d'entraînement et de test sont différentes, les performances sur le jeu de données de test se voient généralement dégradées et il devient nécessaire d'utiliser le DA. La Figure 7 montre un exemple de trois jeux de données souvent utilisés dans la littérature pour tester les approches de DA.

Figure 7 : Trois jeux de données (MNIST, USPS et SVHN) fréquemment utilisés dans la littérature pour tester les approches de DA. Ces jeux de données se concentrent sur la même tâche (classification de caractères), mais possèdent des caractéristiques différentes. (Tiré de Tzeng et coll., 2017)

Dans les applications de vision par ordinateur, cette différence de distribution, aussi appelée décalage de domaine (*domain shift*), est courante dans les applications réelles. Ce décalage est généralement la conséquence de conditions changeantes, telles que l'arrière-plan, l'emplacement, le point de vue, les conditions d'éclairage et autres. En télédétection, ce décalage peut être dû à la variabilité statistique des caractéristiques des objets (couleur, forme, texture) d'une région géographique à l'autre. Les caractéristiques des objets diffèrent également en fonction du capteur utilisé et des conditions d'acquisition (résolutions spatiale, spectrale et radiométrique, angle de visée, angle solaire, etc.). La

Figure 8 présente une comparaison d'imagettes provenant de quelques jeux de données d'apprentissage profond d'imagerie aérienne et satellitaire, soit xView (Lam et coll., 2018), *Cars Overhead With Context* (COWC) (Mundhenk et coll., 2016), SpaceNet (Van Etten et coll., 2018) et notre jeu de données GeoImageNet (Bouroubi et coll., 2019). On y constate le décalage existant entre les divers jeux de données.

Figure 8 : Imagettes de xView, COWC, SpaceNet et GeoImageNet respectivement. On y voit des différences de points de vue, d'angle solaire et de conditions d'éclairement. Les imagettes ont aussi différentes résolutions spatiales : xView (30 cm), COWC (15 cm), SpaceNet (30 et 50 cm), GeoImageNet (50 cm) (Adapté de Lam et coll., 2018)

L'adaptation de domaine a été largement étudiée pour la tâche de classification (Wang et Deng, 2018). Les méthodes proposées essaient généralement de réduire la variabilité statistique entre le domaine source et le domaine cible. Certaines méthodes récentes de DA utilisent des transformations neuronales profondes permettant de représenter les deux domaines dans un même espace de caractéristiques. Ceci est généralement réalisé en optimisant les représentations profondes (*embeddings*) extraites par l'encodeur afin de minimiser une certaine mesure exprimant la différence entre les deux domaines, tel que la différence moyenne maximale (*maximum mean discrepancy*) (Tzeng et coll., 2014; Long et coll., 2015) ou la distance de corrélation (Sun et coll., 2016; Sun et Saenko, 2016).

Récemment, de nouvelles alternatives ont proposé de reconstruire le domaine cible à partir de représentations du domaine source (Ghifary et coll., 2016). Les méthodes d'adaptation adverserielle sont devenues une incarnation de plus en plus populaire de cette approche et ont démontré des résultats très prometteurs (Tzeng et coll., 2017). Ces méthodes utilisent un processus d'apprentissage similaire à celui des réseaux adverseriels génératifs (GANs, *generative adversarial networks*) (Goodfellow et coll., 2014). Les GANs sont apparus dans le domaine de l'apprentissage machine en 2014 et sont rapidement devenus les « chouchous ». Yann LeCun, directeur du département d'intelligence artificielle pour Facebook, mentionne la chose suivante : "*GANs are the most interesting idea in the last 10 years in machine learning*" (Gui et coll., 2020). Gui et coll. (2020) mentionnent également

dans leur revue sur les GANs que 11 800 papiers reliés au GANs ont été publiés en 2018 selon Google scholar, soit 32 papiers par jour, ce qui en dit long sur l'engouement que cette nouvelle architecture a suscité dans le milieu de l'apprentissage machine.

Les GANs sont des modèles génératifs profonds qui opposent deux modèles : un modèle génératif (G) et un discriminateur (D). Dans cette configuration, G est entraîné à produire de nouvelles images ressemblant en tout point à de vraies images de façon à tromper D. Pour sa part, D, en voyant une image, tente de distinguer s'il s'agit d'une vraie image ou d'une fausse image générée par G, et ce en prédisant une étiquette binaire. Les réseaux sont entraînés conjointement en utilisant la rétropropagation de la perte du discriminateur de façon minimax, en changeant simultanément les poids de G pour tromper D et ceux de D pour qu'il évite de se tromper. Dans ce type de réseau, G et D agissent à titre d'adversaire, car G souhaite que le taux d'erreur de D soit élevé (qu'il pense souvent que les images générées sont vraies), tandis que D souhaite que son taux d'erreur soit bas (qu'il différencie avec succès les vraies et les fausses images). En DA, ce principe est utilisé pour entraîner un discriminateur à ne plus être capable de différencier de quel domaine provient les exemples observés (Ganin et Lempitsky, 2014; Liu et Tuzel, 2016). Toutefois, tel que mentionné par Tzeng et coll. (2017), chaque algorithme fait différents choix de conception en ce qui a trait à l'utilisation ou non d'un générateur, à la fonction de perte utilisée et au partage (ou non) des poids entre les encodeurs des différents domaines.

Dans leurs recherches, Tzeng et coll. (2017) observent que, en DA, l'apprentissage de représentations asymétriques permet de mieux modéliser les différences dans les caractéristiques de bas niveau que l'apprentissage symétrique. Dans la veine de ces observations, les auteurs proposent une nouvelle méthode d'adaptation adverserielle non supervisée qu'ils nomment *Adversarial Discriminative Domain Adaptation* (ADDA). ADDA utilise un processus séquentiel d'apprentissage tel qu'illustré sur la Figure 9. D'abord, un modèle de classification (encodeur + classificateur) est entraîné à l'aide des données d'entraînement du domaine source. Ensuite, durant l'adaptation adverserielle, un encodeur « cible » apprend à extraire, sur le domaine cible, des représentations que le discriminateur n'arrive pas à dissocier des représentations extraites par l'encodeur « source » sur les données du domaine source, le tout grâce à un objectif adverseriel. Leur approche atteint des résultats similaires ou meilleurs que les autres approches sur des tâches de DA (Wang et Deng, 2018) entre les chiffres des jeux de données MNIST (LeCun et coll., 1998a), USPS (Hull, 1994) et SVHN (Netzer et coll., 2011).

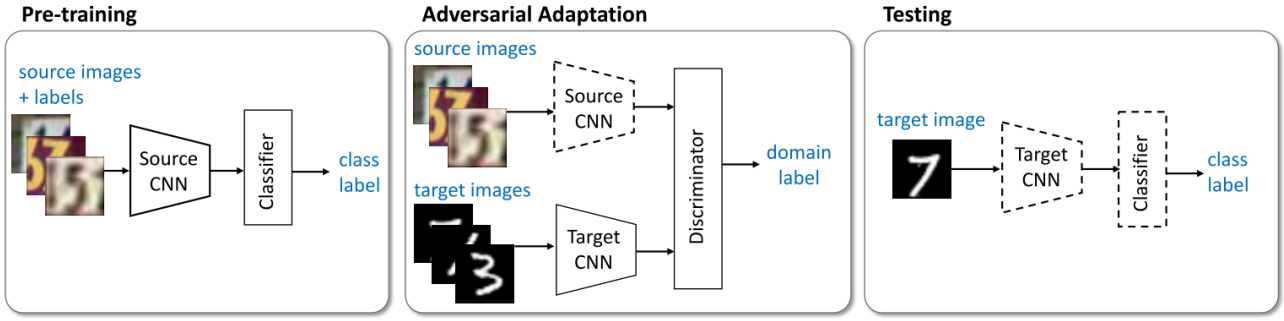


Figure 9 : Séquence d'entraînement de ADDA (Tiré de Tzeng et coll., 2017)

L'approche initiale ADDA est toutefois basée sur un processus d'apprentissage non supervisé, où aucune annotation n'est disponible dans le domaine cible. En télédétection, il est souvent possible d'obtenir de petites quantités d'annotations dans le domaine cible. Dans cette optique, Wang et coll. (2018) proposent une modification à l'architecture initiale afin de permettre l'utilisation d'annotations provenant du domaine cible lors de l'entraînement. Ils nomment cette modification semi-ADDA (*semisupervised* ADDA). ADDA permet d'assurer la concordance statistique entre les représentations extraites par les divers encodeurs (source et cible), mais ne peut garantir que les représentations apprises sont discriminantes, c'est-à-dire que les représentations des diverses classes sont séparables (Wang et coll., 2018). Dans le cas de semi-ADDA, le modèle est conçu de façon à tirer profit des annotations disponibles dans le domaine cible, et ce afin d'éviter ce problème. Leur méthode a été testée sur des images satellitaires RGB pour la classification de panneaux solaires entre deux jeux de données provenant de deux villes californiennes distinctes (Fresno et Stockton) et obtient de très bon résultats surpassant presque systématiquement ceux du *fine-tuning* (Wang et coll., 2018).

Or, bien que les performances atteintes par ADDA et semi-ADDA soient supérieures à celles des autres méthodes de DA, la tâche visée en est une de classification. Lors de la revue de littérature effectuée dans le cadre de ce projet (automne 2018), très peu de travaux s'étaient concentrés sur l'adaptation de domaine appliquée aux autres tâches de vision par ordinateur. La section suivante présente les approches de DA existantes au-delà de la classification.

2.4. Adaptation de domaine au-delà de la classification

Contrairement à la recherche sur le DA appliqué à la classification, l'application du DA aux autres tâches de vision par ordinateur a fait l'objet de moins d'attention dans la littérature. Au moment de la revue effectuée dans le cadre de ce projet, seuls quelques travaux s'étaient concentrés sur la segmentation sémantique (Hoffman et coll., 2016; Chen et coll., 2018a; Zhang et coll., 2017) et sur la

détection d'objets (Xu et coll., 2014; Raj et coll., 2016; Chen et coll., 2018b). Depuis, d'autres approches ont été proposées spécifiquement pour les autres tâches de vision par ordinateur (Michieli et coll., 2020; Hsu et coll., 2020; Rodriguez et Mikolajczyk, 2019). Or, parmi ces travaux, aucun n'a encore traité l'adaptation de ADDA et semi-ADDA à la détection d'objets. Ce développement fait l'objet du présent projet.

3. Cadre expérimental

3.1. Zones d'études et données

Dans le cadre de ce projet de recherche, l'adaptation de domaine est effectuée entre les jeux de données xView (Lam et coll., 2018) et GeoImageNet (Bouroubi et coll., 2019), où xView est le domaine source et GeoImageNet le domaine cible.

Le jeu de données xView un jeu de détection d'objets ouvert comprenant plus d'un million d'annotations, et ce pour 60 classes d'objets (http://xviewdataset.org/). Ce jeu contient 1 129 scènes prises à plusieurs endroits à travers le monde. Les scènes proviennent des satellites WorldView-3 de DigitalGlobe, possèdent trois bandes RGB et une résolution spatiale de 0,3 m. Les images sont préparées avec l'orthorectification, le *pansharpening* et le réglage de la plage dynamique RGB.

Pour sa part, GeoImageNet (Bouroubi et coll., 2019) est un jeu de données en cours d'élaboration qui contiendra, lorsqu'achevé (et dans sa première version), 48 classes d'occupation du sol et 178 classes d'objets. GeoImageNet comprend 40 scènes issues des satellites Pléiades du Centre national d'études spatiales (CNES). Ces scènes possèdent quatre bandes RGBN (*Blue*, *Green*, *Red*, *Near-infrared*) et une résolution spatiale de 0,5 m, après *pansharpening*. Les images sont préparées en y appliquant une orthorectification et un rehaussement radiométrique 8 bits. La Figure 10 montre l'emplacement géographique des toutes les scènes utilisées dans le cadre de ce projet pour xView et GeoImageNet.

Comme on peut constater en regardant les taxonomies de ces jeux de données (voir Annexe 2), il existe peu de concordances parfaites entre les taxonomies de ces jeux de données. Afin d'assurer une certaine correspondance, une généralisation de plusieurs classes a été effectuée afin de créer deux classes génériques : « avion » et « véhicule ». La classe « avion » est une classe générique qui correspond à un amalgame des annotations de la classe mère « *Fixed-Wing Aircraft* » et des classes filles « *Small Aircraft* » et « *Cargo Plane* » de xView. Pour GeoImageNet, il s'agit d'un amalgame

des classes « Avion de type cesna », « Avion de ligne » et « Avion militaire ». La classe « véhicules » est également une classe générique qui correspond à un amalgame des annotations des classes mères « *Passenger Vehicle* » et « *Truck* » et des classes filles « *Small Car* », « *Bus* », « *Pickup Truck* », « *Utility Truck* », « *Truck* », « *Cargo Truck* », « *Truck Tractor* », « *Truck w/Box* », « *Truck w/Flatbed* » et « *Truck w/Liquid* » de xView. Pour GeoImageNet, il s'agit d'un amalgame des classes « Voiture », « Fourgonette », « Camionette » et « Autocar/Autobus ». La section 3.3 présente plus en détail le processus de préparation et de nettoyage des données, et ce pour le domaine source et cible.

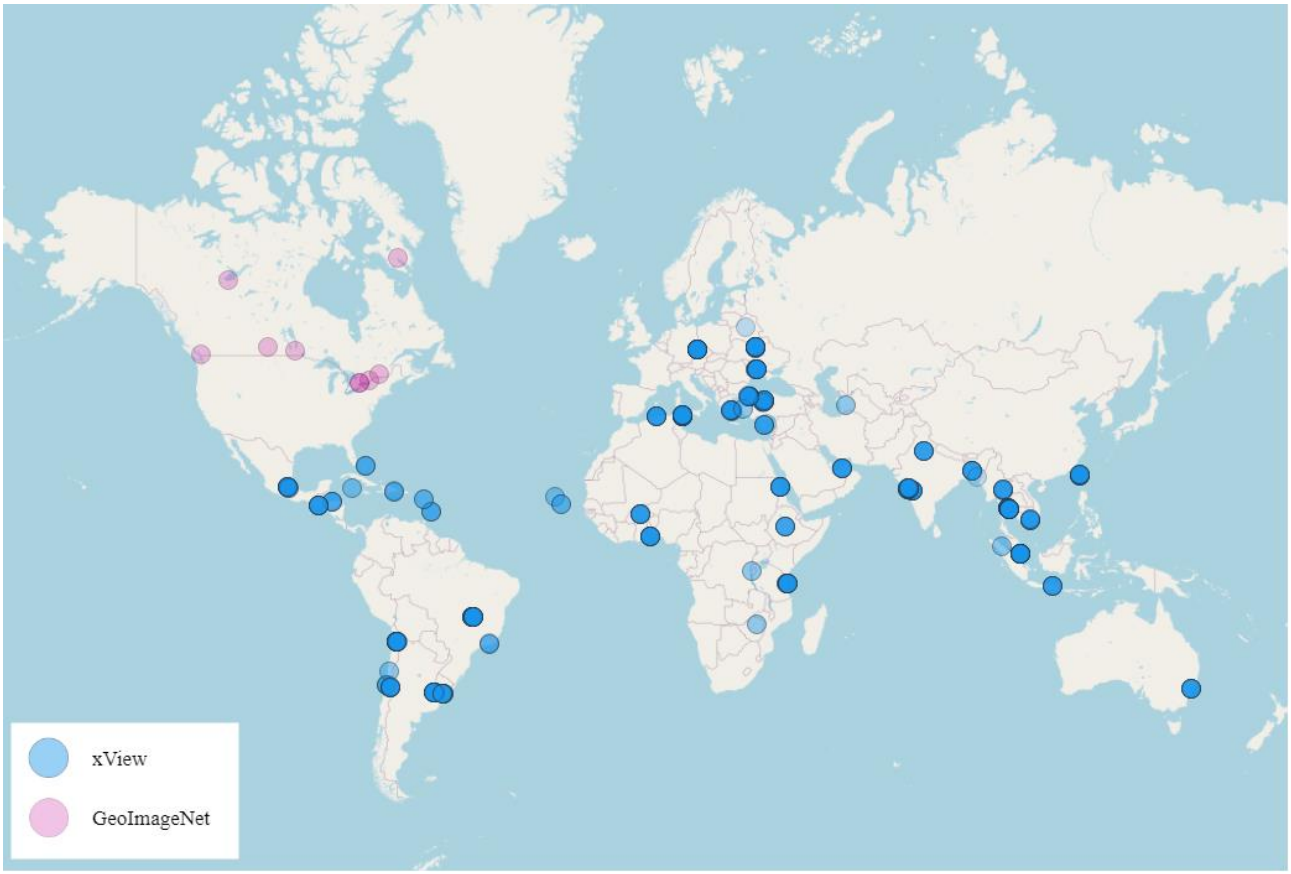


Figure 10 : Emplacement géographique des scènes utilisées pour la génération des imagettes. Chaque image est représentée par un point d'une transparence de 75%, de sorte que les points plus opaques témoignent de la présence de plusieurs images sensiblement au même endroit (Source du fond de carte : OpenStreetMap) (Inspiré de Lam et coll., 2018)

3.2. Démarche méthodologique

La démarche méthodologique du projet comporte sept grandes étapes, et ce dans le but de comparer 4 approches (Figure 11). Ces étapes sont les suivantes :

1) Préparation des données sur chaque domaine;
2) Pré-entraînement du modèle Faster R-CNN sur le domaine source;
3) Évaluation de la première approche sur le domaine cible : aucune adaptation de domaine;
4) Implémentation de la deuxième approche, le *fine-tuning* conventionnel, et évaluation sur le domaine cible.
5) Implémentation de la troisième approche, l'adaptation de domaine avec ADDA, et évaluation sur le domaine cible;
6) Implémentation de la quatrième approche, l'adaptation de domaine avec semi-ADDA, et évaluation sur le domaine cible;
7) Comparaison des résultats des différentes approches.

L'Annexe 3 illustre les processus d'entrainement des 4 approches considérées. Ces approches sont décrites plus en détail dans les sections 4.5 à 4.8.

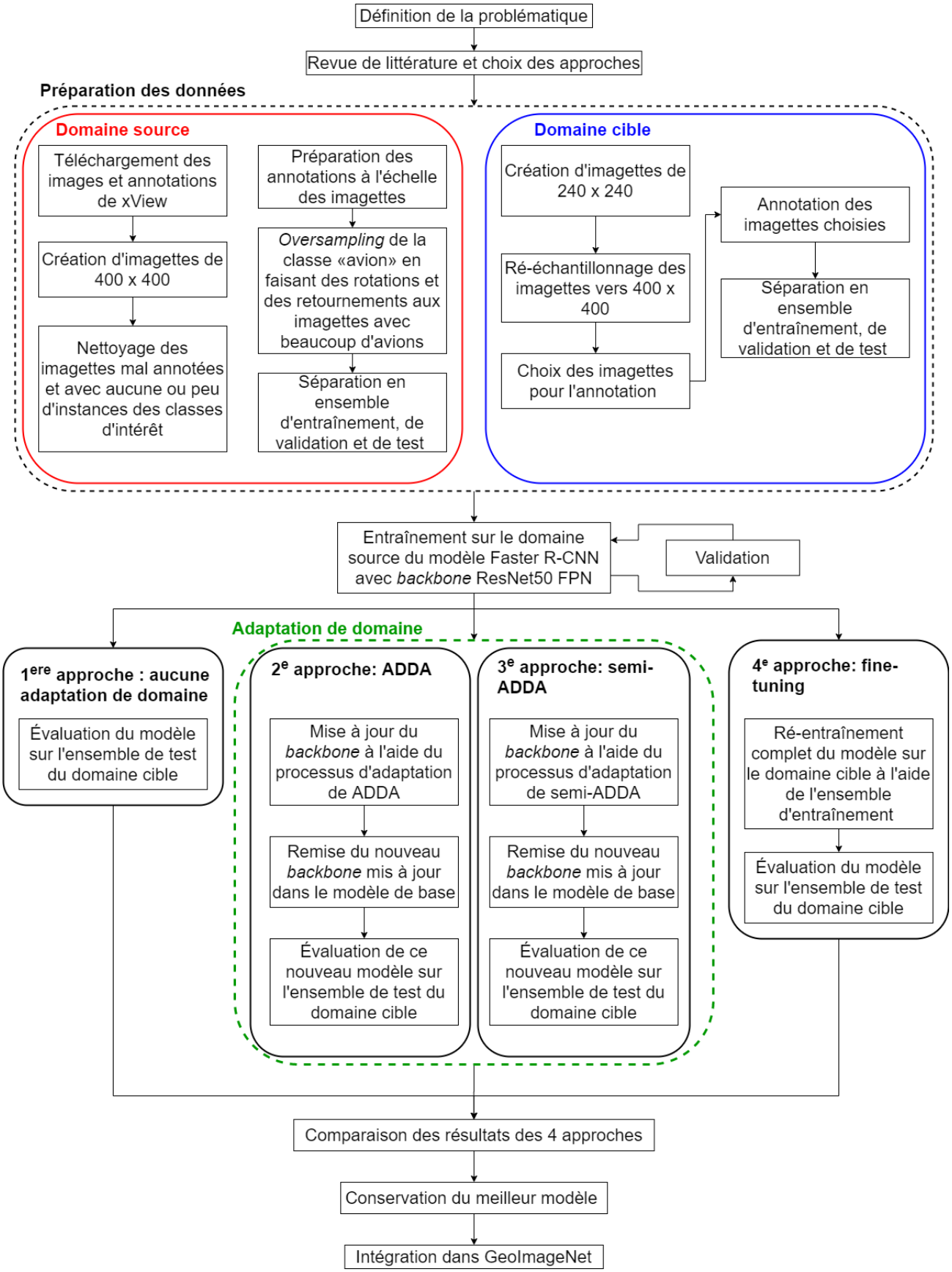


Figure 11 : Organigramme méthodologique du projet

3.3. Préparation et nettoyage des données

3.3.1. Domaine source : xView

Étant un jeu de données ouvert, les données (images et annotations) de xView sont disponibles sur leur site web (http://xviewdataset.org/). Plusieurs étapes de prétraitements sont toutefois nécessaires avant de pouvoir utiliser les données pour l'entraînement de modèles d'apprentissage profond.

D'abord, la première étape consiste à générer des imagettes à partir des scènes. Dans le cadre du présent projet, les entraînements sont effectués sur des imagettes de 400 pixels par 400 pixels, et ce dans le but de pouvoir utiliser un *batch size* de 4 ou plus sans rencontrer de problème en lien avec la mémoire du GPU lors de l'entraînement. Les imagettes ne contenant aucune instance de classes d'intérêt et ayant une trop forte couverture nuageuse ont été supprimées. Ensuite, puisqu'un débalancement important d'occurrence existe entre les deux classes considérées (beaucoup plus de voitures que d'avions), un *oversampling* de la classe « avion » a été effectué en appliquant des rotations et des retournements aux imagettes contenant deux avions ou plus. Un total de 9 810 imagettes est conservé après ces premiers prétraitements.

Des entraînements ont alors été effectués, mais les résultats obtenus étaient surprenamment bas. Après plusieurs tests d'hyperparamètres et beaucoup d'incompréhension, les 9 810 imagettes (et leurs annotations) ont finalement été entièrement passées en revue pour s'assurer de la qualité des annotations. À notre surprise, plusieurs images étaient très mal annotées, contrairement à ce que l'on peut s'attendre d'un jeu de données ouvert (libre accès), ce qui a été très décevant. Parmi ces images mal annotées, plusieurs types d'erreurs ont été observés (instances oubliées, double annotation de la même instance, BB mal délimitée, annotation avec la mauvaise étiquette). On peut d'ailleurs observer plusieurs de ces types d'erreurs sur les Figures 12 et 13. Ceci constitue une leçon importante pour quiconque voulant utiliser un jeu de données ouvert : toujours passer les images (et leurs annotations) en revue avant l'entraînement! Cette étape peut sauver beaucoup de déception et d'incompréhension. La Figure 12 illustre quelques exemples d'imagettes mal annotées.

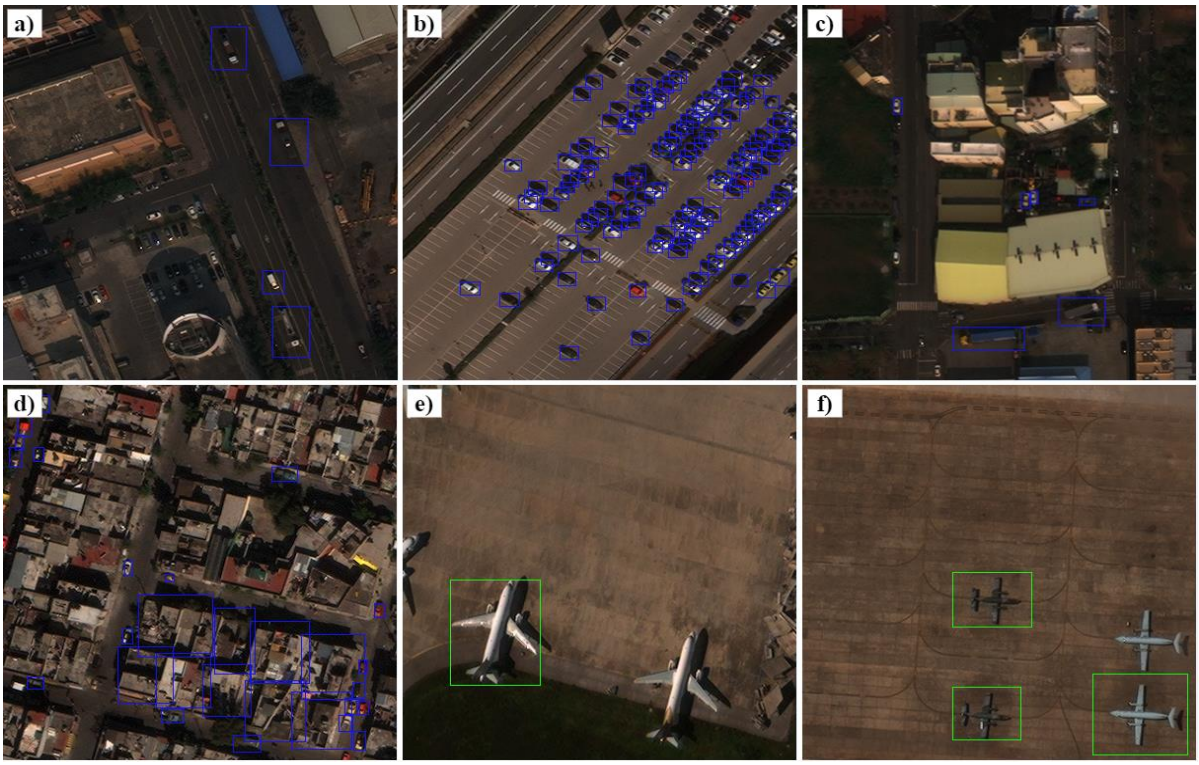


Figure 12 : Quelques exemples d'imagettes mal annotées. En a), b) et c) on voit plusieurs instances de voitures oubliées. En d) on voit des bâtiments étiquetés comme étant des voitures. En e) et f) on voit des instances d'avion oubliés.

Après ce passage en revue des 9 819 imagettes, 2 570 ont été supprimées puisque jugées trop mal annotées pour être conservées. Ceci correspond à environ 26 % des imagettes, ce qui est considérable. En plus de ce nettoyage, un total de 47 imagettes furent réannotées manuellement car ces imagettes contenaient des instances d'avions oubliées ou mal annotées. Les instances d'avions étant tellement rares comparativement aux voitures, il était inconcevable de devoir se départir complètement de ces imagettes. La Figure 13 montre quelques exemples d'imagettes qui ont dû être réannotées. Les imagettes e) et f) de la Figure 12 font également partie des imagettes réannotées.

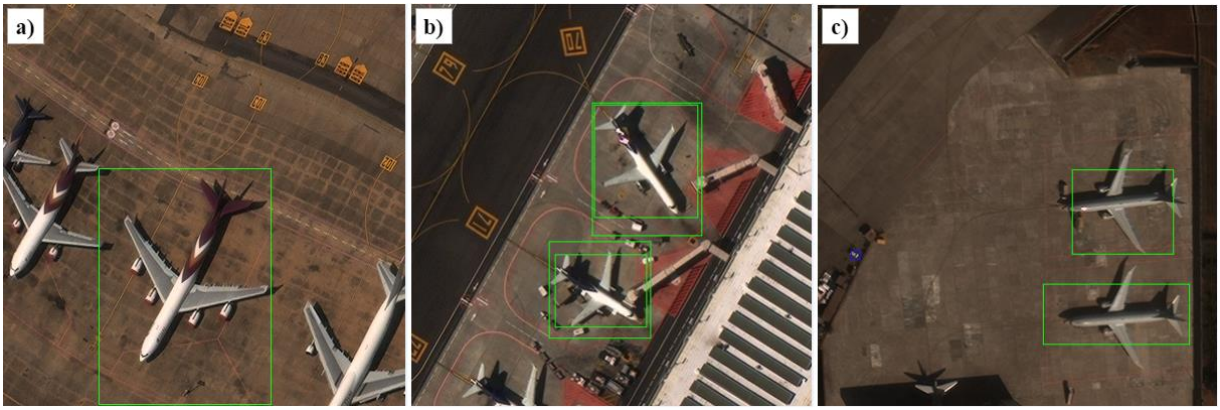


Figure 13 : Exemples d'imagettes mal annotées contenant des avions ayant été réannoté.

Après cet important nettoyage, un total de 7 240 imagettes demeure. Ces imagettes ont ensuite été séparées en ensemble d'entraînement (4 888 imagettes), de validation (1 176 imagettes) et de test (1 176 imagettes). L'ensemble d'entraînement contient 107 724 annotations (1 668 avions et 106 056 véhicules), l'ensemble de validation 23 971 annotations (387 avions et 23 584 véhicules) et l'ensemble de test en contient 25 767 (354 avions et 25 413 véhicules).

Pour la partie non supervisée de ADDA et semi-ADDA, 1 000 imagettes de xView ont été sélectionnées sous condition de contenir au moins une instance des classes d'intérêt. Ces imagettes serviront à apprendre au discriminateur à distinguer le domaine de provenance des représentations extraites par l'encodeur et ne nécessitent donc pas d'annotations. La seule étiquette nécessaire pour ces imagettes est celle de leur domaine de provenance, soit 0 pour GeoImageNet et 1 pour xView (voir sections 3.7 et 3.8).

3.3.2. Domaine cible : GeoImageNet

GeoImageNet est pour sa part un jeu de données en cours de création. Pour les besoins du projet, les instances d'avions et de voitures ont été entièrement annotées sur un total de 187 imagettes. Puisque les scènes de GeoImageNet possèdent une résolution de 0,5 m, les imagettes ont été générées à une taille de 240 pixels par 240 pixels. Celles-ci ont ensuite été rééchantillonnées vers une taille de 400 pixels par 400 pixels, et ce dans le but de générer une fausse résolution de 0,3 . Plusieurs algorithmes de rééchantillonnage ont été testés et celui performant le mieux sur l'ensemble de test a été utilisé pour rééchantillonner l'entièreté des images du domaine cible. Les résultats de ces tests sont présentés dans la section 4.2.

Les 187 imagettes ont été ensuite séparées en ensemble d'entraînement (97 imagettes), de validation (30 imagettes) et de test (60 imagettes). L'ensemble d'entraînement contient 2 937 annotations (85 avions et 2 852 véhicules), l'ensemble de validation 639 (25 avions et 614 véhicules) et l'ensemble de test en contient 1 886 (57 avions et 1 829 véhicules). L'ensemble de test contient davantage d'images que l'ensemble de validation puisque c'est sur cet ensemble que les performances des différentes approches sont évaluées et comparées. L'ensemble d'entraînement est pour sa part utilisé pour effectuer le *fine-tuning* du modèle préalablement entrainé sur le domaine source dans le cadre de la deuxième approche et pour le *fine-tuning* de l'encodeur cible dans la quatrième approche, semi-ADDA.

Pour la partie non supervisée de ADDA et semi-ADDA, 780 imagettes de GeoImageNet ont été sélectionnées sous la même condition que sur le domaine source, soit contenir au moins une instance des classes d'intérêt.

3.4. Pré-entraînement sur le domaine source

Les quatre approches considérées dans le présent projet ont en commun un pré-entraînement du modèle Faster R-CNN sur le domaine source, xView. Cette étape est cruciale puisque les poids trouvés serviront de fondement aux approches considérées. Cette section présente le fonctionnement de Faster R-CNN et de son optimisation.

Faster R-CNN est une architecture de détection d'objets en deux stades composée d'un encodeur et de deux modules supplémentaires. Le premier module est un réseau profond entièrement convolutif (FCN, *fully convolutional network*) appelé *Region Proposal Network* (RPN) ayant pour but de proposer un ensemble de RoIs pouvant potentiellement contenir des objets. Le second est le détecteur Fast R-CNN, celui-ci ayant pour but de classifier les RoIs proposées par le RPN et d'affiner leur position. Ces deux modules sont implémentés conjointement dans un réseau unique de détection d'objets. Le RPN et Fast R-CNN partagent plusieurs couches de convolution de l'encodeur afin de réduire le coût de calcul (Ren et coll., 2015). L'encodeur utilisé dans le cadre de ce projet est le ResNet50 *Feature Pyramid Network* (FPN), et ce puisqu'il s'agit de celui utilisé par la formule gagnante du *COCO-2017 detection challenge* (Peng et coll., 2018). La Figure 14 présente les architectures de Fast R-CNN et Faster R-CNN. La seule différence entre ces deux architectures réside dans l'étape de proposition des RoIs qui est gérée par le RPN dans Faster R-CNN plutôt que par un algorithme externe.

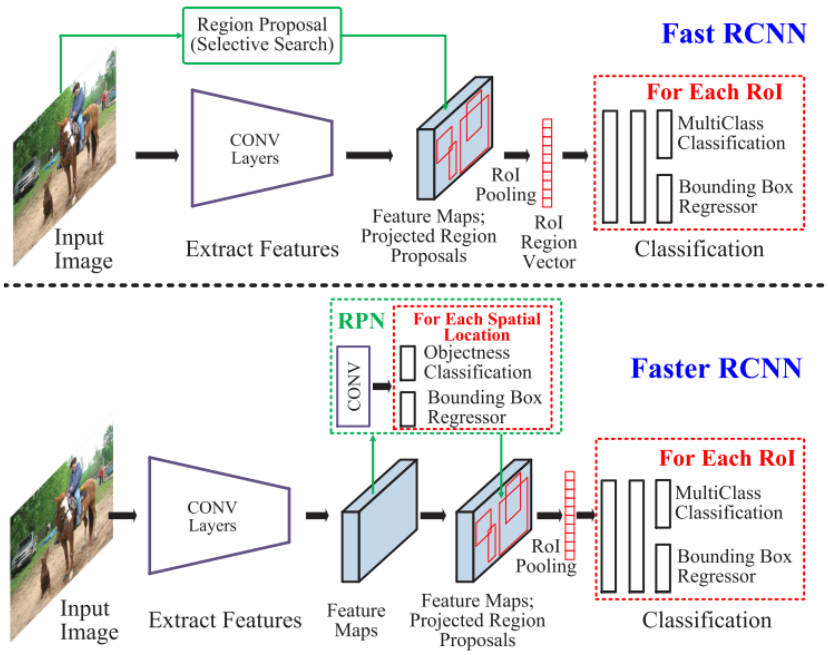


Figure 14 : Représentation graphique des architectures de détection d'objets Fast R-CNN et Faster R-CNN (Tiré de Liu et coll., 2018)

La section 3.4.1 présente le fonctionnement du RPN. La section 3.4.2 présente le fonctionnement de Fast R-CNN. Finalement, la section 3.4.3 présente les détails d'implémentation.

3.4.1. *Region Proposal Network* (RPN)

Le RPN prend en entrée une image (de n'importe quelle taille) et génère en sortie un ensemble de RoIs rectangulaires, chacune avec un score d'*objectness*[2]. Le processus de génération des RoIs est effectué en glissant un petit réseau convolutif sur la *feature map* extraite par la dernière couche de convolution partagée. Ce petit réseau effectue des convolutions de 3 x 3 suivies de deux convolutions 1 x 1 pour la prédiction des scores d'*objectness* (*cls layer*) et pour la régression des BBs (*reg layer*) tel qu'illustré sur la Figure 15. Ces convolutions sont appelées la « tête » du RPN. À chaque position, le RPN génère k boites de références appelées *anchors*. La régression vers des RoIs réelles est effectuée à partir de ces boites de références. Les *anchors* sont centrées à la position de la convolution et possèdent une taille et un ratio de forme qui leur est propre. Dans le papier original, les auteurs utilisent 3 tailles et 3 ratios de forme, ce qui donne k=9 *anchors* à chaque position lors de la convolution.

[2] L'*objectness* est un score entre 0 et 1 mesurant l'appartenance à un ensemble de classe vs. *Background*. En d'autres mots, ce score exprime à quel point le modèle pense qu'un objet se trouve dans ce RoIs.

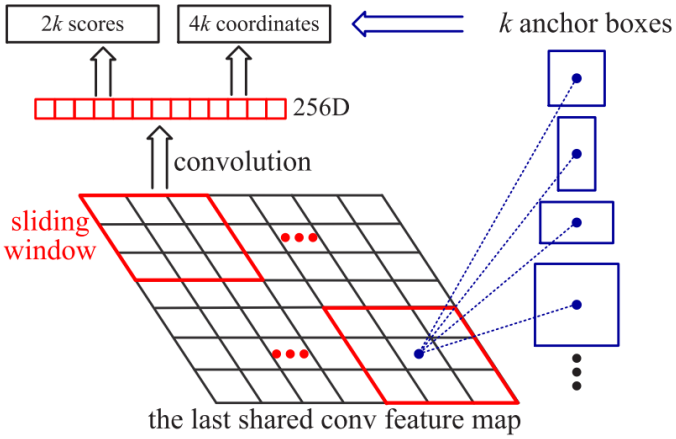


Figure 15 : Illustration du fonctionnement du *Region Proposal Network* (RPN) de Faster R-CNN. (Tiré de Liu et coll., 2018)

Dans le cadre du présent projet, l'encodeur utilisé est un ResNet50 FPN. Ce type d'encodeur, plutôt que de produire une seule représentation, produit une pyramide de représentations à 5 étages, tel qu'illustré sur la Figure 16. Dans cette configuration, la tête du RPN est appliquée à chaque échelle de la pyramide plutôt que d'être appliquée seulement sur la dernière couche de convolution partagée, ce qui signifie que le processus illustré sur la Figure 15 se répète 5 fois à 5 échelles différentes.

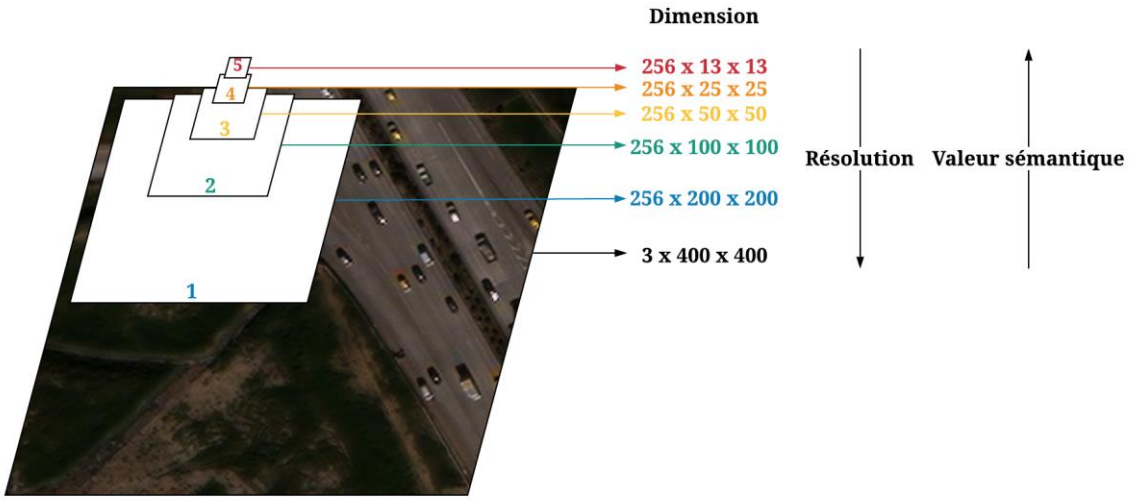


Figure 16 : Illustration de la pyramide de représentations extraites par un ResNet50 FPN sur une image RGB de 400 pixels par 400 pixels. À chaque étage, la résolution diminue au profit de la valeur sémantique. Une dimension de 256 x 13 x 13 signifie 256 *feature maps* de 13 pixels par 13 pixels.

Dans la tête du RPN, la *reg layer* produit en sortie 4*k* paramètres exprimant les coordonnées affinées des *k anchors*, tandis que la *cls layer* produit 2*k* scores *d'objectness*. Lors de l'entraînement du RPN, une étiquette de classe binaire est attribuée à chaque *anchor* visant à exprimer si un objet s'y trouve ou pas. Une étiquette de classe positive (1) est attribuée à deux types d'*anchors* : (i) le ou les *anchor(s)*

avec le plus grand IoU avec une boîte de vérité au sol (*ground truth box*), ou (ii) les *anchors* possédant un IoU supérieur à 0,7 avec n'importe quelle boîte de vérité au sol. Habituellement, la deuxième condition est suffisante pour déterminer les échantillons positifs, mais la première condition est adoptée lorsqu'aucun *anchor* ne respecte la deuxième. Une étiquette de classe négative (0) est donnée à tous les *anchors* possédant un IoU inférieur à 0,3 avec toutes les boîtes de vérité au sol. Les *anchors* qui ne sont ni positifs, ni négatifs ne contribuent tout simplement pas à l'entraînement.

Avec ces définitions, la fonction de perte à minimiser est une fonction multi-tâches similaire à celle de Fast R-CNN (Girshick, 2015). Cette fonction est exprimée comme suit :

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*) \quad (1)$$

Dans cette formulation, i est l'index d'un *anchor* dans une *mini-batch* et p_i est la probabilité prédite que l'*anchor* i soit un objet. L'étiquette exprimant la vérité terrain (p_i^*) est 1 si l'*anchor* est positif, et 0 si l'*anchor* est négatif. t_i est un vecteur contenant les 4 coordonnées affinées de la BB prédite, et t_i^* est un vecteur contenant les 4 coordonnées de la boîte de vérité au sol associé à un *anchor* positif. La fonction de perte de classification (L_{cls}) est une *log loss* classique vers 2 classes (objet vs non-objet). Pour la fonction de perte de régression, $L_{reg}(t_i, t_i^*) = S(t_i - t_i^*)$ est utilisé où S est la *smooth L1 loss* définit dans Fast R-CNN (Girshick, 2015). Le terme $p_i^* L_{reg}$ signifie que la fonction de perte de régression est seulement activée pour les *anchors* positifs ($p_i^* = 1$) et désactivée sinon ($p_i^* = 0$). Les sorties des têtes de classification (*cls layer*) et de régression (*reg layer*) sont $\{p_i\}$ et $\{t_i\}$ respectivement. Les deux termes de cette fonction sont normalisés par N_{cls} et N_{reg} et balancés par le paramètre λ. Le terme de classification est normalisé par la taille de la mini-batch (par ex., $N_{cls} = 256$), tandis que le terme de régression est normalisé par le nombre de position où des *anchors* ont été générés sur la *feature map* (par ex., $N_{reg} \sim 2400$).

3.4.2. Fast R-CNN

Une fois les propositions d'objet (RoIs) générées par le RPN, le détecteur Fast R-CNN est ensuite utilisé pour les classifier. Dans Fast R-CNN, les caractéristiques profondes de chaque proposition d'objets sont reprojetées vers un vecteur de taille fixe grâce à la couche de RoI *pooling*, une couche spéciale de *pooling* permettant de passer d'un tenseur de n'importe quelle dimension vers une dimension fixe. Chaque vecteur de caractéristique est ensuite envoyé dans un *multi layer perceptron* (MLP) qui finit par se diviser en deux « têtes » distinctes : un classificateur et un BBR. Le

classificateur produit, pour chaque RoI, une distribution de probabilité d'appartenance aux $k+1$ classes, soit $p = (p_0, \ldots, p_{k+1})$. Le BBR produit pour sa part un ensemble de 4 *offsets* exprimant le décalage existant entre le RoI et la BB réelle, soit $t^k = (t_x^k, t_y^k, t_w^k, t_h^k)$, et ce pour chaque classe, indexé par k.

Ainsi, comme pour le RPN de Faster R-CNN, Fast R-CNN possède deux couches de sortie séparées, un BBR et un classificateur, mais leur but est différent. Dans le RPN, le but de classification est de produire, pour chaque *anchor*, une probabilité d'appartenance aux classes « objet » et « non-objet » et le but de régression est de peaufiner la position de l'*anchor* pour se rapprocher d'une boîte de vérité au sol à proximité. Dans Fast R-CNN, le but de classification est de produire, pour chaque RoI provenant du RPN, une probabilité d'appartenance aux $k+1$ classes et le but de régression est de peaufiner la position du RoI pour se rapprocher d'une boîte de vérité au sol.

Ainsi, comme pour le RPN, l'entraînement de Fast R-CNN s'effectue avec une fonction de perte multi-tâches, mais quelque peu différente de celle du RPN. Cette fonction est exprimée comme suit :

$$L(p, k, t^k, v) = w_k(L_{cls}(p, k)) + \lambda[k \geq 1]L_{reg}(k, v) \qquad (2)$$

Dans cette formulation, L_{cls} et L_{reg} sont des fonctions de perte visant à faire la classification et la régression respectivement. L_{cls} est une *cross entropy loss* de la prédiction p vers la classe réelle k. Cette fonction est pondérée à l'aide de poids spécifiques à chaque classe, ici dénoté w_k. Le deuxième terme, L_{loc}, est définit par un tuple de vrai régression pour la classe k, $v = (v_x, v_y, v_w, v_h)$, et un tuple de régression prédit $t^k = (t_x^k, t_y^k, t_w^k, t_h^k)$, toujours pour la classe k. Le crochet d'Iverson (*Iverson bracket*), $[k \geq 1]$, donne 1 lorsque $k \geq 1$, sinon 0. Par convention, $k = 0$ pour la classe « arrière-plan ». Pour cette classe, il n'y a pas de notion de boîte de vérité terrain associée et donc le terme de régression est ignoré. Comme pour le RPN, la fonction de perte de régression (L_{reg}) est la *smooth L1 loss* définie dans le papier original de Fast R-CNN (Girshick, 2015). Encore une fois, λ est un facteur visant à balancer les deux termes de l'équation.

3.4.3. Détails d'implémentation

Tel que mentionné préalablement, l'encodeur utilisé dans le cadre de ce projet est le ResNet50 FPN, et ce puisqu'il s'agit de celui utilisé par la formule gagnante du *COCO-2017 detection challenge* (Peng et coll., 2018). Afin de mettre à profit l'apprentissage par transfert, le modèle complet est initialisé à l'aide de poids pré-entraînés sur le jeu de données *COCO-2017*. Comme dans l'implémentation originale de Faster R-CNN, les *anchors* sont générés à 3 échelles (128^2, 256^2 et 512^2 pixels) et 3 ratios

de forme (1:1, 1:2 et 2:1). Afin de pallier le débalancement des classes, la *cross entropy loss* de la tête de classification de Fast R-CNN est pondérée à partir de poids centrés autour de 1. Ces poids sont les suivants : 1 (arrière-plan), 1,7 (avion) et 0,45 (véhicule). L'entraînement est effectué sous Windows avec la libraire d'apprentissage machine PyTorch (Paszke et coll., 2017). L'optimisation du modèle est effectuée à l'aide de l'optimiseur *Adam* (Kingma et Ba, 2014), un *batch size* de 8 et un *learning rate* constant de 1e-4.

Le meilleur modèle est sélectionné sur la base des performances obtenues sur l'ensemble de validation. La métrique utilisée pour évaluer les performances est l'*average precision* (AP) à 50% d'IoU (Everingham et coll., 2010; Lin et coll., 2014). La librairie permettant de calculer cette métrique (*pycocotools*) étant seulement disponible sur Linux, la librairie développée par Rafael Padilla pour Windows a été utilisée[3].

3.5. Première approche : aucune adaptation de domaine

La première approche consiste à n'appliquer aucune adaptation de domaine. Cela signifie que le modèle préalablement entraîné sur le domaine source est directement utilisé pour performer sur l'ensemble de test du domaine cible, tel qu'illustré sur la Figure 17. Cette approche a pour but d'évaluer les pertes de performances occasionnées par le changement de domaine.

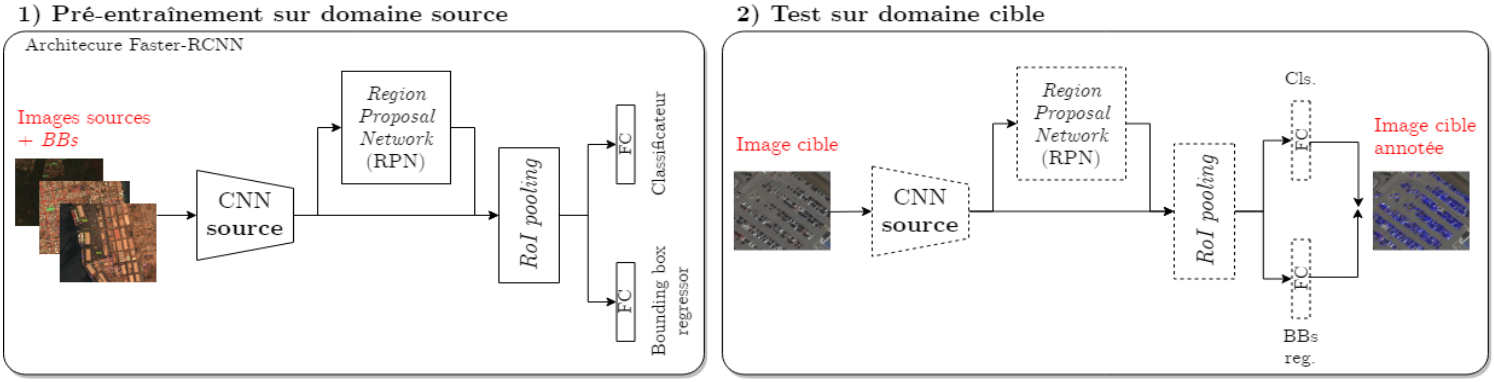


Figure 17 : Processus d'entraînement de la 1ere approche : aucune adaptation de domaine. Faster R-CNN est d'abord entraîné sur le domaine source, xView. Ce modèle est ensuite directement utilisé pour les tests sur le domaine cible, GeoImageNet. Les lignes pointillées indiquent que les poids sont figés. (Inspiré de Tzeng et coll., 2017).

[3] https://github.com/rafaelpadilla/Object-Detection-Metrics

3.6. Deuxième approche : *fine-tuning*

La deuxième approche est celle du *fine-tuning*. Le *fine-tuning* est une technique de *transfer learning* consistant à utiliser comme base d'entraînement un modèle préalablement entraîné sur un type de problème pour l'entraîner sur un autre problème ou un autre domaine. Cette technique est très courante en apprentissage profond et particulièrement utile lorsque la quantité de données d'entraînement est limitée. Dans ce cas-ci, le modèle pré-entraîné sur le domaine source est utilisé comme base d'entraînement et un *fine-tuning* est effectué vers le domaine cible à l'aide de l'ensemble d'entraînement disponible dans ce domaine. La Figure 18 illustre le processus d'entraînement de cette approche. Le *fine-tuning* est effectué avec l'optimiseur *Adam* (Kingma et Ba, 2014), un *batch size* de 8 et un *learning rate* constant de 1e-4. Aucune couche de l'architecture n'est gelée lors du *fine-tuning*.

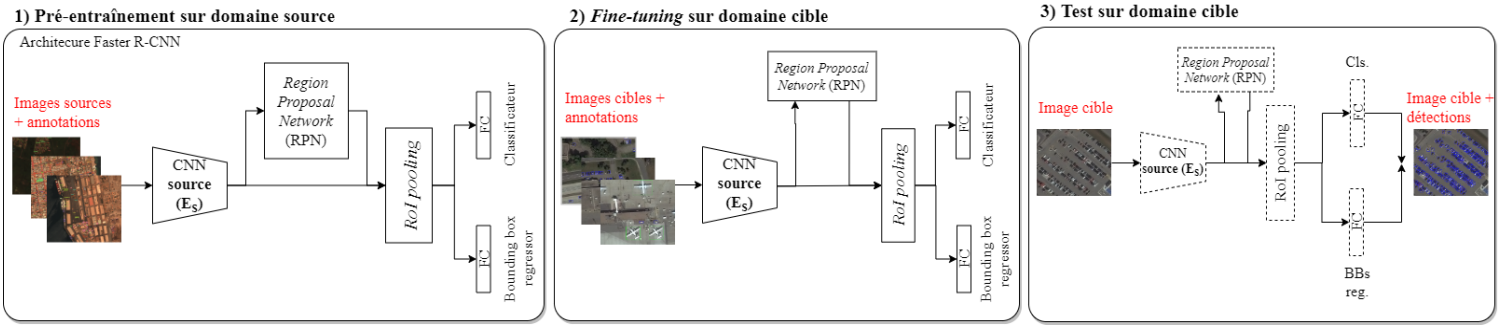


Figure 18 : Processus d'entraînement de la 2e approche : le *fine-tuning*. Faster R-CNN est d'abord entraîné sur le domaine source, xView. Ensuite, un *fine-tuning* est effectué sur le modèle préalablement entraîné sur le domaine source à l'aide de l'ensemble d'entraînement disponible dans le domaine cible. Après le *fine-tuning*, le modèle est utilisé pour performer sur l'ensemble de test du domaine cible. Les lignes pointillées indiquent que les poids sont figés. (Inspiré de Tzeng et coll., 2017)

3.7. Troisième approche : ADDA

La troisième approche est basée sur l'implémentation originale de ADDA. L'architecture est toutefois adaptée pour la détection d'objets en remplaçant le modèle de classification originale de ADDA (LeNet modifié) par Faster R-CNN. La Figure 19 illustre le processus d'entraînement de cette approche.

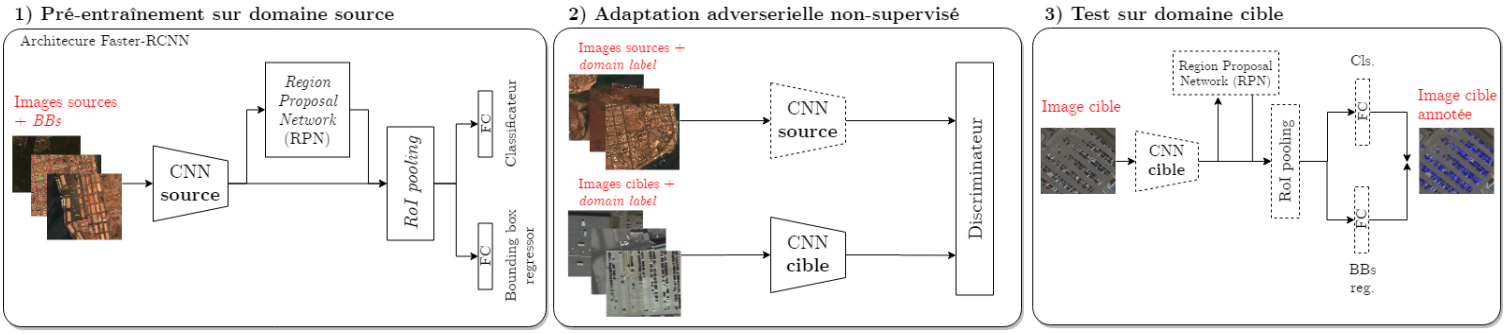


Figure 19 : Processus d'entraînement de la 3e approche : ADDA. Faster R-CNN est d'abord entraîné sur le domaine source, xView. Ensuite, durant l'adaptation adverserielle, un encodeur cible apprend à extraire des représentations sur le domaine cible statistiquement indissociables de celles extraites par l'encodeur source sur le domaine source de façon à tromper le discriminateur. Durant le test, l'encodeur cible est remis dans l'architecture de base pour performer sur le domaine cible. Les lignes pointillées indiquent que les poids sont figés. (Inspiré de Tzeng et coll., 2017)

Après le pré-entraînement sur le domaine source, un processus d'adaptation de domaine est mis en place de façon à apprendre à un encodeur cible (E_c) à extraire, sur le domaine cible, des représentations statistiquement indissociables des celles extraites par l'encodeur source (E_s) sur les données du domaine source, et ce grâce à un discriminateur (D) et un objectif adverseriel. Dans cette configuration, D est un MLP qui tente de distinguer de quel domaine provient les représentations qui lui sont présentées en donnant un chiffre entre 0 et 1 : 0 s'il pense que la représentation provient du domaine cible et 1 du domaine source.

Pendant l'adaptation adverserielle, les poids de E_c sont initialisés avec ceux de l'encodeur préalablement entraîné sur le domaine source (E_s) et les poids de E_s sont ensuite figés durant l'entraînement. Chaque itération s'effectue en deux étapes : 1) l'entrainement du discriminateur (D); et 2) l'entrainement de l'encodeur cible (E_c). Les fonctions de perte utilisées pour chacun de ces entraînements sont définies dans le paragraphe suivant. D'abord, pour entraîner D, N image(s) de chaque domaine sont envoyées à travers leur encodeur respectif, où N est le *batch size*. Les représentations extraites sont ensuite envoyées à D, chacune arrivant avec une étiquette exprimant son domaine de provenance, 0 pour le domaine cible et 1 pour le domaine source. D s'entraîne ainsi à différencier avec succès le domaine de provenance des représentations qu'il observe, et ce en rétropropageant dans D l'erreur entre le chiffre prédit et l'étiquette. Ensuite, pour entraîner E_c, N image(s) du domaine cible sont envoyées à travers E_c. Les représentations extraites sont ensuite envoyées à D avec une fausse étiquette. Plutôt que d'envoyer un 0, un 1 est envoyé et l'erreur de cette fausse étiquette est retropropagée dans E_c. Ainsi, si D donne 1 (c.-à-d. qu'il pense que la représentation provient du domaine source), les poids ne changeront pas. En effet, si D donne 1, cela signifie que E_c a produit, sur le domaine cible, une représentation tellement similaire à celles de E_s sur le domaine

source que D a été trompé. Au contraire, si D donne 0, c'est signe qu'il croit que la représentation provient du domaine cible et l'erreur maximale est donc retropropagée dans E_c. Chaque itération se déroule ainsi dans un processus adverseriel où D est mis à jour de façon à distinguer avec succès le domaine de provenance des représentations et E_c est mis à jour de façon à tromper D. Cet arrangement est similaire à celui des GANs où un générateur est mis à jour jusqu'à ce qu'il arrive à produire des images tellement similaires à celles de la distribution réelle que le discriminateur est trompé. Idéalement, après l'adaptation adverserielle, E_c extrait, sur le domaine cible, des représentations tellement similaires à celles de E_s sur le domaine source que D n'arrive plus à les différencier, ce dernier donne constamment 0,5.

D'un point de vue mathématique, ADDA minimise les distances entre les représentations source et cible en minimisant itérativement les fonctions suivantes, très similaires aux fonctions de perte classique des GANs :

$$\min_D \mathcal{L}_{adv_D}(X_s, X_c, E_s, E_c) = -\mathbb{E}_{x_s \sim X_s}\left[\log D\big(E_s(x_s)\big)\right] - \mathbb{E}_{x_c \sim X_c}[\log\Big(1 - D\big(E_c(x_c)\big)\Big)] \quad (3)$$

$$\min_{E_c} \mathcal{L}_{adv_E}(X_s, X_c, D) = -\mathbb{E}_{x_c \sim X_c}\left[\log D\big(E_c(x_c)\big)\right] \quad (4)$$

Où $E_s(x_s)$ et $E_c(x_c)$ correspondent aux représentations extraites respectivement par les encodeurs source et cible (E_s et E_c) à partir d'une image (x) issue des distributions de données source et cible (X_s et X_c). $D(E_s(x_s))$ correspond au score donné par le discriminateur (D) en voyant une représentation extraite par l'encodeur source (E_s) sur une image du domaine source (x_s), tandis que $D(E_c(x_c))$ correspond à ce même score, mais pour une représentation extraite sur image du domaine cible. $\mathcal{L}_{adv_D}$ est minimisé lors de l'entraînement du discriminateur. Dans cette formulation, si le discriminateur donne 1 en voyant une représentation extraite par l'encodeur source sur le domaine source ($D\big(E_s(x_s)\big) = 1$) et 0 en voyant une représentation extraite par l'encodeur cible sur le domaine cible ($D\big(E_c(x_c)\big) = 0$), les deux termes de la fonction sont nuls ($\log(1) - \log(1 - 0) = 0$), donc aucune erreur n'est retropropagée dans D. Au contraire, si D donne 0 sur le domaine cible et 1 sur le domaine source, l'erreur maximale est retropropagée puisque le discriminateur s'est trompé. Pour sa part, $\mathcal{L}_{adv_E}$ est minimisé lors de l'entraînement de l'encodeur cible. Comme mentionné préalablement, pour l'entraînement de l'encodeur cible, une fausse étiquette est envoyée (1 plutôt que 0). Ainsi, l'erreur maximale est retropropagée lorsque le discriminateur donne 0 ($-\log(0) = \infty$), soit lorsque le discriminateur pense que la représentation provient du domaine cible. Ceci est logique

puisque le but ici est de tromper le discriminateur. Après ce processus d'adaptation adverserielle, l'encodeur cible est remis dans l'architecture de Faster R-CNN pour performer sur l'ensemble de test du domaine cible tel qu'illustré sur la Figure 19.

Dans l'implémentation originale de ADDA, la tâche est très simple (classification de caractères – MNIST) donc le modèle de classification utilisé l'est également; il s'agit du LeNet modifié (Tzeng et coll., 2017). Lors de l'extraction des caractéristiques profondes, l'encodeur de ce modèle produit un total de 50 *feature maps* de 4 par 4, soit une représentation de $50\ x\ 4\ x\ 4$. De plus, avant d'être envoyée au classificateur, la dimensionnalité de cette représentation est davantage réduite vers un vecteur de caractéristique de 500 valeurs. En raison de la simplicité de la tâche, le discriminateur de l'implémentation originale est composé de seulement trois couches pleinement connectées, soit deux couches cachées de 500 neurones et la couche de sortie du discriminateur. Chaque couche cachée est suivie de la fonction d'activation ReLU.

Dans le cadre du projet actuel, la tâche est beaucoup plus complexe et le modèle utilisé l'est également (Faster R-CNN vs. LeNet modifié). Tel que mentionné préalablement, l'encodeur utilisé dans le présent projet est un ResNet50 FPN. Cet encodeur, plutôt que de produire une seule représentation, produit une pyramide de représentations à 5 étages, tel qu'illustré préalablement sur la Figure 16. On constate sur cette figure que même la représentation de plus petite dimension ($256\ x\ 13\ x\ 13$) possède environ 87 fois plus de paramètres que celle de l'implémentation originale (43 264 vs 500). Ainsi, pour pallier l'importante différence de paramètres, la complexité du discriminateur a été augmentée en ajoutant une couche cachée (pour un total de 3) et en augmentant la quantité de neurones à 1 000 dans chaque couche cachée. Du *dropout* a également été ajouté après chaque couche cachée, et ce à une probabilité de 25 %. Durant l'adaptation adverserielle, seule la dernière représentation de la pyramide est envoyée au discriminateur, soit celle de dimension $256\ x\ 13\ x\ 13$. Envoyer la pyramide complète aurait signifié envoyer 13 643 264 paramètres au discriminateur, ce qui aurait été très lourd même pour un discriminateur complexe (beaucoup de couches et beaucoup de neurones par couche) et très demandant en termes de mémoire. Les expérimentations sont effectuées à l'aide de l'optimiseur *Adam* (Kingma et Ba, 2014), d'un *batch size* de 2 et d'un *learning rate* constant de 1e-5 pour E_c et 1e-4 pour D.

3.8. Quatrième approche : semi-ADDA

La quatrième et dernière approche considérée, semi-ADDA, est très similaire à ADDA. La principale différence est que semi-ADDA, lors de l'adaptation adverserielle, effectue un *fine-tuning* occasionnel du modèle complet (E_c + parties de détection) en utilisant les quelques données d'entraînement disponibles dans le domaine cible. De cette façon, en plus de tirer profit des images non annotées du domaine cible, le modèle profite également des annotations disponibles dans ce domaine pour le guider dans l'apprentissage de ses poids. Le processus d'entraînement de semi-ADDA est présenté sur la Figure 20.

Dans l'implémentation originale de semi-ADDA, le *fine-tuning* est effectué à chaque 10 itérations de ADDA pendant 1 itération avec un *batch size* de 128. Puisque le modèle et les images utilisés dans le projet actuel sont beaucoup plus complexes, un *batch size* inférieur (2) a été utilisé, et ce dans le but d'éviter des problèmes en lien avec la mémoire du GPU lors de l'entraînement. Pour compenser le petit *batch size* utilisé dans notre implémentation, le *fine-tuning* est effectué à chaque 10 itérations de ADDA, mais pendant plus longtemps, soit 30 itérations. Ainsi, à chaque *fine-tuning,* plutôt que de voir 128 images avec leurs annotations, le modèle en voit 60. *Le fine-tuning* est effectué avec l'optimiseur *Adam* et un *learning rate* de 1e-4. Les hyperparamètres de ADDA (*batch size*, *learning rate*, optimiseur) demeurent les mêmes.

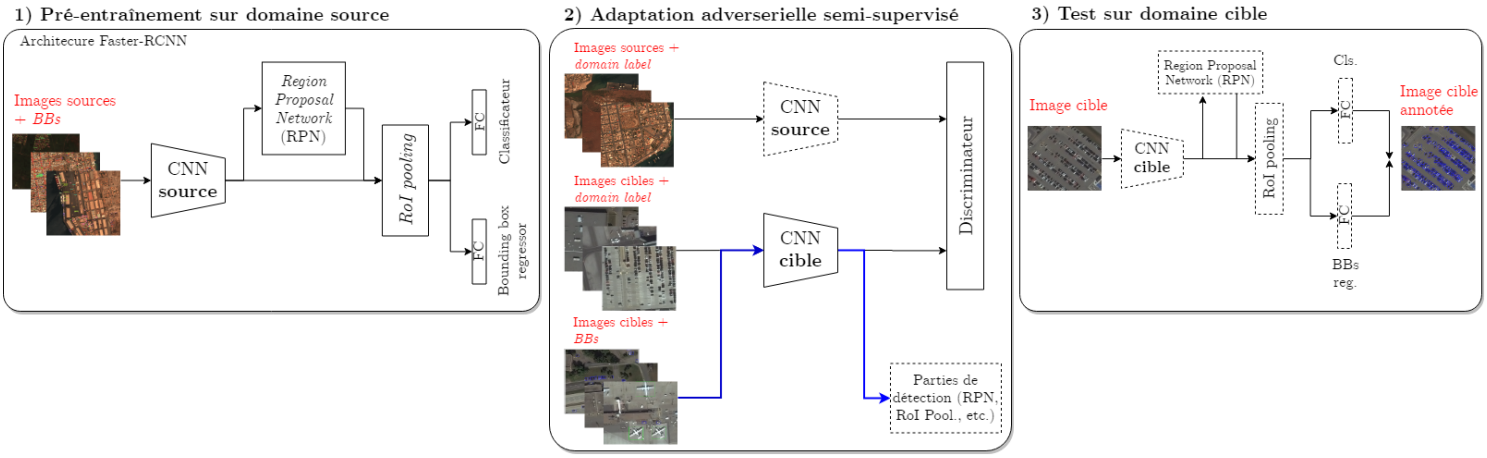


Figure 20 : Processus d'entraînement de la 3e approche : semi-ADDA. Faster R-CNN est d'abord entraîné sur le domaine source, xView. Ensuite, durant l'adaptation adverserielle, un encodeur cible apprend à extraire des représentations sur le domaine cible statistiquement indissociables de celles extraites par l'encodeur source sur le domaine source de façon à tromper le discriminateur. Durant cette étape, un *fine-tuning* du modèle complet (encodeur cible + parties de détection) est effectué occasionnellement à l'aide des annotations disponibles dans le domaine cible. Durant le test, l'encodeur cible est remis dans l'architecture de base pour performer sur le domaine cible. Les lignes pointillées indiquent que les poids sont figés. (Inspiré de Tzeng et coll., 2017)

4. Présentation des résultats

4.1. Pré-entraînement sur le domaine source

D'abord, il importe de présenter les résultats du pré-entraînement sur le domaine source, puisque cette étape constitue le fondement des diverses approches considérées. En raison de l'importance de cette étape, plusieurs tests ont été effectués dans le but d'optimiser la configuration de paramètres (*learning rate*, *batch size*, type d'optimiseur, pondération de la fonction de perte et augmentation de données). Après optimisation, le meilleur modèle obtient un mAP de 90,3 % sur l'ensemble de test de xView, soit un AP de 97,5 % pour la classe « avion » et de 83 % pour la classe « véhicule ». Considérant la piètre qualité des annotations provenant de xView, ce résultat est satisfaisant. Les prochaines sections montrent les résultats obtenus sur le domaine cible par les diverses approches considérées.

4.2. Première approche : aucune adaptation de domaine

Ici, le modèle préalablement entraîné sur le domaine source est directement utilisé pour performer sur l'ensemble de test du domaine cible. Cette approche a pour but d'évaluer les pertes de performances occasionnées par le changement de domaine. Tel que mentionné préalablement, puisque les images de GeoImageNet ont une résolution de 50 cm, un rééchantillonnage est effectué pour générer une fausse résolution de 30 cm. Le choix d'algorithme de rééchantillonnage a toutefois un impact sur les performances du modèle. Dans le cadre de ce projet, la libraire python *Pillow* a été utilisée pour cette tâche. Cette libraire offre plusieurs filtres de rééchantillonnage. À titre comparatif, quatre des filtres disponibles ont été testés. Les résultats sont présentés dans le Tableau 2. Antialias, qui est en fait un filtre Lanczos (Duchon, 1979), étant le filtre qui performe le mieux, celui-ci a été utilisé pour rééchantillonner l'entièreté des images du domaine cible.

Tableau 2 : Performances du modèle non adapté sur l'ensemble de test du domaine cible (GeoImageNet) en fonction du type de filtre de rééchantillonnage utilisé.

Filtre	AP@0.5 - Avion	AP@0.5 - Véhicule	mAP@0.5
Plus proche voisin	0,7532	0,2396	0,4964
Bilinéaire	0,8208	0,3824	0,6016
Bicubic	0,8235	0,435	0,6269
Antialias	**0,8242**	**0,4607**	**0,6425**

On constate donc que le changement de domaine occasionne des pertes significatives de performance avec un passage de 90,8 % de mAP sur le domaine source à 64,3 % sur le domaine cible, soit un AP de 82,4 % pour la classe « avion » et de 46,1 % pour la classe « véhicule » tel qu'illustré sur la courbe de Précision x Rappel présentée sur la Figure 21. Il est toutefois très difficile d'établir une comparaison valable, puisque les deux ensembles de test (source et cible) ne contiennent pas du tout le même nombre d'images (1 176 et 60 respectivement). Ces résultats permettent néanmoins de connaître l'ampleur des pertes de performance occasionnées par le changement de domaine.

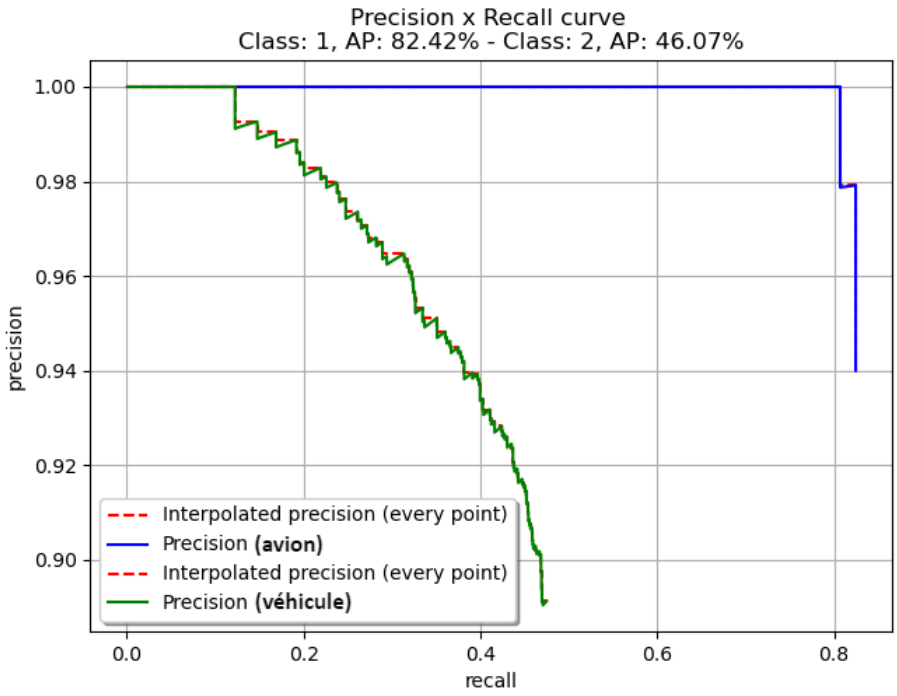


Figure 21 : Courbe de Précision x Rappel du modèle non adapté sur l'ensemble de test du domaine cible, GeoImageNet.

La précision et le rappel ont également été calculés sur l'entièreté de l'ensemble de test du domaine cible, et ce en produisant la matrice de confusion de chaque classe. Ces matrices sont présentées sur la Figure 22. Pour la classe « avion », on trouve une précision de 94 % et un rappel de 82,5 %. La classe « véhicule » possède pour sa part une précision de 89,1 % et un rappel de 47,5 %.

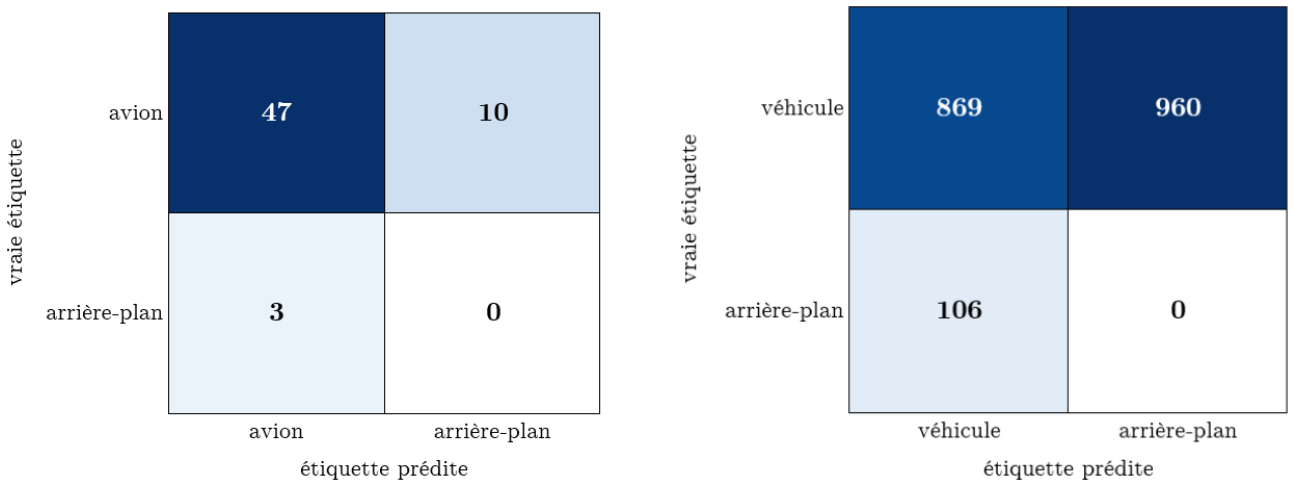


Figure 22 : À gauche, la matrice de confusion de la classe « avion » pour le modèle non adapté sur l'ensemble de test de GeoImageNet et, à droite, celle de la classe « véhicule ».

4.3. Deuxième approche : *fine-tuning*

La méthode du *fine-tuning* permet d'obtenir un mAP de 89,5 % sur l'ensemble de test du domaine cible, soit un AP de 89,4 % pour la classe « avion » et de 89,6 % pour la classe « véhicule » tel qu'illustré sur la courbe de Précision x Rappel présentée sur la Figure 23. Le meilleur résultat a été obtenu à la 3^{e} *epoch*.

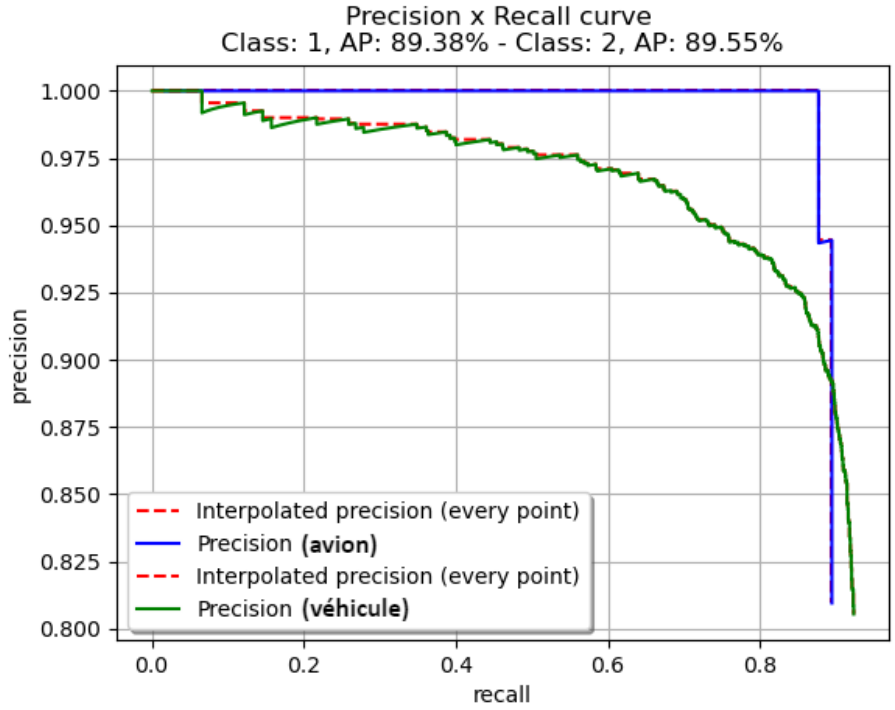


Figure 23 : Courbe de Précision x Rappel du modèle issu du *fine-tuning* sur l'ensemble de test du domaine cible, GeoImageNet.

Les matrices de confusion résultantes sont présentées sur la Figure 24. Pour la classe « avion », on trouve une précision de 81 % et un rappel de 89,5 %. La classe « véhicule » possède pour sa part une précision de 80,5 % et un rappel de 92,3 %.

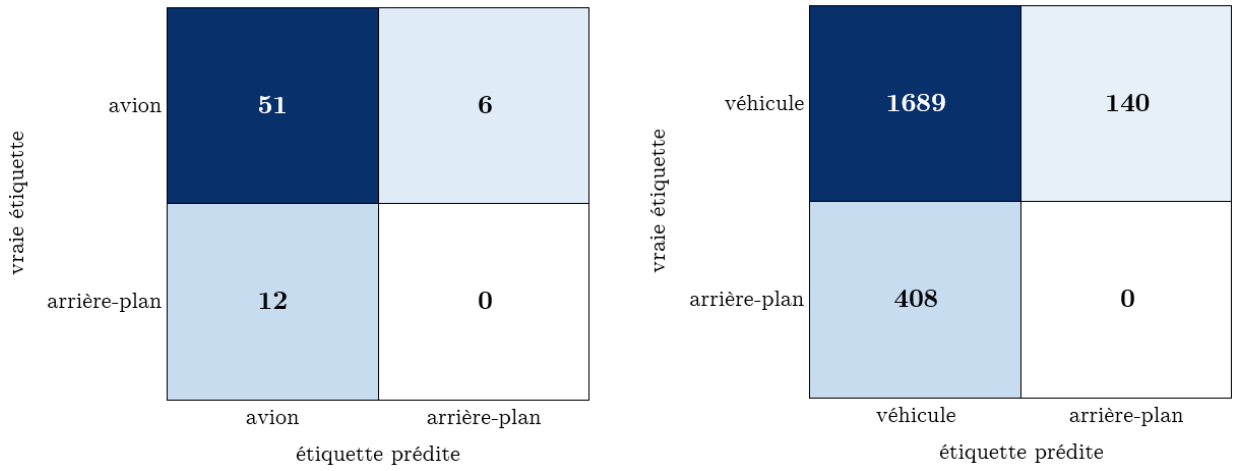


Figure 24 : À gauche, la matrice de confusion de la classe « avion » pour le modèle issu du *fine-tuning* sur l'ensemble de test de GeoImageNet et, à droite, celle de la classe « véhicule ».

4.4. Troisième approche : ADDA

L'implémentation de l'architecture d'adaptation de domaine ADDA augmente les performances du modèle en atteignant un mAP de 87,3 % sur l'ensemble de test du domaine cible, soit un AP de 94,8 % pour la classe « avion » et de 79,8 % pour la classe « véhicule » tel qu'illustré sur la courbe de Précision x Rappel présentée sur la Figure 25.

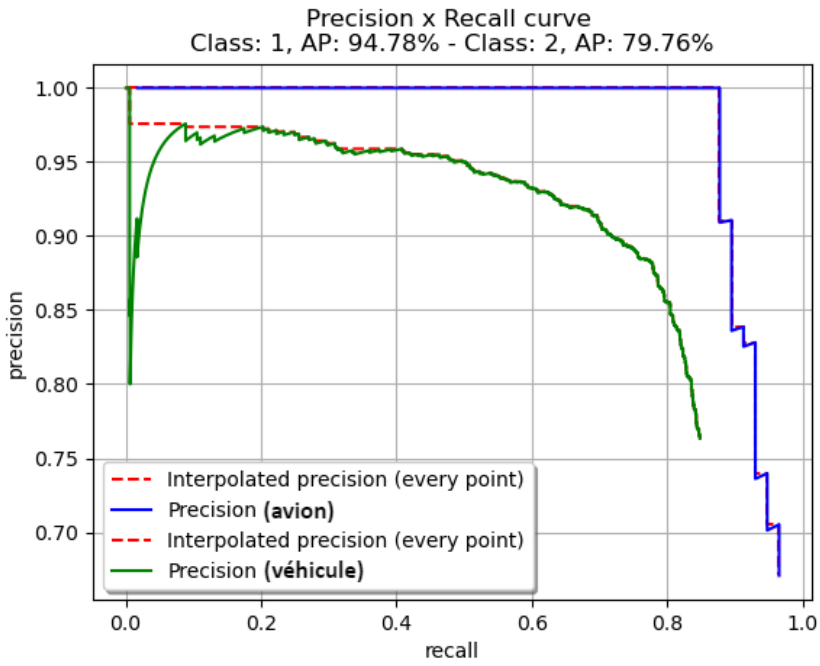


Figure 25 : Courbe de Précision x Rappel du modèle issu de ADDA sur l'ensemble de test du domaine cible, GeoImageNet.

La Figure 26 présente l'évolution du mAP sur l'ensemble de validation au fil des 200 premières itérations. Durant les premières itérations, ADDA permet de surpasser les performances atteintes avant l'adaptation de domaine (ligne pointillée orange = 64,3 %). Or, on constate qu'environ à la 100^{e} itération, le modèle commence à diverger. Une hypothèse expliquant ce comportement est postulée à la Section 5, « Interprétation des résultats ». Le meilleur résultat fut obtenu à la 29^{e} itération.

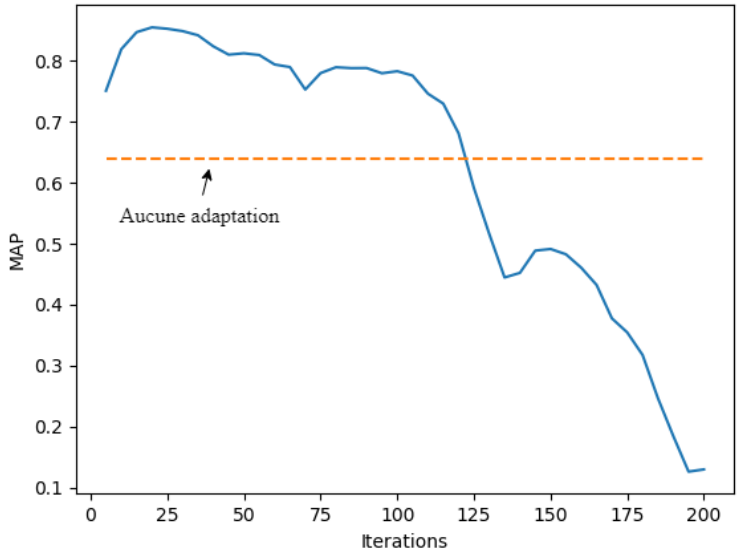


Figure 26 : Évolution du mAP sur l'ensemble de validation de domaine cible au fil des 200 premières itérations de ADDA. La ligne pointillée orange correspond à la performance du modèle non adapté.

Les matrices de confusion résultantes sont présentées sur la Figure 27. Pour la classe « avion », on trouve une précision de 67,1 % et un rappel de 96,5 %. La classe « véhicule » possède pour sa part une précision de 76,3 % et un rappel de 84,8 %.

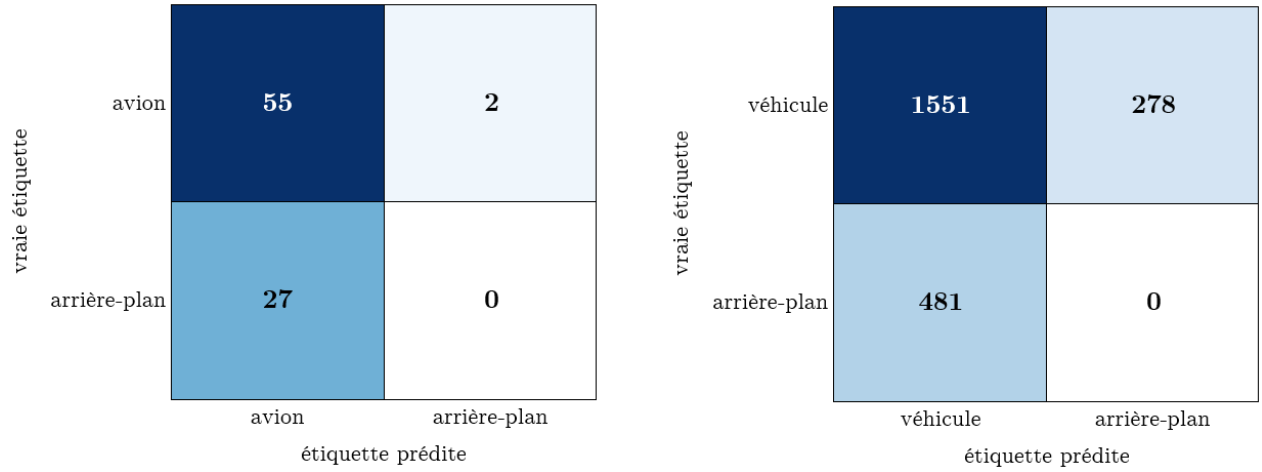


Figure 27 : À gauche, la matrice de confusion de la classe « avion » pour le modèle issu de ADDA sur l'ensemble de test de GeoImageNet et, à droite, celle de la classe « véhicule ».

4.5. Quatrième approche : semi-ADDA

Finalement, l'implémentation de semi-ADDA améliore davantage les performances, et ce avec l'obtention d'un mAP de 95,5 % sur l'ensemble de test du domaine cible, soit un AP de 98,8 % pour

la classe « avion » et de 92,1 % pour la classe « véhicule » tel qu'illustré sur la courbe de Précision x Rappel présentée sur la Figure 28.

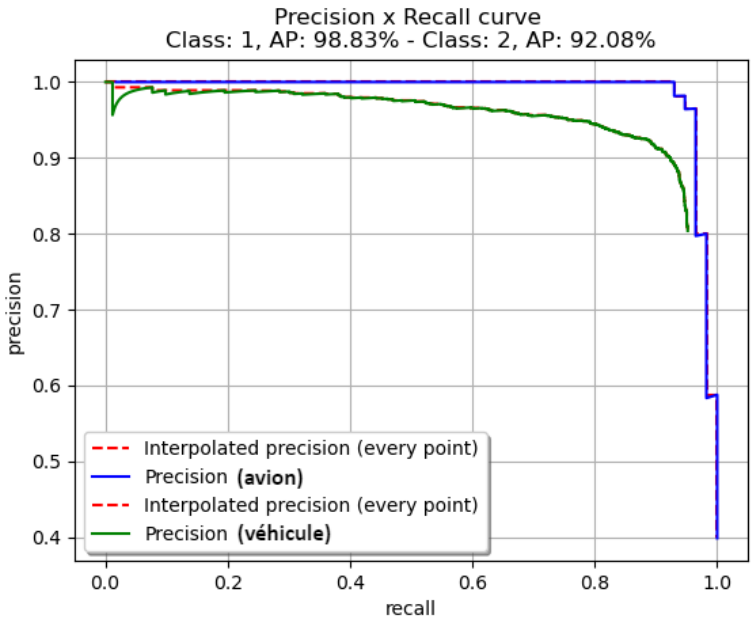


Figure 28 : Courbe de Précision x Rappel du modèle issu de semi-ADDA sur l'ensemble de test du domaine cible, GeoImageNet.

La Figure 29 présente l'évolution du mAP sur l'ensemble de validation au fil des 200 premières itérations. On constate que, contrairement à ADDA, l'entraînement ne diverge pas lors des 200 premières itérations. Le meilleur résultat fut obtenu à la 105^{e} itération.

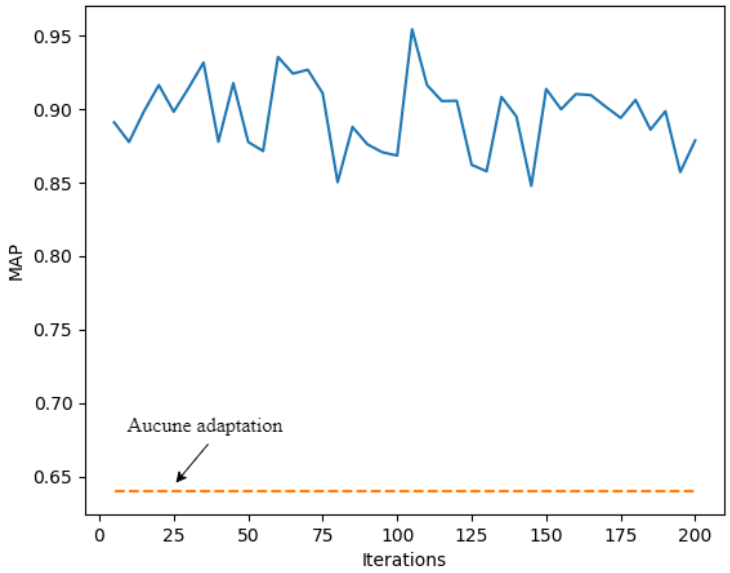


Figure 29 : Évolution du mAP sur l'ensemble de validation de domaine cible au fil des 200 premières itérations de semi-ADDA. La ligne pointillée orange correspond à la performance du modèle non adapté.

Les matrices de confusion résultantes sont présentées sur la Figure 30. Pour la classe « avion », on trouve une précision de 44,2 % et un rappel de 100 %. La classe « véhicule » possède pour sa part une précision de 73,1 % et un rappel de 95,7 %.

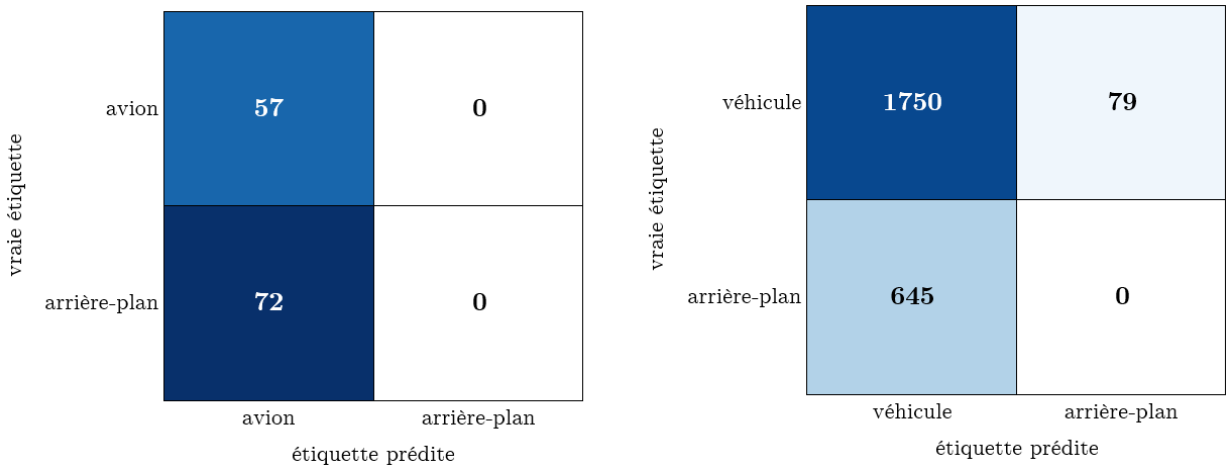


Figure 30 : À gauche, la matrice de confusion de la classe « avion » pour le modèle issu de semi-ADDA sur l'ensemble de test de GeoImageNet et, à droite, celle de la classe « véhicule ».

5. Interprétation des résultats

Le Tableau 2 présente une synthèse des résultats des diverses approches appliquées. Les résultats montrent l'efficacité de ADDA et semi-ADDA, même lorsqu'étendues à la détection d'objets,

améliorant significativement les performances sur le domaine cible comparativement au modèle non-adapté. On constate que semi-ADDA surpasse les autres méthodes avec une augmentation de 31,2 % de mAP par rapport au modèle non-adapté. Pour leur part, ADDA et le *fine-tuning* obtiennent des résultats similaires avec une augmentation de 23 % et 25,2 % de mAP respectivement par rapport au modèle non adapté.

Tableau 3 : Performances sur l'ensemble test du domaine cible (GeoImageNet) en fonction de l'approche utilisée.

Approche	**AP@0.5 - Avion**	**AP@0.5 - Véhicule**	**mAP@0.5**
Aucune adaptation	0.8242	0.4607	0.6425
Fine-tuning	0.8938	0.8955	0.8947
ADDA	0.9478	0.7976	0.8727
Semi-ADDA	**0.9881**	**0.9156**	**0.9518**

Pour ADDA, l'augmentation de 23 % est impressionnante considérant que le processus d'adaptation de cette architecture est entièrement non supervisé. À aucun moment, lors de l'entraînement, le modèle ne voit d'annotations provenant du domaine cible. Or, même en ne voyant aucune annotation, ADDA arrive à atteindre des résultats sur le domaine cible se rapprochant de ceux du *fine-tuning*.

On constate toutefois sur la Figure 10 que ADDA diverge rapidement lors de l'entraînement. Considérant que, avec un *batch size* de 2, une *epoch* dure environ 500 itérations, l'entraînement diverge avant même d'avoir fait la moitié d'une *epoch*. Il est fort probable que ce comportement s'explique par la nature non supervisée du processus. En effet, pendant l'adaptation adverserielle, les poids de l'encodeur cible sont mis à jour strictement dans le but de tromper le discriminateur et non dans une optique de détection d'objets. Lors des premières itérations, l'encodeur cible apprend à extraire des représentations de plus en plus similaires à celles extraites par l'encodeur source sur le domaine source et cela aide le modèle à atteindre de meilleures performances de détection sur le domaine cible. Or, rapidement, les représentations extraites perdent leur intérêt dans un contexte de détection d'objets. En effet, on observe sur la Figure 10 que, après 200 itérations, le mAP du modèle est de près de 15 % et est en chute libre depuis une centaine d'itérations. Si l'entraînement s'était poursuivi, l'encodeur cible aurait continué d'apprendre à extraire, sur le domaine cible, des représentations de plus en plus similaires à celles extraites par l'encodeur source sur le domaine source, mais ces représentations seraient devenues de moins en moins pertinentes pour la détection d'objets et le mAP aurait continué de chuter.

Dans semi-ADDA, en plus du processus non supervisé ayant bien fonctionné dans ADDA, le modèle complet est constamment redirigé vers la tâche d'intérêt (la détection d'objets) grâce à un *fine-tuning* ayant lieu à chaque 10 itérations de ADDA. Il en résulte un entraînement qui ne diverge pas et qui atteint un mAP de 95,5 % surpassant les autres méthodes.

Les résultats atteints par ADDA et semi-ADDA sont d'autant plus impressionnant considérant que, tel que mentionné dans la section 4.6, seul un étage de la pyramide de représentation extraite par l'encodeur est envoyé au discriminateur pendant le processus d'adaptation adverseriel. En classification, une seule représentation est généralement suffisante pour assurer la classification de l'image. Or, en détection d'objets, chaque étage de la pyramide joue un rôle important que ce soit pour la génération de RoIs par le RPN ou pour la classification de ces RoIs. En comparant l'entièreté de la pyramide, le discriminateur aurait peut-être eu plus de facilité à déceler les différences entre les domaines et à appliquer des changements significatifs aux poids de l'encodeur, ce qui aurait potentiellement permis d'augmenter la performance de ces approches.

En regardant les matrices de confusions issues de l'application des divers modèles sur le jeu de test de GeoImageNet, on remarque toutefois que les modèles issus de ADDA et semi-ADDA enregistrent une perte significative de précision. En effet, le modèle sans adaptation possède une précision de 94 % et 89,1 % sur les classes « avion » et « véhicule » respectivement. Or, après l'adaptation, la précision chute à 67,1 % et 76,3 % pour ces classes pour ADDA et à 44,2 % et 73,1 % pour semi-ADDA. Au contraire, le rappel du modèle sans adaptation est de 82,5 % pour la classe « avion » et de 47,5 % pour la classe « véhicule ». Suite à l'adaptation de domaine, le rappel augmente à 96,5 % (avion) et 84,8 % (véhicule) pour ADDA et à 100 % et 95,7 % pour semi-ADDA. On constate donc que les modèles avec adaptation sont nettement meilleurs pour trouver l'entièreté des objets d'intérêts au sein d'une image (rappel élevé), mais ce au détriment de leur précision. Pour sa part, le modèle issu du *fine-tuning* possède une précision et un rappel plus uniforme, et ce peu importe la classe. En effet, ce modèle possède une précision de 81 % pour la classe « avion » et 80,5 % pour la classe « véhicule » et un rappel de 89,5 % et 92,3 % pour ces classes respectivement.

La Figure 31 illustre quelques exemples de détections issues des divers modèles sur quatre images de l'ensemble de test de GeoImageNet comparativement à la vérité terrain. L'utilisation de métrique comme le mAP est très intéressante pour pouvoir comparer les performances de divers détecteurs, mais l'observation visuelle des résultats demeure une méthode de comparaison également très utile et instructive.

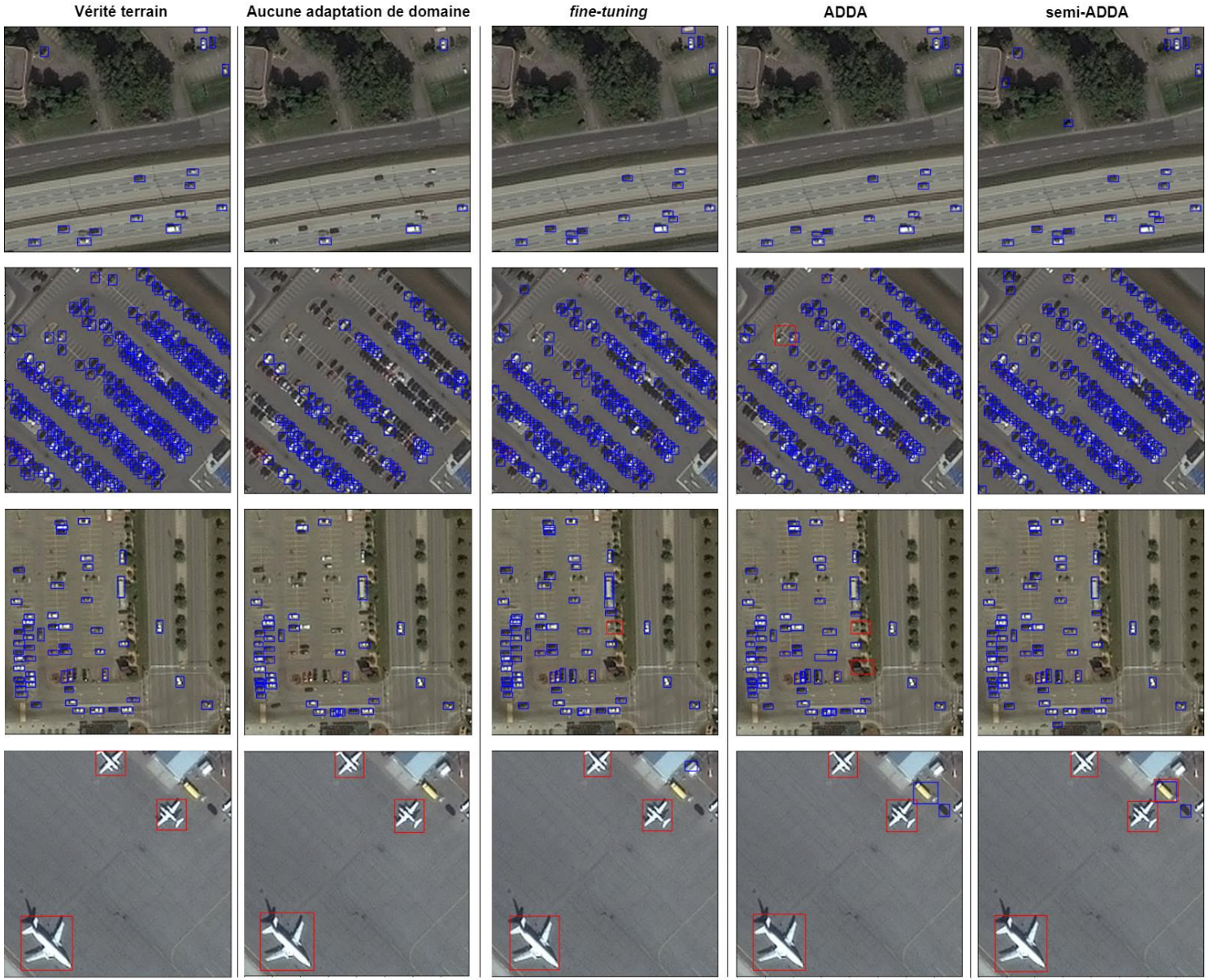


Figure 31 : Quelques exemples de détections effectuées par les différents modèles sur des images de l'ensemble de test de GeoImageNet comparativement à la vérité terrain.

On remarque clairement sur ces images les améliorations issues des méthodes d'adaptation de domaine par rapport au modèle non-adapté. On constate également que les différents modèles ont en commun une faiblesse : ils ont plus de difficulté à détecter les voitures noires, ce qui est comprenable considérant que les ces voitures ne possèdent pas autant de texture et de contraste que les voitures d'autres couleurs. Sur la deuxième image en partant du haut, on peut voir que ce phénomène est exacerbé lorsque plusieurs voitures noires se trouvent près l'une de l'autre. Lorsque plusieurs voitures noires sont regroupées, le modèle ne peut utiliser l'arrière-plan comme contraste dans la BB ce qui rend la tâche encore plus difficile. On observe cette différence sur la Figure 32 où on peut voir, à gauche, un regroupement de voitures blanches et, à droite, un regroupement de voitures noires. Les voitures blanches, en raison du contraste entre le pare-brise et la carrosserie, possèdent plus de texture et de contraste et sont donc plus faciles à identifier. Pour leur part, les voitures noires ressemblent seulement à un amoncellement de pixels noirs et possèdent très peu de contraste et de texture. Or, durant l'entraînement de Faster R-CNN, le modèle voit énormément de RoIs qui tombent sur des amoncellements de pixels noirs et qui ne sont pas des voitures, comme des plans d'eau ou des ombrages de bâtiments, et le modèle est entraîné à classifier ces RoIs comme appartenant à l'arrière-plan. Il est donc comprenable qu'il y ait une certaine confusion de la part du modèle pour ces voitures, puisque les différences de caractéristiques entre ces voitures et d'autres éléments d'arrière-plan peuvent parfois être subtils.

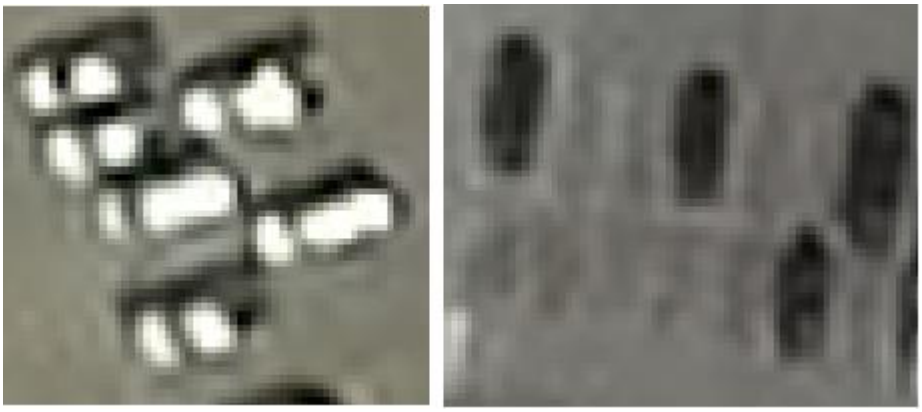


Figure 32 : À gauche, des voitures blanches et, à droite, des voitures noires. Issue d'image du domaine cible.

Dans le cadre de ce projet, le modèle final sera utilisé pour générer des détections qui seront envoyées sur la plateforme GeoImageNet en validation par les utilisateurs de la plateforme. Lors du processus de validation, trois options s'offrent à l'utilisateur : 1) accepter l'annotation telle quelle; 2) refuser l'annotation; ou 3) peaufiner l'annotation (affiner le positionnement de la boîte par exemple). Dans ce contexte, il est légitime de vouloir prioriser le rappel à la précision. En effet, si le modèle trouve

presque tous les objets d'intérêts (rappel élevé), supprimer les faux positifs est rapide. Il suffit d'un clic de souris ou deux et l'annotation erronée est supprimée. Au contraire, produire une nouvelle annotation à partir de rien signifie pour l'utilisateur de devoir délimiter la BB et l'assigner à la bonne classe. Dans un autre contexte où la précision aurait été la métrique la plus importante, les résultats montrent que le *fine-tuning* surpasse ADDA et semi-ADDA.

6. Discussions et perspectives

Dans l'implémentation originale de semi-ADDA, les auteurs testent leur architecture sur une tâche de classification de panneaux solaires. L'adaptation de domaine est effectuée entre deux villes américaines de l'état de la Californie : Fresno et Stockton. Toutes les images sont acquises lors du même mois de l'année 2013 et proviennent du même capteur. Dans cette expérimentation, les auteurs montrent que semi-ADDA surpasse constamment la méthode du *fine-tuning*, et ce peu importe la quantité d'annotations disponibles dans le domaine cible (Wang et coll., 2018).

Les résultats du présent projet concordent avec ceux trouvés par Wang et coll. (2018) et indiquent que ADDA et semi-ADDA peuvent également être étendues à la détection d'objets avec succès. Ici, semi-ADDA surpasse également la méthode du *fine-tuning*, et ce pour une quantité fixe d'annotations dans le domaine cible (2 937). Ces résultats montrent également que ces architectures sont en mesure de s'attaquer à des problèmes plus complexes d'adaptation de domaine où les images des domaines sources et cibles proviennent de régions différentes, mais également de capteurs différents.

Des développements subséquents peuvent être considérés afin d'améliorer la compréhension des résultats de ce projet. Premièrement, il serait intéressant d'évaluer l'impact de la structure du discriminateur sur les performances de ADDA et semi-ADDA. Dans le cadre du présent projet, une seule structure a été testée, soit 3 couches cachées (+ 1 couche de sortie) avec 1 000 neurones par couche et 25 % de probabilité de *dropout*. Or, le discriminateur étant une pièce maîtresse du processus d'adaptation adverseriel, sa structure peut potentiellement avoir un impact sur l'efficacité de ce processus. Dans des travaux futurs, il serait pertinent d'évaluer l'impact de la structure du discriminateur sur l'efficacité du processus d'adaptation en testant plusieurs configurations des paramètres suivants : nombre de couches, nombre de neurones par couches et probabilité de *dropout*.

Deuxièmement, il serait également intéressant d'évaluer l'impact du choix de la (ou des) représentation(s) envoyée(s) au discriminateur sur l'efficacité du processus d'adaptation adverseriel. Dans le cadre du présent projet, seule la dernière représentation de la pyramide extraite par l'encodeur

est envoyée au discriminateur. Il aurait théoriquement été possible d'envoyer la pyramide complète, mais cela aurait signifié envoyer 13 643 264 paramètres au discriminateur, ce qui aurait été très lourd même pour un discriminateur avec une structure complexe et demandant en termes de mémoire. Une solution potentielle aurait été de rééchantillonner les représentations des divers étages vers une même résolution (Par ex. $256\ x\ 50\ x\ 50$) pour pouvoir les concaténer et les envoyer au discriminateur. Pour une résolution de $256\ x\ 50\ x\ 50$, cela correspond à envoyer 3 200 000 paramètres au discriminateur, ce qui est beaucoup, mais déjà plus réaliste que 13 643 264. Il aurait aussi été possible d'envoyer seulement le 4^e étage ($256\ x\ 25\ x\ 25$) ou seulement le 3^e ($256\ x\ 50\ x\ 50$) ou même de faire des combinaisons en envoyant une concaténation des 4^e et 5^e étages. Bref, il y a beaucoup de combinaisons possibles et ce serait intéressant dans des travaux futurs d'évaluer l'impact de ce choix sur l'efficacité du processus d'adaptation adverseriel de ADDA et semi-ADDA. Il est toutefois important de mentionner que ce choix aura un impact sur la complexité nécessaire dans le discriminateur. Il devient donc difficile de tester conjointement la structure du discriminateur et le choix de la (des) représentation(s) envoyée(s) sans devoir faire énormément de tests.

Troisièmement, il serait également intéressant d'étudier l'impact des choix d'hyperparamètres (*learning rate*, *batch size*, optimiseur, etc.) sur les résultats des diverses approches. Ici, très peu de tests ont été effectués, sauf lors de l'entraînement sur le domaine cible. Or, il serait intéressant de mettre en place un processus d'optimisation des hyperparamètres à l'aide de méthodes de *grid searching* (LeCun et coll., 1998b), de *random searching* (Bergstra et Bengio, 2012) ou d'optimisation bayésienne (Balandat et coll., 2019).

Finalement, les résultats obtenus portent à croire que des travaux ultérieurs pourront étendre la méthode à d'autres algorithmes de détection d'objets. Par exemple, YOLO (Redmon et coll., 2016), SSD (Liu et coll., 2016) et RetinaNet (Lin et coll., 2017) commencent également leur *pipeline* par l'extraction d'une représentation à l'aide d'un encodeur. Moyennant quelques modifications à la méthode proposée dans le cadre de ce projet, il serait possible d'étendre la méthode à ces architectures. De plus, il est certainement possible d'étendre la méthode à la segmentation d'images à l'aide, par exemple, de l'architecture Mask R-CNN. Cette architecture est presque identique à Faster R-CNN à l'exception d'une branche supplémentaire dans la tête du modèle servant à prédire le masque de l'objet (He et coll., 2017). Il est donc légitime de croire que la méthode présentée dans ce projet puisse être étendue à Mask R-CNN avec succès. De plus, plusieurs architectures de segmentation possèdent une structure de type encodeur-décodeur, comme par exemple U-Net (Ronneberger et coll., 2015) ou

encore Deeplab V3+ (Chen et coll., 2018c). Ces architectures seraient également très bien adaptées pour une adaptation de domaine à l'aide de ADDA et semi-ADDA. La Figure 32 présente ce à quoi pourrait ressembler l'architecture de semi-ADDA pour le modèle Deeplab V3+.

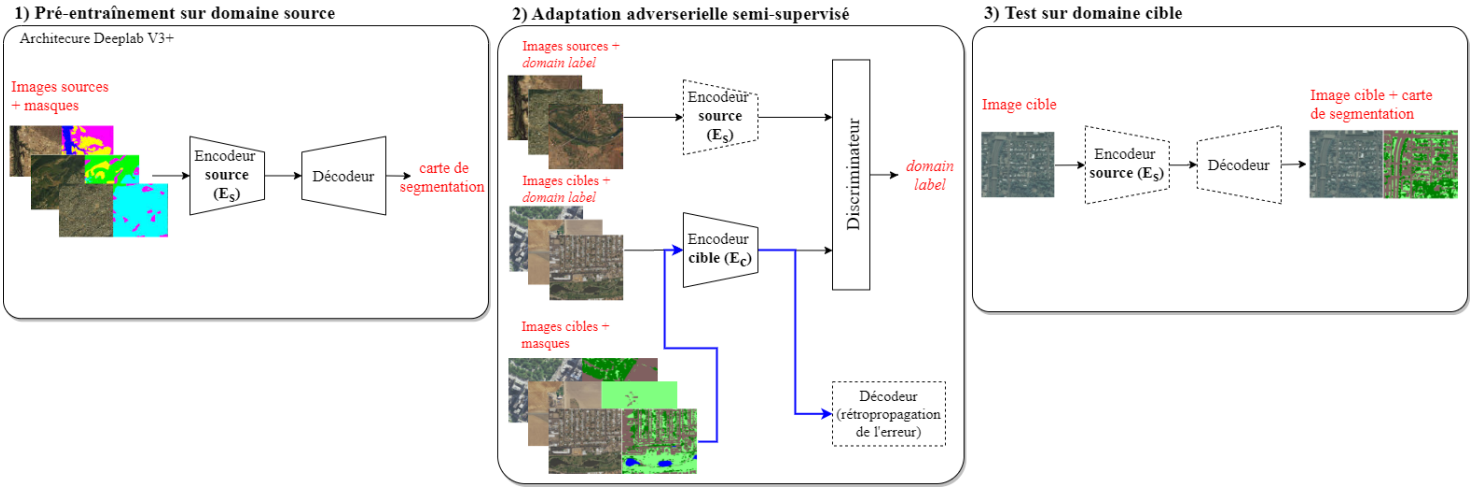


Figure 33 : Architecture proposée pour l'application de semi-ADDA à la segmentation sémantique à l'aide du modèle Deeplab V3+. (Inspiré de Tzeng et coll., 2017)

7. Conclusion

En raison de sa vaste superficie, le Canada utilise les images satellitaires pour un grand nombre d'applications de gestion du territoire, d'estimation des ressources naturelles, de suivi environnemental, de planification urbaine, de développement du Grand Nord et autres. Dans cette optique, la méthode développée et les images annotées pourront profiter aux utilisateurs de la plateforme GeoImageNet afin de réaliser davantage d'applications et de développements dans le domaine de l'apprentissage profond appliqué à la télédétection. Cette innovation contribuera à maintenir le Canada à l'avant-garde des solutions numériques innovantes pour la gestion durable du territoire.

De plus, étant une amélioration d'architectures existantes, les méthodes proposées dans le cadre de cette recherche permettent l'avancement de la science dans le champ de l'adaptation de domaine et de la détection d'objets en télédétection. Cette recherche permettra possiblement d'améliorer les performances des applications pratiques de détection d'objets, puisque celles-ci demeurent confrontées à d'importants défis liés au décalage pouvant exister entre les données d'entraînement et de test. Considérant que les architectures d'adaptation de domaine ADDA et semi-ADDA n'avaient jamais été adaptées à la détection d'objets, les résultats de ce projet sont très prometteurs.

Dans le cadre de GeoImageNet, les méthodes pourront potentiellement être étendues à de nouveaux jeux de données de détection d'objets ou de segmentation possédant des classes en concordance avec la taxonomie de GeoImageNet afin de réduire davantage le fardeau d'annotation inhérent à la création de ce jeu de données.

8. Références

Bai, X., Zhang, H., et Zhou, J., 2014, VHR object detection based on structural feature extraction and query expansion, In IEEE Transactions on Geoscience and Remote Sensing, tome 52, n°10, p. 6508-6520.

Balandat, M., Karrer, B., Jiang, D. R., Daulton, S., Letham, B., Wilson, A. G., et Bakshy, E., 2019, Botorch: Programmable bayesian optimization in pytorch, arXiv preprint arXiv:1910.06403.

Bergstra, J., et Bengio, Y., 2012, Random search for hyper-parameter optimization, In The Journal of Machine Learning Research, tome 13, n°1, p. 281-305.

Bouroubi, Y., Chapdelaine, C., Foucher, S., Byrns, D., Beaulieu, M., St-Charles, P.-L., Germain, M., Lauzier-Hudon, É., Bugnet, P., Sabo, N. et Gosselin, C., 2019, GeoImageNet: a Collaborative Platform for Deep Learning Application to Very High Resolution EO Images, 40 ième Symposium canadien de télédétection, 4-6 juin 2019.

Chen, Y., Li, W. et Van Gool, L., 2018a, Road: Reality oriented adaptation for semantic segmentation of urban scenes, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 7892-7901.

Chen, Y., Li, W., Sakaridis, C., Dai, D. et Van Gool, L., 2018b, Domain adaptive faster r-cnn for object detection in the wild, Proceedings of the IEEE conference on computer vision and pattern recognition, p. 3339-3348.

Chen, L. C., Zhu, Y., Papandreou, G., Schroff, F., et Adam, H., 2018c, Encoder-decoder with atrous separable convolution for semantic image segmentation, In Proceedings of the European conference on computer vision (ECCV), p. 801-818.

Cheng, G., Guo, L., Zhao, T., Han, J., Li, H., et Fang, J., 2013a, Automatic landslide detection from remote-sensing imagery using a scene classification method based on BoVW and pLSA, In International Journal of Remote Sensing, tome 34, n°1, p. 45-59.

Cheng, G., Han, J., Guo, L., Qian, X., Zhou, P., Yao, X., et Hu, X., 2013b, Object detection in remote sensing imagery using a discriminatively trained mixture model, In ISPRS Journal of Photogrammetry and Remote Sensing, tome 85, p. 32-43.

Cheng, G., et Han, J., 2016, A survey on object detection in optical remote sensing images, In ISPRS Journal of Photogrammetry and Remote Sensing, tome 117, p. 11-28.

Christie, G., Fendley, N., Wilson, J. et Mukherjee, R., 2018, Functional map of the world, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 6172-6180.

Csurka, G., 2017, Domain adaptation for visual applications: A comprehensive survey, arXiv preprint arXiv:1702.05374.

Dahmane, M., Foucher, S., Beaulieu, M., Riendeau, F., Bouroubi, Y., et Benoit, M., 2016, Object detection in pleiades images using deep features, In 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), p. 1552-1555.

Dai, J., Li, Y., He, K. et Sun, J., 2016, R-fcn: Object detection via region-based fully convolutional networks, Advances in neural information processing systems, p. 379-387.

Demir, I., Koperski, K., Lindenbaum, D., Pang, G., Huang, J., Basu, S. et Raska, R., 2018, Deepglobe 2018: A challenge to parse the earth through satellite images, In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, p. 172-17209.

Duchon, C. E., 1979, Lanczos filtering in one and two dimensions, In Journal of applied meteorology, tome 18, n°8, p. 1016-1022.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J. et Zisserman, A., 2010, The pascal visual object classes (voc) challenge, International journal of computer vision, tome 88, n°2, p. 303-338.

Ganin, Y. et Lempitsky, V., 2014, Unsupervised domain adaptation by backpropagation, arXiv preprint arXiv:1409.7495.

Ghifary, M., Kleijn, W. B., Zhang, M., Balduzzi, D. et Li, W., 2016, Deep reconstruction-classification networks for unsupervised domain adaptation, European Conference on Computer Vision, p. 597-613.

Gidaris, S. et Komodakis, N., 2015, Object detection via a multi-region and semantic segmentation-aware cnn model, Proceedings of the IEEE international conference on computer vision, p. 1134-1142.

Girshick, R., Donahue, J., Darrell, T. et Malik, J., 2014, Rich feature hierarchies for accurate object detection and semantic segmentation, In Proceedings of the IEEE conference on computer vision and pattern recognition, p. 580-587.

Girshick, R., 2015, Fast r-cnn, Proceedings of the IEEE international conference on computer vision, p. 1440-1448.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S. et Bengio, Y., 2014, Generative adversarial nets, Advances in neural information processing systems, p. 2672-2680.¸

Gui, J., Sun, Z., Wen, Y., Tao, D., et Ye, J., 2020, A review on generative adversarial networks: Algorithms, theory, and applications, arXiv preprint arXiv:2001.06937.¸

Han, J., Zhou, P., Zhang, D., Cheng, G., Guo, L., Liu, Z., ... et Wu, J., 2014, Efficient, simultaneous detection of multi-class geospatial targets based on visual saliency modeling and discriminative learning of sparse coding, In ISPRS Journal of Photogrammetry and Remote Sensing, tome 89, p. 37-48.

He, K., Zhang, X., Ren, S. et Sun, J., 2015, Spatial pyramid pooling in deep convolutional networks for visual recognition, IEEE transactions on pattern analysis and machine intelligence, tome 37, n° 9, p. 1904-1916.

He, K., Zhang, X., Ren, S. et Sun, J., 2016, Deep residual learning for image recognition, Proceedings of the IEEE conference on computer vision and pattern recognition, p. 770-778.

He, K., Gkioxari, G., Dollár, P. et Girshick, R., 2017, Mask r-cnn, Proceedings of the IEEE international conference on computer vision, p. 2961-2969.

Hoffman, J., Wang, D., Yu, F. et Darrell, T., 2016, Fcns in the wild: Pixel-level adversarial and constraint-based adaptation, arXiv preprint arXiv:1612.02649.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A. et Murphy, K., 2017, Speed/accuracy trade-offs for modern convolutional object detectors, Proceedings of the IEEE conference on computer vision and pattern recognition, p. 7310-7311.

Hull, J. J., 1994, A database for handwritten text recognition research, In IEEE Transactions on pattern analysis and machine intelligence, tome 16, n°5, p. 550-554.

Hsu, H. K., Yao, C. H., Tsai, Y. H., Hung, W. C., Tseng, H. Y., Singh, M., et Yang, M. H., 2020, Progressive domain adaptation for object detection, In The IEEE Winter Conference on Applications of Computer Vision, p. 749-757.

Kingma, D. P. et Ba, J., 2014, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

Krizhevsky, A., Sutskever, I. et Hinton, G.E., 2012, Imagenet classification with deep convolutional neural networks, Advances in neural information processing systems, p. 1097-1105.

Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., ... et Ferrari, V., 2018, The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale, arXiv preprint arXiv:1811.00982.

Lam, D., Kuzma, R., McGee, K., Dooley, S., Laielli, M., Klaric, M. et McCord, B., 2018, xview: Objects in context in overhead imagery, arXiv preprint arXiv:1802.07856.

LeCun, Y., Bottou, L., Bengio, Y. et Haffner, P., 1998a, Gradient-based learning applied to document recognition, Proceedings of the IEEE, tome 86, n°11, p. 2278-2324.

LeCun, Y., Bottou, L., Orr, G., et Muller, K. R., 1998b, Efficient backprop, In Neural Networks: Tricks of the Trade, New York: Springer.

Li, K., Cheng, G., Bu, S., et You, X., 2018, Rotation-insensitive and context-augmented object detection in remote sensing images. In IEEE Transactions on Geoscience and Remote Sensing, tome 56, n°4, p. 2337-2348.

Li, K., Wan, G., Cheng, G., Meng, L., et Han, J., 2020, Object detection in optical remote sensing images: A survey and a new benchmark, In ISPRS Journal of Photogrammetry and Remote Sensing, tome 159, p. 296-307.

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D. et Zitnick, C. L., 2014, Microsoft coco: Common objects in context, European conference on computer vision, p. 740-755.

Lin, T. Y., Goyal, P., Girshick, R., He, K., et Dollár, P., 2017, Focal loss for dense object detection, In Proceedings of the IEEE international conference on computer vision, p. 2980-2988.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y. et Berg, A. C., 2016, Ssd: Single shot multibox detector, European conference on computer vision, p. 21-37.

Liu, M. Y. et Tuzel, O., 2016, Coupled generative adversarial networks, Advances in neural information processing systems, p. 469-477.

Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X. et Pietikäinen, M., 2018, Deep learning for generic object detection: A survey, arXiv preprint arXiv:1809.02165.

Long, M., Cao, Y., Wang, J. et Jordan, M. I., 2015, Learning transferable features with deep adaptation networks, arXiv preprint arXiv:1502.02791.

Maggiori, E., Tarabalka, Y., Charpiat, G., et Alliez, P., 2017, Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark, In 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), p. 3226-3229.

Michieli, U., Biasetton, M., Agresti, G., et Zanuttigh, P., 2020, Adversarial learning and self-teaching techniques for domain adaptation in semantic segmentation, In IEEE Transactions on Intelligent Vehicles.

Mundhenk, T. N., Konjevod, G., Sakla, W. A. et Boakye, K., 2016, A large contextual dataset for classification, detection and counting of cars with deep learning, European Conference on Computer Vision, p. 785-800.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B. et Ng, A. Y., 2011, Reading digits in natural images with unsupervised feature learning.

Oquab, M., Bottou, L., Laptev, I. et Sivic, J., 2015, Is object localization for free?-weakly-supervised learning with convolutional neural networks, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 685-694.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., ... et Lerer, A., 2017, Automatic differentiation in pytorch.

Peng, C., Xiao, T., Li, Z., Jiang, Y., Zhang, X., Jia, K., ... et Sun, J., 2018, Megdet: A large mini-batch object detector, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 6181-6189.

Raj, A., Namboodiri, V. P. et Tuytelaars, T., 2015, Subspace alignment based domain adaptation for rcnn detector, arXiv preprint arXiv:1507.05578.

Redmon, J., Divvala, S., Girshick, R. et Farhadi, A., 2016, You only look once: Unified, real-time object detection, Proceedings of the IEEE conference on computer vision and pattern recognition, p. 779-788.

Ren, X. et Ramanan, D., 2013, Histograms of sparse codes for object detection, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 3246-3253.

Ren, S., He, K., Girshick, R. et Sun, J., 2015, Faster r-cnn: Towards real-time object detection with region proposal networks, Advances in neural information processing systems, p. 91-99.

Rodriguez, A. L., et Mikolajczyk, K., 2019, Domain adaptation for object detection via style consistency. arXiv preprint arXiv:1911.10033.

Ronneberger, O., Fischer, P., et Brox, T., 2015, U-net: Convolutional networks for biomedical image segmentation, In International Conference on Medical image computing and computer-assisted intervention, p. 234-24.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstain, M. et autres, 2015, Imagenet large scale visual recognition challenge, International journal of computer vision, tome 115, n°3, p. 211-252.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R. et LeCun, Y., 2013, Overfeat: Integrated recognition, localization and detection using convolutional networks, arXiv preprint arXiv:1312.6229.

Sghaier, M. O., Foucher, S., Lepage, R., et Dahmane, M., 2016, Combination of texture and shape analysis for a rapid rivers extraction from high resolution SAR images, In 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), p. 673-676.

Sun, B., Feng, J. et Saenko, K., 2016, Return of frustratingly easy domain adaptation, Thirtieth AAAI Conference on Artificial Intelligence.

Sun, B. et Saenko, K., 2016, Deep coral: Correlation alignment for deep domain adaptation, European Conference on Computer Vision, p. 443-450.

Turgeon-Pelchat, M, 2019, Literature review on the potential of high-resolution optical satellite imagery for the extraction of mapping information, Geomatics Canada, Open File 50.

Tzeng, E., Hoffman, J., Zhang, N., Saenko, K. et Darrell, T., 2014, Deep domain confusion: Maximizing for domain invariance, arXiv preprint arXiv:1412.3474.

Tzeng, E., Hoffman, J., Saenko, K. et Darrell, T., 2017, Adversarial discriminative domain adaptation, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, p. 7167-7176.

Uijlings, J. R., Van De Sande, K. E., Gevers, T. et Smeulders, A. W., 2013, Selective search for object recognition, International journal of computer vision, tome 104, n°2, p. 154-171.

Van Etten, A., Lindenbaum, D. et Bacastow, T. M., 2018, Spacenet: A remote sensing dataset and challenge series, arXiv preprint arXiv:1807.01232.

Voulodimos, A., Doulamis, N., Doulamis, A., et Protopapadakis, E., 2018, Deep learning for computer vision: A brief review, Computational intelligence and neuroscience, 2018.

Wang, M. et Deng, W., 2018, Deep visual domain adaptation: A survey, Neurocomputing, tome 312, p. 135-153.

Wang, S., Bai, M., Mattyus, G., Chu, H., Luo, W., Yang, B. et Urtasun, R., 2016, Torontocity: Seeing the world with a million eyes, arXiv preprint arXiv:1612.00423.

Wang, R., Camilo, J., Collins, L. M., Bradbury, K. et Malof, J. M., 2017, The poor generalization of deep convolutional networks to aerial imagery from new geographic locations: an empirical

study with solar array detection, 2017 IEEE Applied Imagery Pattern Recognition Workshop, p. 1-8.

Wang, R., Collins, L. M., Bradbury, K. et Malof, J. M., 2018, Semisupervised Adversarial Discriminative Domain Adaptation, with Application to Remote Sensing Data, 2018 IEEE International Geoscience and Remote Sensing Symposium, p. 3611-3614.

Xu, J., Ramos, S., Vázquez, D. et López, A. M., 2014, Domain adaptation of deformable part-based models, IEEE transactions on pattern analysis and machine intelligence, tome 36, n°12, p. 2367-2380.

Yang, Y. et Newsam, S., 2010, Bag-of-visual-words and spatial extensions for land-use classification, In Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems, p. 270-279.

Zitnick, C. L. et Dollár, P., 2014, Edge boxes: Locating object proposals from edges, European conference on computer vision, p. 391-405.

Zhang, Y., David, P. et Gong, B., 2017, Curriculum domain adaptation for semantic segmentation of urban scenes, Proceedings of the IEEE International Conference on Computer Vision, p. 2020-2030.

Zhao, Z. Q., Zheng, P., Xu, S. T. et Wu, X., 2019, Object detection with deep learning: A review. IEEE transactions on neural networks and learning systems.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A. et Torralba, A., 2014, Object detectors emerge in deep scene cnns, arXiv preprint arXiv:1412.6856.

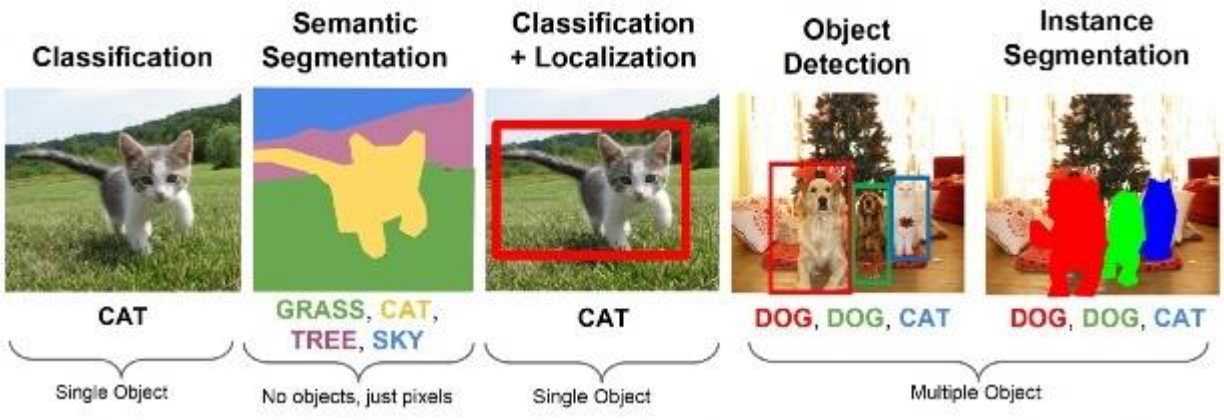
Zhou, P., Cheng, G., Liu, Z., Bu, S., et Hu, X., 2016a, Weakly supervised target detection in remote sensing images based on transferred deep features and negative bootstrapping, In Multidimensional Systems and Signal Processing, tome 27, n°4, p. 925-944.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A. et Torralba, A., 2016b, Learning deep features for discriminative localization, Proceedings of the IEEE conference on computer vision and pattern recognition, p. 2921-2929.

Zhu, X. X., Tuia, D., Mou, L., Xia, G. S., Zhang, L., Xu, F., et Fraundorfer, F., 2017, Deep learning in remote sensing: A comprehensive review and list of resources. IEEE Geoscience and Remote Sensing Magazine, tome 5, n°4, p 8-36.

9. Annexes

Annexe 1 : Quelques tâches de vision par ordinateur



Annexe 2 : Taxonomie des objets de GeoImageNet et xView

GeoImageNet

Type de l'objet	Objet (classe)
Batiment résidentiel	Unifamilial
	Unifamiliale détaché
	Unifamiliale jumelé
	Multifamiliale
	Maison mobile
	Autre bâtiment
Batiment commercial et service	Centre commercial
	Commerce indépendant (pharmacie, épicerie, ...)
	Commerce de véhicules
	Station de service et garage
	Hôtel
	Motel
	Restaurant, brasserie
	Autre commerce/service
Batiment de service et d'utilité publique	Édifice gouvennemental
	Hôtel de ville
	Établissement d'incarcération
	Poste de police
	Poste de pompiers
	Hôpital
	Église
	Maison de religieux
	Université, collège
	École primaire/secondaire
	Cinéma extérieur
	Aréna
	Stade
	Centre sportif
	Terrain de jeu, parc urbain
	Cimetière
	Poste de douane
	Observatoire
	Chateau d'eau
	Bassin de rétention
Installations sportives extérieures	Terrain de baseball
	Terrain de soccer
	Terrain de football
	Piscine extérieure
	Terrain de tennis
	Terrain de basketball
	Piste d'athlétisme
	Terrain à vocation multiples
	Terrain de volley-ball
	Terrain de golf
	Centre de ski
	Piste de course
	Parc d'attractions
	hippodrome
Batiment ou infrastructure industriel	Usine d'extraction (carrière, sablière, gravière)
	Usine de fabrication/transformation légère
	Usine de fabrication/transformation lourde
	Entrepôt de marchandise
	Entrepôt d'hydrocarbures
	Station d'épuration des eaux
	Bassin de décantation
	Site d'enfouissement (dépôt de déchets)
	Réservoir
	Cheminée
	Usine autre

Type de l'objet - suite	Objet (classe) - suite
Autre objets bâtis (mobilier urbain)	Clôture
	Lampadaire
	Trottoir
	Haie
	Poteau électrique
	Abribus
Infrastructure de transport routier	Autoroute
	Échangeur
	Route pavée
	Route non pavée
	Pont
	Viaduc
	Balance routière
	Centre de services autoroutiers
	Poste de péage
	Signalisation routière
	Stationnement
	Balance routière
	Paraneige
	Tunnel (entrée et sortie)
Infrastructure de transport ferroviaire	Gare de train
	Voie ferrée unique
	Voie ferrée multiple
	Voie de garage
	Gare de triage
	Aiguillage ferroviaire
Infrastructure de transport maritime	Port commercial
	Marina / Port de plaisance
	Port militaire
	Quai
	Rampe de mise à l'eau
	Écluse
Infrastructure de transport aérien	Aéroport
	Piste de décollage
	Hangar
Moyen de transport mobile	Voiture
	Fourgonnette
	Camionnette
	Autocar/autobus
	Train - locomotive
	Train - wagon de passagers
	Train - wagon de marchandise
	Bateau de plaisance, voilier
	Bateau de marchandise
	Porte contenaires
	Pétrolier/méthanier
	Avion type cesna
	Avion de ligne
	Avion militaire
	Train routier
	Paquebot
	Navire militaire
Télécommunications	Tour (dédiée)
	Antenne de communication (sur toît)
	Station de réception satellite (grandes paraboles)

Type de l'objet - suite	Objet (classe) - suite
Énergie	Ligne de haute tension
	Pipeline
	Pylône
	Centrale thermique
	Centrale hydro-électrique
	Centrale nucléaire
	Éolienne
	Panneau solaire
	Transformateur d'électricité, poste électrique
	Puit de pétrole/gaz
	Raffinerie
Structures anthropiques hydrographiques	Digue
	Brise-Lame
	Cale sèche
	Entrave à la navigation
	Mur de protection (seawall)
	Repère de navigation
	Vivier / pisciculture
	Échelle à poisson
Exploitation agricole	Champ - grandes cultures
	Champ - cultures maraîchères
	Champ - horticulture
	Champ - fourrager
	Friche
	Verger
	Vignoble
	Enclos
Batiments agricoles	Grange, entrepôt, hangar
	Silo
	Ferme d'élevage
	Fosse lisier
	Serre d'exploitation agricole
	Serre commerciale
	Élévateur à grain
Arbres (en divers milieux)	Arbre feuillus (Deciduous tree)
	Arbre conifère (Evergreen Forest Land)
Hydrographie linéaire	Rivière (Streams)
	Ruisseau
	Fossé
	Chute
	Rapide
	Cours d'eau tari
Hydrographie surfacique	Eau
Milieux humides	Bog - arboré
	Bog - non arboré
	Fen
	Marécage
	Marais
	Eau peu profonde
Mines	Pit
	Mine shaft
	Tailings
	Bassin de décantation

xView

Classe mère	Classe fille
Fixed-Wing Aircraft	Small Aircraft
	Cargo Plane
Passenger Vehicle	Small Car
	Bus
Building	Hut/Tent
	Shed
	Aircraft Hangar
	Damaged Building
	Facility
Truck	Pickup Truck
	Utility Truck
	Cargo Truck
	Truck w/Box
	Truck Tractor Trailer
	Truck w/Flatbed
	Truck w/Liquid
Railway Vehicle	Passenger Car
	Cargo Car
	Flat Car
	Tank Car
	Locomotive
Maritime Vessel	Motoboat
	Sailboat
	Tugboat
	Barge
	Fishing Vessel
	Ferry
	Yacht
	Container Ship
	Oil Tanker
Engineering Vessel	Tower Crane
	Container Crane
	Reach Stacker
	Straddle Carrier
	Mobile Crane
	Dump Truck
	Haul Truck
	Scrapper/Tractor
	Front Loader
	Excavator
	Cement Mixer
	Ground Grader
	Crane Truck
None	Helipad
	Pylon
	Shipping Container
	Shipping Container Lot
	Storage Tank
	Vehicle Lot
	Construction Site
	Tower Structure
	Helicopter

Annexe 3 : Processus d'entraînement des diverses approches considérées