MASTER 1 INFORMATIQUE - STAGE DIGITAG

Segmentation automatique d'images aériennes de systèmes agroforestiers pour caractériser leur structure

Hugo Maitre

Encadrants:
Frédéric BORNE
Marc JAEGER
Marc CHAUMONT

30 août 2022

Remerciements

Ce travail a bénéficié d'une aide de l'État gérée par l'Agence Nationale de la Recherche au titre du programme d'Investissements d'Avenir portant la référence ANR-16-CONV-0004 (DigitAg)

Table des matières

1 Contexte général

Dans les multiples défis auxquels sont confrontées aujourd'hui les sociétés, l'agriculture occupe une place déterminante. Le modèle intensif n'est plus d'actualité mais requiert des évolutions importantes. La notion de production est petit à petit remplacée par celles de services écosystémiques de systèmes écologiques couvrant à la fois des aspects de production, de conservation, d'entretien et de mise en valeur des paysages. En effet, les services écosystémiques que remplissent les systèmes agro-agroécologiques sont aujourd'hui largement reconnus comme la biodiversité et le stockage de carbone, mais sont complexes à comprendre, concevoir et développer. Ils sont au cœur des travaux de recherche du Cirad. Dans ce stage nous présenterons différentes méthodes de segmentation d'images satellitaires qui fourniront en sortie une carte de d'occupation des sols. L'accent est mis sur l'agroforesterie en choisissant au préalable assez de classes caractérisant chaque ensemble agroforestier qu'il est possible de rencontrer sur les données satellites mises à disposition.

1.1 Le Cirad

Le Cirad est un établissement public à intérêt commercial (EPIC). Cette organisme de recherche de 1650 salariés, dont 1 140 scientifiques, est présente dans plus de cinquante pays et opère avec un réseau mondial d'environ 200 partenaires. Le Cirad a défini six thématiques pour orienter sa recherche finalisée (Approches territoriales, Biodiversité, Changement climatique, Systèmes alimentaires, Transitions agroécologiques, Une seule santé) Ses deux principaux centres de recherche sont dans l'agglomération montpelliéraine, sur les campus de Lavalette, et de Baillarguet. Douze directions régionales réparties sur l'ensemble des continents, en particulier sur les territoires outre-mer français

permettent une coopération active au Sud avec une centaine de pays. Le Cirad comprend 29 unités de recherche réparties dans trois départements scientifiques : Systèmes biologiques (Bios), Performances des systèmes de production et de transformation tropicaux (Persyst) et Environnements et sociétés (ES).

1.2 UMR AMAP

Le stage se déroule au sein de l'UMR AMAP, du département BIOS qui recouvre des problématiques de recherche plutôt méthodologiques. L'UMR AMAP compte environ 90 permanents et 120 personnes non-permanentes salariés du CIRAD, de l'IRD, de l'Inrae, du CNRS et de l'Université de Montpellier. La recherche y est organisée selon trois axes : deux axes thématiques que sont Biodiversité et Biomasse, et le dernier axe est un axe méthodologique : Plantes numériques. Chaque axe est composé de 3 à 4 thèmes (équipes projets de recherches). L'axe Plantes Numériques comprend les thèmes d'imagerie et Machine Learning (BIAS, thème dans lequel ce stage s'inscrit), Modèles structure-fonction pour les plantes et agrosystèmes (FSPM), et développements mathématiques et statistiques (MAGNET). L'UMR AMAP est reconnue comme étant pionnière dans la modélisation 3D de végétaux, avec des approches génériques de description de plantes (la botanique architecturale, Barthélémy et al, 2007), de modélisation de leur structure (Reffye et al, 1988, Jaeger et al, 1992) jusqu'à la simulation de leur fonctionnement (Reffye et al, 2016). Elle dispose donc de multiples outils permettant de calculer et visualiser la structure de plantes 3D. L'unité est aussi pionnière de l'usage du big data et des réseaux de convolution pour l'identification d'espèce végétale via le projet et l'application Pl@ntNet (Joly et al., 2016) avec plus de 5 millions d'utilisateurs et près d'un demi-milliard de requêtes. Elle possède aussi l'expérience de l'analyse d'imagerie aérienne avec des méthodes supervisées ou par apprentissage (Borne, Viennois, Kennel et al.,

2017) L'UMR développe de nombreux projets de recherche au sein de ses trois axes, et est particulièrement impliquée dans des projets disposant d'un financement de l'Institut de convergence DigitAg qui promeut des actions autour du numérique et de l'agriculture, en particulier sur la caractérisation et le design de nouveaux systèmes agro-écologiques. C'est dans ce cadre que ce stage est financé.

1.3 Contexte et sujet de stage

On se propose ici d'appréhender la structure des systèmes agro-écologiques à partir de l'analyse d'images aériennes. Cependant les méthodes traditionnelles de segmentation de segmentation – localisation des ressources végétales – et de classification – identification fine de ces ressources pouvant aller a l'echelle de l'individu arbre - n'ont jusqu'à présent répondu que partiellement à ce besoin de caractérisation fine des systèmes. Elles demandent un lourd travail d'expertise, de surcroit difficile à généraliser d'une image à l'autre. C'est pourquoi il est proposé d'évaluer la faisabilité de la classification automatique par réseaux de neurones convolutionnels avec les derniers développements méthodologiques du domaine et une comparaison avec une approche supervisée de télédétection. A cette fin, une classification automatique par Deep Learning d'images aériennes de paysage comprenant des systèmes ruraux dans un environnement dédié sera sélectionnée et implémentée, puis comparée avec une classification traditionnelle sur des exemples ciblés. La cartographie fine automatisée de systèmes agro écologiques permet de multiples développements de recherches et d'application couvrant, parmi lesquels on peut citer, de manière non exclusive : L'aide aux inventaires, en particulier dans le suivi d'expérimentations à différentes échelles pour la gestion des territoires agricoles.
La méthodologie permet également d'aborder la quantification des espaces dé-

diés au pastoralisme dans un milieu ouvert.
Les données mobilisées dans les jeux d'apprentissage et de tests s'appuient sur des ressources existantes et à acquérir dans des projets en cours (au Nord comme au Sud).
D'un point de vue méthodologique, l'approche proposée, partant d'images de systèmes relativement simples vers des systèmes de plus en plus ouverts, à structure complexe, est une voie intéressante pour appréhender les difficultés et tester les capacités de Transfer Learning. Enfin, les cartographies générées permettent d'alimenter divers modèles spatialisés, descriptifs ou dynamiques, permettant par exemple de calculer des grandeurs caractéristiques des exploitations sous-jacentes, biophysiques ou économiques.

2 Données et Mise en Place

2.1 Données

Nous avons à notre disposition 4096 tuiles d'une grande image satellite couvrant un carré de 32 km de côté. Chaque tuile est une image carrée de 1024*1024 pixels, ce qui nous fait donc une résolution de 1 pixel pour 50 cm. La résolution est bonne mais pas assez si l'on veut rentrer plus dans le détail et faire une segmentation très fine d'individus de parcelles agricoles par exemple. Ainsi la segmentation effectuée se concentre sur des ensembles forestiers et est assez grossière. La qualité des images est correcte, leur netteté est très homogène, en revanche leur colorimétrie souffre de grosses variations, certaines images sont trop bleutées. Par ailleurs certaines tuiles ont été acquises à la fin de l'hiver d'où la présence de neiges ce qui va grandement impacter la qualité de la segmentation. On s'est donc assurés de ne pas inclure ces tuiles dont notre jeu d'apprentissage et de validation.

Figure 1 – Echantillon du Dataset

2.2 Annotations

Une longue partie du stage a été consacrée à la constitution d'un jeu d'apprentissage. L'IGN offre beaucoup de jeux de données de couvertures des sols, c'est d'ailleurs ce sur quoi s'est appuyé le travail de segmentation précédent, effectué en 2021. Le jeu d'apprentissage composé avait alors 6 classes de couvertures de sols possibles et utilisait la vérité-terrain de l'IGN. Toutefois pour qualifier plus précisément les systèmes agroforestiers, nous avons introduit plus de classes caractéristiques. On fait désormais la distinction entre un champ, un verger ou une lande herbeuse, et entre une forêt claire et des buissons. La volonté d'avoir nos propres classes et un meilleur contrôle de notre jeu d'apprentissage a aussi motivé le choix de faire les annotations nous-mêmes.

Pour effectuer ces annotations nous avons utilisé un logiciel en ligne apeer, qui rend la tâche plus aisée. Le jeu de données comporte donc 11 classes et 1 classe background qui rassemble toutes les zones de transition (les haies, talus

de bord de route) et les zones indéterminées:

— Background;

— buisson;

— champs;

— eau;

— falaise;

— foret dense;

— foret claire;

— pierrier;

— prairie;

— route;

— verger;

— batîment;

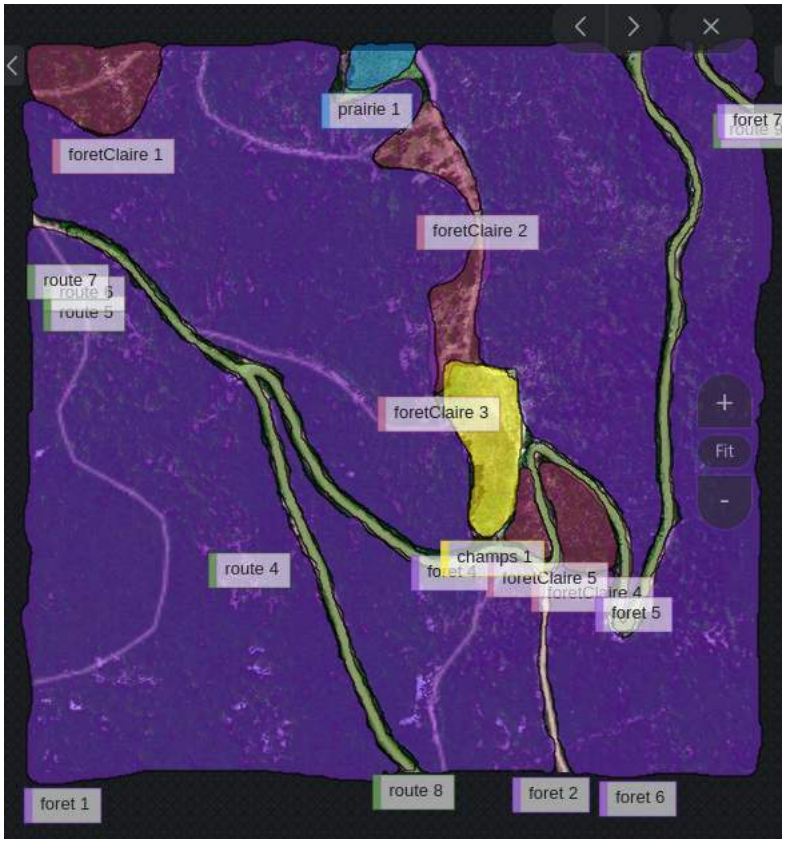


Figure 2 – Visualisation d'une image labelisée sous Apeer

Pour plus de facilité nous avons choisi comme zone de travail une zone rectangulaire incluse dans la grande image de la zone globale. Cela nous a permis de repérer plus facilement la position de chaque tuile puisque celles-ci n'étaient pas géoréférencées. Nous avons pu utiliser aussi un outil comme google earth pour vérifier directement sur le logiciel la pertinence de nos annotations. En effet on pouvait assez facilement repérer la position sur google earth de chaque tuile et vérifier à l'aide d'images satellites plus précises ou du street view la composition des paysages. Cette zone a aussi la particularité d'inclure l'ensemble des paysages possible: à la fois de la forêt, des plaines agricoles,de la montagne... toutes nos classes y sont bien représentées.

On peut voir ci-dessous en rouge la zone globale d'étude et en violet la zone restreinte sur laquelle nous nous sommes concentrés.

Figure 3 – Zone de labellisation en violet et zone totale du jeu Ampli en rouge

La zone de labellisation violette comprend environ 300 images, parmi lesquelles seulement 70 images ont été labellisées pour des soucis de temps. On peut voir ci-dessous le rectangle recomposé en image satellite (image du haut) et en zone labellisée (image du bas).

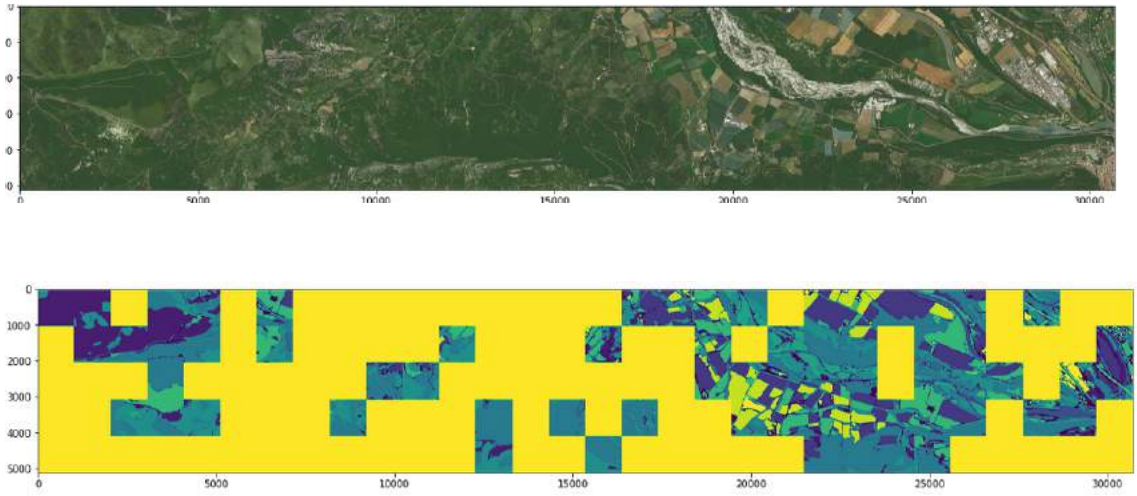


Figure 4 – Images recomposées en haut et cartographie recomposée en bas

On peut voir ci-dessus un zoom sur le rectangle violet où nous nous sommes restreint pour les annotations. Sur les 220 tuiles qui composent le rectangle environ 2/3 des images n'ont pas été annoté (parties jaunes sur l'image du bas) et le reste l'a été.

Nous avons corrigé le jeu d'annotations à deux reprises avec un intervalle de 45 jours mais par la même personne. Le premier jeu a servi à quelques entraînements, le deuxième jeu est une correction du premier, les annotations sont plus fines et comportent moins d'oublis.

Obtenir des annotations de qualité et en grande quantité s'est révélé être un des plus gros challenges de ce stage. Leur qualité est discutable puisque sujette au biais de ma propre perception des zones sur l'image. N'étant ni un expert du terrain à labelliser, ni un expert en agroforesterie, il était souvent difficile de distinguer une zone de forêt claire d'une zone de forêt dense ou bien encore d'un champ de blé au printemps (encore vert) d'une prairie... On a donc volontairement sélectionné 70 images où la confusion était la moins présente. En plus des problèmes d'interprétations les annotations sont souvent grossières ou com-

portent des oublis, cf ci dessous nous avons oublié la route (image du centre) ce qui pénalise à tort le réseau qui prédit la zone correctement (image de gauche). Ce qui manque à notre travail est certainement un processus plus rigoureux d'évaluation des erreurs générées lors des annotations. Il aurait aussi été intéressant de constituer le même jeu d'annotations par différentes personnes.

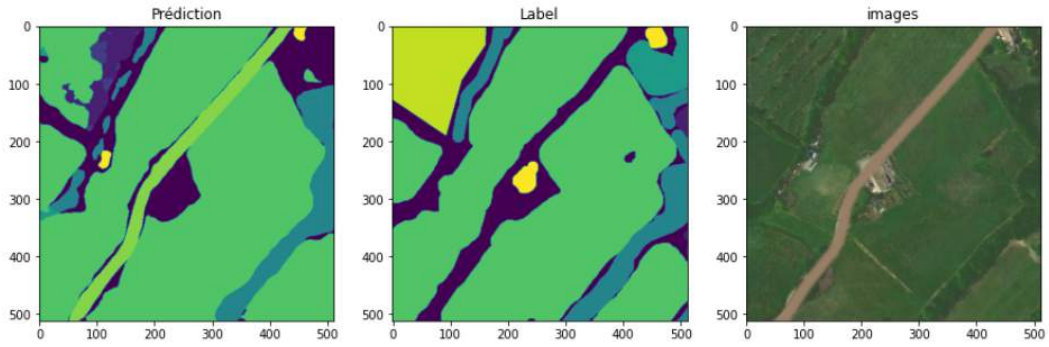


Figure 5 – Image incorrectement labelisée

2.3 Choix techniques

2.3.1 Python et ses librairies

Parmi les différentes options qui s'offraient à nous, le langage de programmation Python semblait de loin le plus adapté pour les tâches de Computer Vision. Le langage dispose d'une immense communauté surtout dans tous les domaines scientifiques. Dans le domaine du Deep Learning appliqué à l'image dans le domaine du deep learning appliqué à l'image, il propose des solutions rapides et faciles à mettre en place.

Parmi les librairies souvent utilisées j'en citerai quelques-unes. Opencv : pour toutes les tâches de manipulations d'images, crop rotation et contraste ainsi que des opérations plus complexes (homographie, corrections d'histogramme).

Numpy : Librairie permettant la manipulation plus rapide et performante de

tableaux (et donc d'images) par le processus de "vectorisation".
Matplotlib : Utile pour la visualisation de données, pour tracer des graphiques etc...
Enfin Tensorflow et Torch librairie utile pour la mise en place des réseaux de neuronnes.

2.3.2 Accès SSH

Étant limité par la carte graphique du poste principal (deepZ800), les entraînements s'effectuent avec des batch size de 2 ou 4, ce qui nuisait beaucoup aux performances des modèles entraînés. Après quelques apprentissages infructueux (temps trop long avant la convergence ...),nous avons très vite configuré avec l'aide de Frédéric Théveny un accès à un serveur de calcul AMAP équipé d'une GPU performante. Le GPU auquel nous avons accès est une Quadro RTX 6000 avec 24GB de mémoire. L'accès se fait via ssh depuis mon poste de travail et nous permet de lancer de très gros calculs. Nous avons utilisé Jupyter pour l'ensemble de la programmation. Pour les entraînements effectués par ssh il est donc important de lancer Jupyter via les commandes ssh. Pour ce faire il faut ouvrir 2 terminal et taper respectivement les commandes suivantes.

```
ssh hugo@hugobis.local
source <PATH TO ENV>
jupyter notebook --no-browser --port=8070
```

```
ssh -L 8080:localhost:8070 hugo@hugobis.local
# and go to http://localhost:8080
```

2.3.3 Environnement Python

Pour chaque sous-projet, nous refaisons un environnement python ce qui nous permettait de mieux gérer les problèmes de compatibilité entre versions python et d'éviter les bugs récurrents lorsque plusieurs versions de la même librairie cohabitent sur un même système. Avant chaque lancement de jupyter il est donc nécessaire d'activer l'environnement et ensuite d'installer les librairies nécessaires.

2.4 Préparation du jeu de données

La préparation du jeu de données suit globalement le même schéma pour tous les entraînements. Il y'a cependant deux architectures différentes pour les dossiers à utiliser en fonction des réseaux.

Architecture 1 à utiliser pour tous les réseaux sous Torch.

```
datasetName/
    images/
        img1.png
        img2.png
        ...
    annotations/
        img1.png
        img2.png
        ...
```

Architecture 2 à utiliser pour Unet dans son implémentation en Tensorflow ainsi que pour les GAN.

```
datasetName/
```

```
images/
    training/
        img1.png
        img2.png
        img4.png
        ...
    validation/
        img3.png
        ...
annotations/
    training/
        img1.png
        img2.png
        img4.png
        ...
    validation/
        img3.png
        ...
```

Tous nos réseaux ont eu en entrée des images de 512*512 pixels donc 1/4 d'une tuile de base. En effet nous voulions que les images ne soient pas redimensionées et qu'elle soit assez grosse pour conserver le contexte environnant les classes. On a donc divisé chacune des 73 tuiles de base en 4 petites imagettes de 512*512 pixels. Les annotations prennent la forme d'images raster.

3 Développement, Méthodes et Métriques

3.1 Première Implementation avec Tensorflow

Une première implémentation d'un réseau de segmentation Unet a été effectué le premier mois du stage. L'implémentation s'est faite via tensorflow (sous dossier Satellite-Image-Segmentation-tensorFlow dans le repo github). Voir le README. Dans le but de poursuivre l'analyse à d'autres réseaux et d'utiliser les GPU nous sommes ensuite passer à l'outil mmsegmentation.

3.2 Implémentation sous Torch

Pour la suite du stage nous utilisions mmsegmentation un outil proposant l'implémentation d'une grande quantité de réseaux de segmentations d'images qui permet facilement de réaliser une benchmark de plusieurs réseaux. En revanche pour un but d'intégration sur une application cet outil est totalement inadapté. Voir le README et le README officiel.

3.3 Discussion sur les Métriques

Une tâche conséquente de ce stage à porté sur les mesures pour évaluer correctement les sorties d'un réseau. Nous avons choisis une comparaison pixel à pixel et non par objet. Un pixel est bien classé si sa classe prédite est la même que la classe du pixel correspondant dans l'annotation. Nous avons établi un pipeline de différents niveaux dd'analyse de nos indicateurs: du plus en détail au plus général. Dans un premier temps nous effectuons une analyse par matrice de confusion de 12*12 cases ce qui nous donne pour chaque classe des informations sur le nombre de faux positifs, faux négatifs. En normalisant la matrice sur les lignes (resp colonnes) nous avons une vue plus précise sur la

répartition des faux positifs (resp faux négatifs) et des confusions entre classes. Attention la matrice des faux positifs (resp faux négatifs) se lit uniquement par lignes (resp colonnes), les valeurs de 2 lignes (resp colonnes) différentes ne sont pas comprables. Nous obtenons ainsi des informations sur les classes qui sont souvent confondues ou au contraire bien différenciées. Ensuite nous regardons le f1-score, precision recall pour chaque classe, qui nous apportent un score général sur chaque classe.

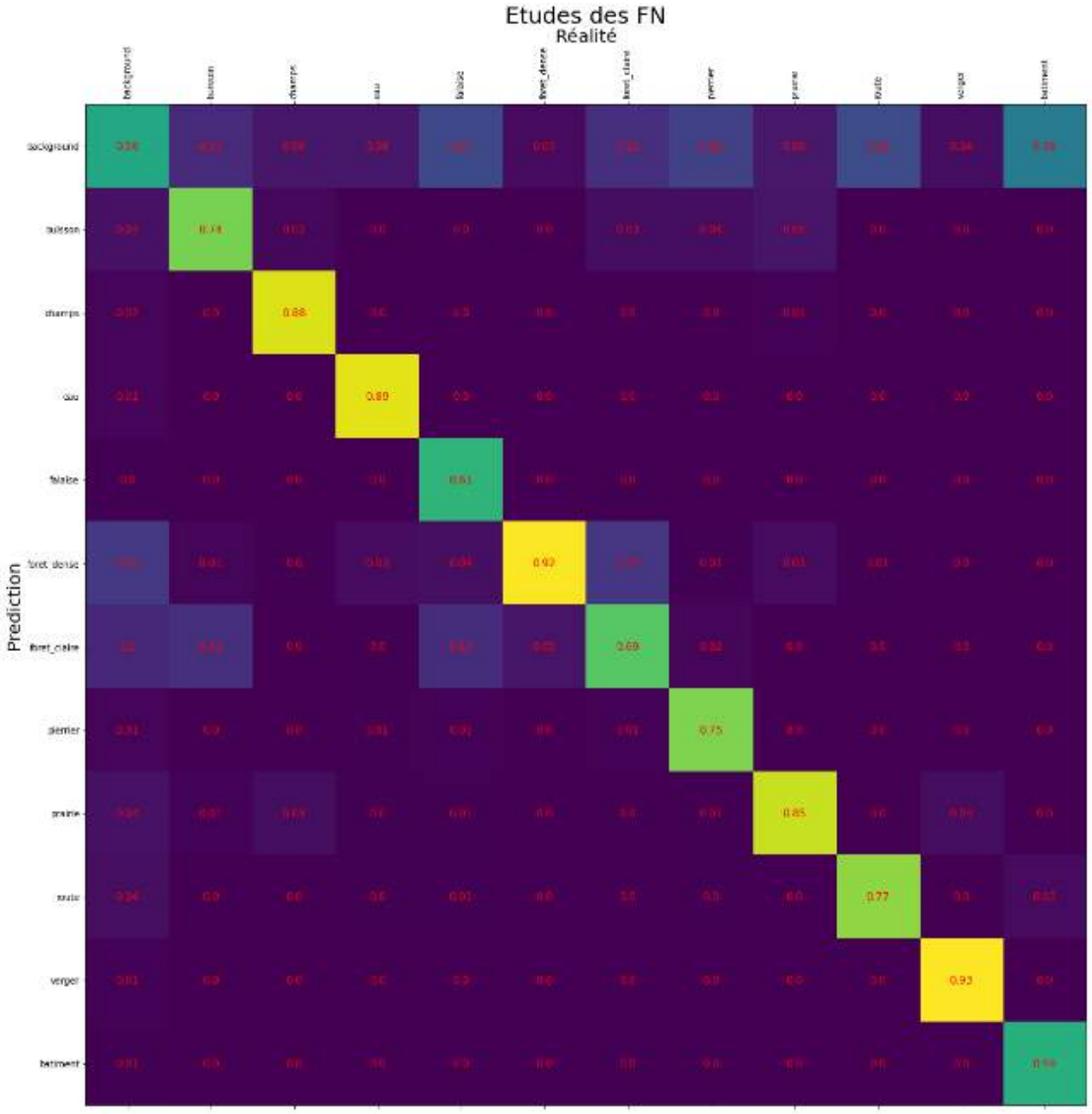


Figure 6 – Matrice de confusion normalisé sur les colonnes (pour l'étude des FN)

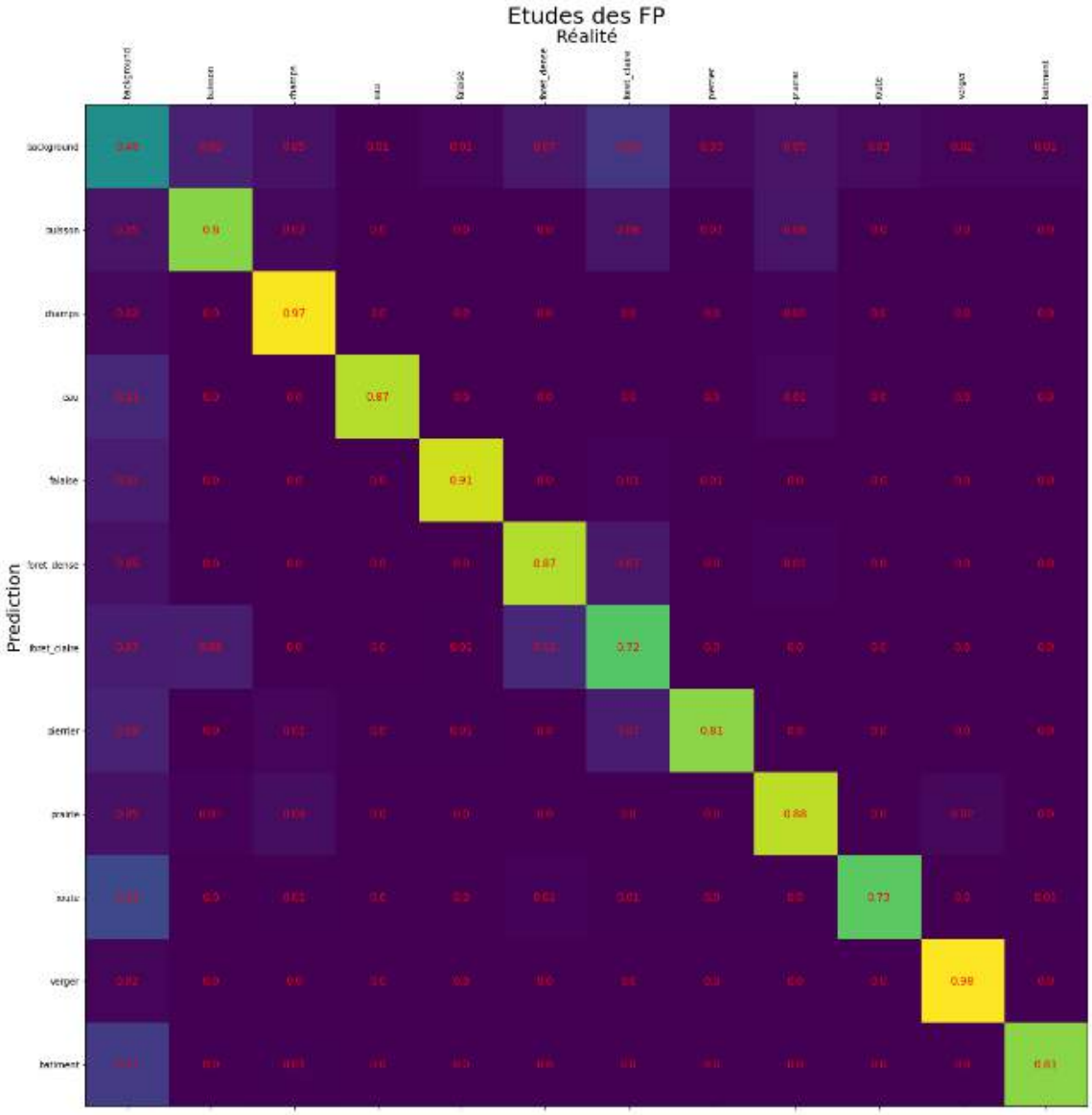


Figure 7 – Matrice de confusion sur les lignes (pour l'étude des FP)

En dernier on calcule des métriques classiques en segmentation, l'IoU moyen et l'accuracy moyenne qui nous donnent une performance globale du réseau sur notre jeu de données.

4 Préparation du jeu de données artificiel

4.1 Interet d'un jeu de données artificiel

Un des problèmes relevés précédemment est la difficulté d'avoir une quantité importante de paires d'images brutes et leurs annotations correctes. En effet nous avons en notre possession une faible quantité d'annotations et celles-ci manquent de précision. Pour pallier ce problème nous avons cherché un moyen de créer des images artificielles. Jusqu'à maintenant nous partions d'une image satellite et créions l'annotation à la main en assignant chaque zone caractéristique d'une image à une classe. L'idée est de faire l'inverse, partir d'une annotation (cartographie avec 11 zones possibles pour chaque classe excepté la classe background) et fabriquer une image à partir de cette carte. Pour cela nous avons besoin d'images comportant une classe unique c'est à dire des images avec seulement des champs, de la forêt... Les images comportant une classe unique sont crées en assemblant plein d'images de texture de notre jeu de donnée de manière à former une tuile complète comportant une classe unique. Nous avons aussi besoin de cartes, pour cela nous prenons les cartes dans la base de données ORTHO Ign en récupérant la carte de labels de zones issus des alpes maritimes et de Loire Atlantique. Les cartes comportent 15 classes, d'où la nécessité de fusionner quelques zones pour créer des cartes à 11 classes. Nous avons ainsi accès à des cartes réalistes de zones, reste maintenant à les remplir avec nos images de classes uniques. On s'assure aussi que la proportion de chaque classe dans la base des cartes respecte environ les proportions réelles du jeu de données AMPLI.

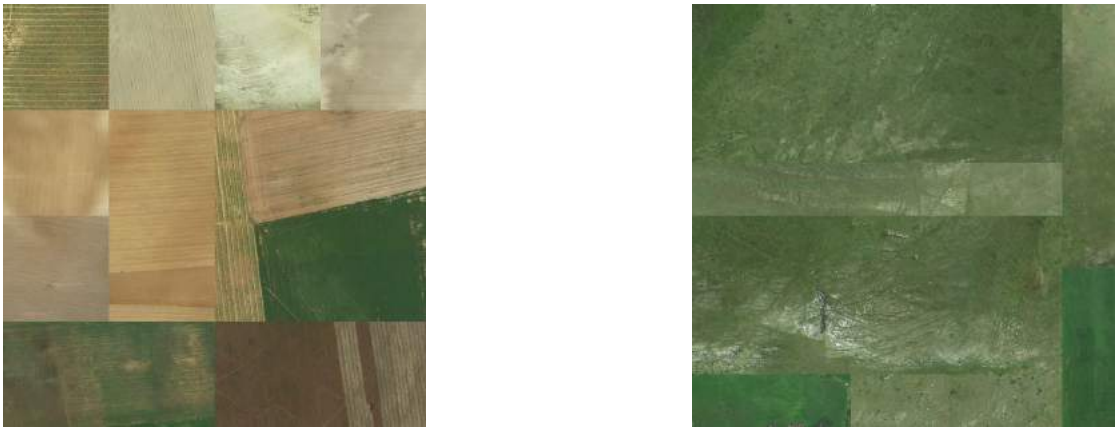


Figure 8 – Processus de création d'une image artificielle

On constitue ainsi une image en assignant à chaque zone de la cartographie des portions de l'image comportant une classe unique de la classe de la zone correspondante. De cette manière chaque pixel sera assigné à une des 11 classes, on s'assure ainsi d'avoir une labélisation précise au pixel près. On peut aussi générer une grande quantité d'images à partir de différentes annotations rapidement à condition d'avoir une grande variété d'images de classes uniques pour avoir une grande variété de contenus dans les images.

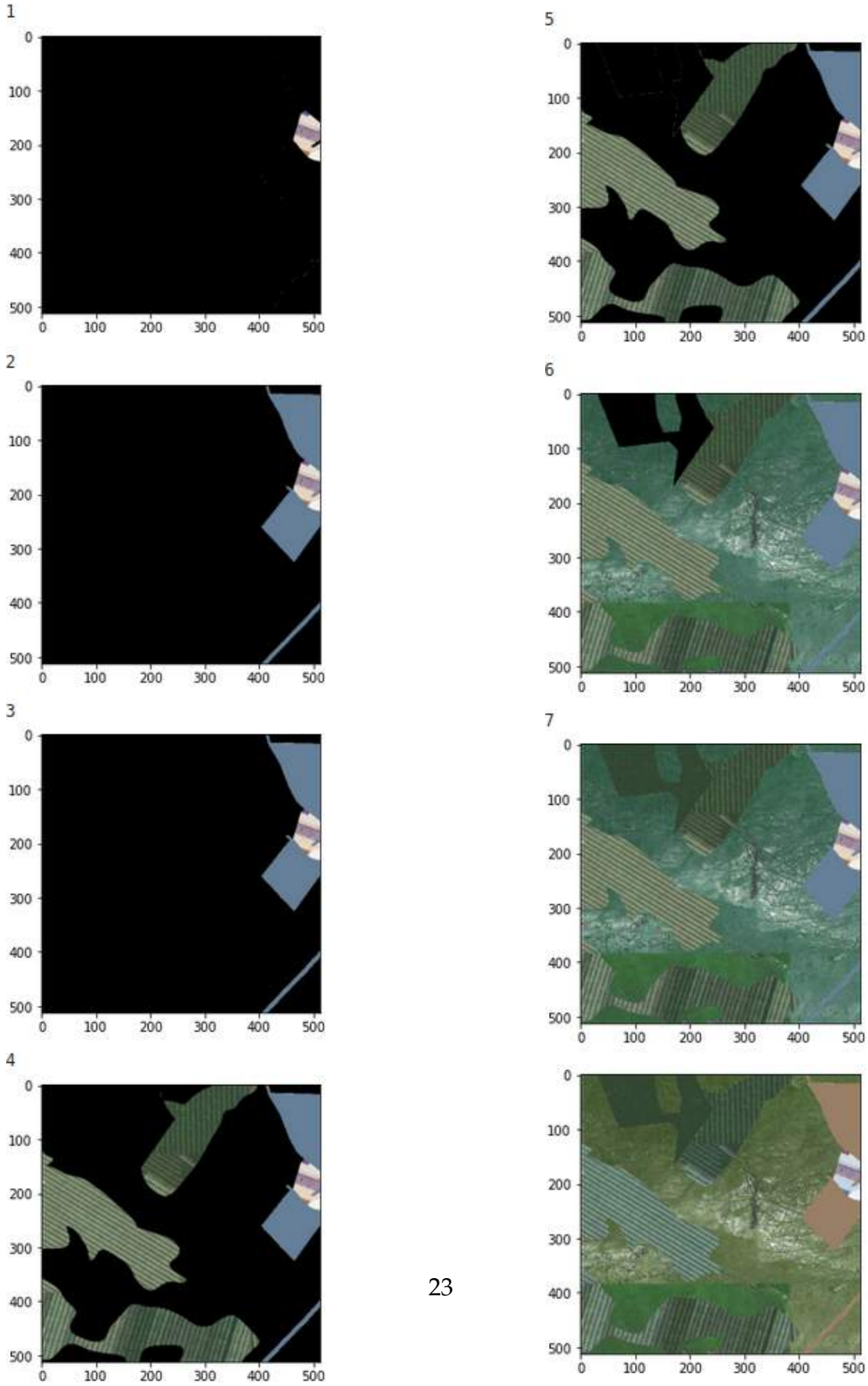


Figure 9 – Processus de création d'une image artificielle

4.2 Limite du premier jeu de données artificiel

Nous avons ensuite réalisé quelques expériences pour voir comment un réseau de segmentation réagissait sur notre jeu de données artificiel. L'entraînement s'est fait sur un réseau d'architecture type Unet. [1].

Entraînement sur \ Test su	Jeu Réelle (VDS=64)			Jeu Artif(VDS=102)		
Jeu Réelle(TDS=204)	0,79	0,78	0,79	0,78	0,74	0,75
Jeu Artif(TDS=412)	0,74	0,65	0,66	0,98	0,97	0,97

Mean-F1
Mean-Recall
Mean-Precision

Figure 10 – Résultats des expériences avec Unet sur jeu réel + artificiel

Nous avons donc réalisé deux entraînements un sur le jeu réel et un sur le jeu de données artificiel. L'entraînement sur le jeu réel donne de bons résultats à la fois sur le jeu de validation réel et le jeu de validation artificiel. En revanche le modèle entrainé sur le jeu de données artificiel lui généralise très mal sur des données réelles. En effet il performe très bien sur son jeu de validation mais assez mal sur le jeu réel. Nous avons pu identifier 2 possibles causes à la mauvaise généralisation de ce dernier modèle. Les images artificielles sont générées par assemblage d'images réelles sans transition entre classes, les images comportent des ruptures très fortes ce qui est très peu réaliste et crée une forte hétérogénéité sur le jeu de données. L'autre problème peut venir de la faible variété d'images de classes uniques. Le même contenu peut apparaître plusieurs fois même si la forme de la zone est différente (voir l'exemple dans la paire d'images ci des-

1. partie implémentation sur Tensorflow du repository

sous).

Figure 11 – Répétition des textures sur les images artificielles

4.3 Amélioration du jeu de données artificiel

La seconde version du jeu de données artificiel apporte deux corrections majeures au premier jeu. Dans un premier temps nous avons lissé les bords entre chaque classe afin de faire une transition qui paraît plus naturelle pour le réseau. Puis dans un second temps nous avons grandement enrichi la base d'images comportant une classe unique pour éviter la répétition de contenus. Pour atténuer les ruptures on a appliqué une série de kernels. Dans un premier temps, un kernel de détection de contours produit une image de contours.

$$\text{Kernel\_detection\_contour} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Lorsque les contours sont détectés on applique ensuite une dilatation sur l'image des contours. De cette manière on peut créer une image avec des contours plus ou moins larges. .

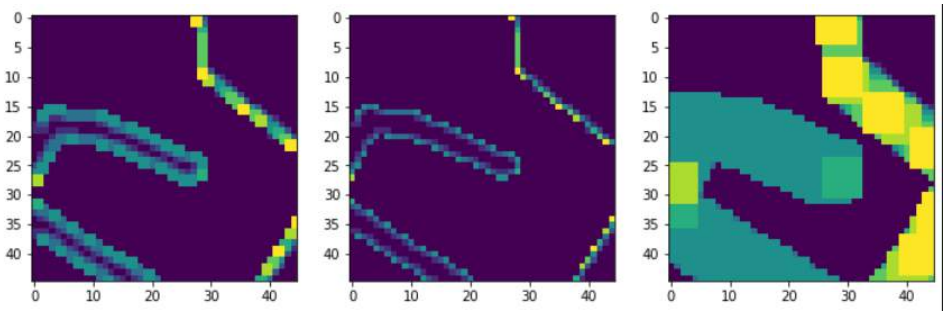
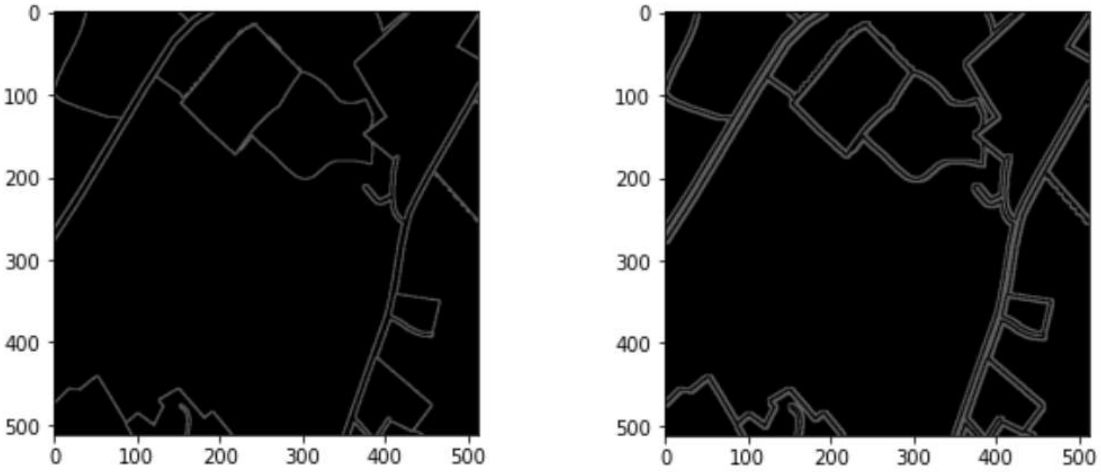


Figure 12 – Effet de la dilatation: image de contour au centre, dilation forte à droite, moins forte à gauche

On calcule ensuite le masque de dilatation obtenu en calculant l'image binaire associée à l'image obtenue par dilatation de l'image de contours: 1 pour toutes valeurs supérieures à 0 et 0 sinon.
On applique ensuite un floutage sur l'image artificielle sur les parties non nulle du masque obtenu. A noter que l'on calcule deux masques pour des valeurs de dilatations différentes, un masque de dilatation faible et un de dilatation forte. De cette manière on peut appliquer des floutages plus fort sur le masque de dilatation faible (aire très proche de la rupture de classe) et un floutage plus léger (aire plus éloigné de la rupture de classe).



$$Kernel\_floutage\_fort = \frac{1}{256} \times \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

$$Kernel\_floutage\_leger = \frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Figure 14 – Images avec ruptures non floutées à gauche floutés à droite

En plus du floutage des ruptures nous avons enrichi les images de classes uniques en allant chercher des tuiles de même résolution sur google earth. De plus, lors de chaque assignation d'une zone de la cartographie une rotation aléatoire est appliquée. On s'assure ainsi de n'avoir quasiment jamais de répétition de structure.

5 Préparation du jeu GAN

De manière à enrichir d'une meilleure façon notre jeu de donnée nous souhaiterions utiliser un réseau de neurones générant des images satellite à partir de carte de label.De cette manière on pourrait augmenter notre jeu de données de manière très précise comme avec le jeu de données artificiel mais aussi de manière plus réaliste. Nous avons décidé d'utiliser un GAN qui générerait de manière très réaliste une image satellite à partir d'une carte de label et d'un tenseur de bruit en entrée.

Notre choix s'est porté sur le réseau OASIS dont l'architecture est décrite ci dessous.

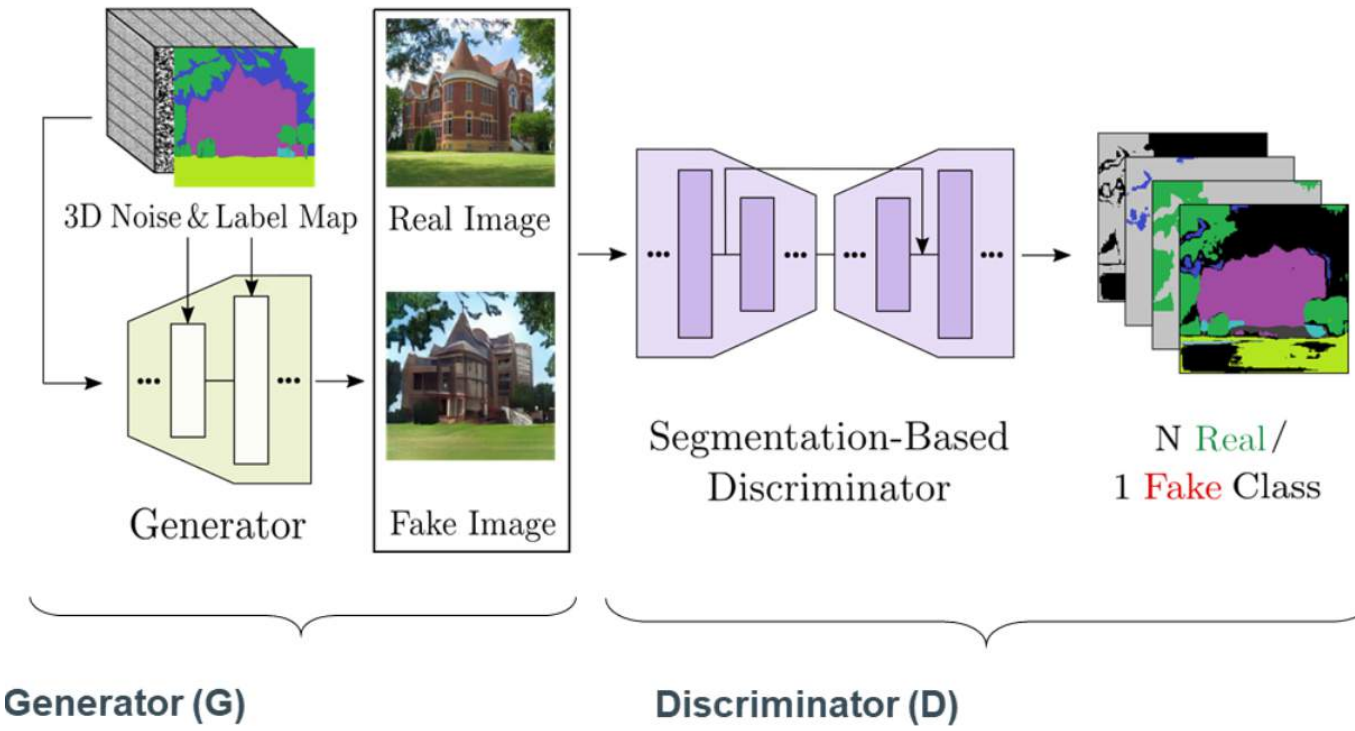


Figure 15 – Architecture du réseau OASIS source: github OASIS

Ce réseau est récent et assez bien documenté ce qui facilite pour nous son

utilisation et répond parfaitement au problème posé.

Le générateur se base sur des architectures précédentes de GAN accomplissant les mêmes types de tâches. À noter que le générateur a seulement en entrée la carte des labels et des tenseurs de bruits. Habituellement la vraie image est aussi donnée en entrée. La vraie nouveauté ici provient du discriminateur. En effet le réseau se dote d'un réseau de segmentation de type UNet en guise de discriminateur avec N + 1 classes où N correspond au nombre de classes possibles et 1 pour désigner la classe Fake class. Donc si une fausse image est donnée en entrée, le discriminateur essaie de segmenter cette image entière en Fake class. Si une vraie image est passée en entrée, alors le discriminateur essaie de faire une segmentation classique.

L'apprentissage est supervisé puisque notre jeu pour entraîner le GAN est le jeu d'entraînement classique utilisé depuis le départ. Les annotations sont données en entrée au générateur et les images associées, en entrée au discriminateur.

Puisque nous étions limités en temps et en puissance matériel, l'apprentissage par GAN ne s'est pas effectué dans des conditions optimales. Les créateurs d'OASIS recommandent l'utilisation de 4 GPU de 32 GB. Nous possédons donc environ un cinquième de la puissance recommandée. De plus, les durée d'entraînements d'un GAN sont souvent très longues. Pour OASIS il faut compter environ cinq à dix jours d'entraînements. Nous n'avions pas tout ce temps à notre disposition, nous avons donc dû resizer nos images, se contenter de petit batch size et stopper l'apprentissage alors que celui-ci aurait certainement pus être encore prolongé.

Method	year	val FID	crop size	Batch Size
OASIS	2021	167	(256,256)	12

Une bonne pratique avec les GAN lorsque le temps et les ressources matérielles sont limitées est de réduire drastiquement la résolution des images en entrée pour augmenter le batch size. De cette manière on diminue grandement le temps d'apprentissage.

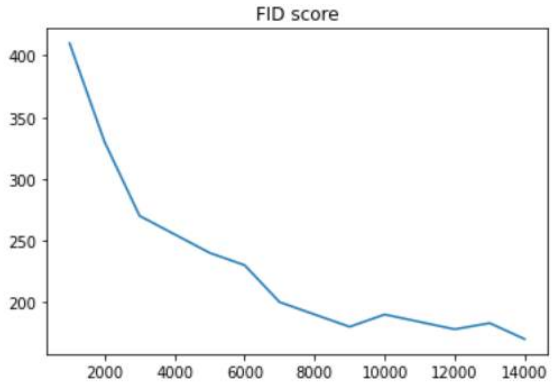


Figure 16 – Score FID durant l'apprentissage

Le temps d'apprentissage d'un GAN est extrêmement long, ainsi nous avons laissé l'algorithme tourné plusieurs jours jusqu'à obtenir une stabilisation du score FID. Parallèlement nous avons pu entraîner le réseau sur nos images redimensionnées sur du 64*64 pixels avec un batch size de 128. Dans ces conditions les résultats étaient bien meilleurs. On observe très rapidement une convergence du score FID sous les 50 en moins de 24h. Il n'est évidemment pas possible pour nous de descendre à une telle résolution pour la génération de nos images, mais il existe des méthodes par GAN pour augmenter à nouveau la taille de l'image générée, ceci pourra être vu comme une piste d'amélioration à ce stage. [2]

2. Fréchet inception distance

Figure 17 – Images générées au bout de 3000 epochs

Figure 18 – Images générées au bout de 15000 epochs

Les images paraissent légèrement plus contrastées au bout de 2000 epochs mais présentent de gros défauts dans les textures. On observe la présence de diagonales parallèles dans les champs d'une mosaïque de formes arrondis dans la forêt... Au bout de 15000 epochs les images sont moins contrastées mais sont plus réalistes surtout sur les textures de forêt, de champs et de verger.

6 Méthodes classiques

Nous devions aussi réaliser une classification au pixel par approche classique, la méthode retenue est basée sur un maximum de vraisemblance. La classification a été réalisé à l'aide du logiciel ArcGis, cependant nous n'avons pu effectuer l'analyse des résultats pour plusieurs raisons:

— La palette des couleurs de la tuile annotation(vérité terrain) et de la tuile classification (la prédiction) lors de l'export sous ArcGis, ne correspondent pas et cette information est manquante. Il est impossible de savoir à quel numéro correspond chaque classe.

— Lors de l'export de la tuile annotation sous ArcGis nous avons souffert d'un problème de résolution d'images. L'image exportée était en effet d'une définition bien moins précise que la tuile de classification. Or nous avons besoin que ces 2 tuiles soit exactement de la même définition (même nombre de pixels).

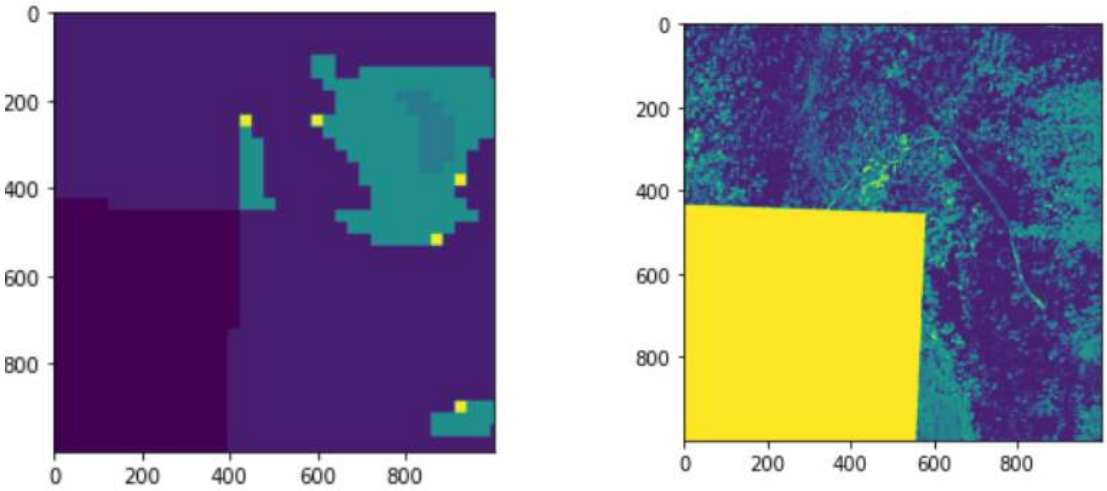


Figure 19 – Zoom sur la différence de résolutions entres la tuile annotation (à gauche) et la tuile prédiction (à droite)

— Le géoréférencement des tuiles est inexact comme on peut le voir sur les

images ci-dessus on observe un décalage entre la tuile annotation et la tuile classification.

7 Résultats

7.1 Résultats sur les classes

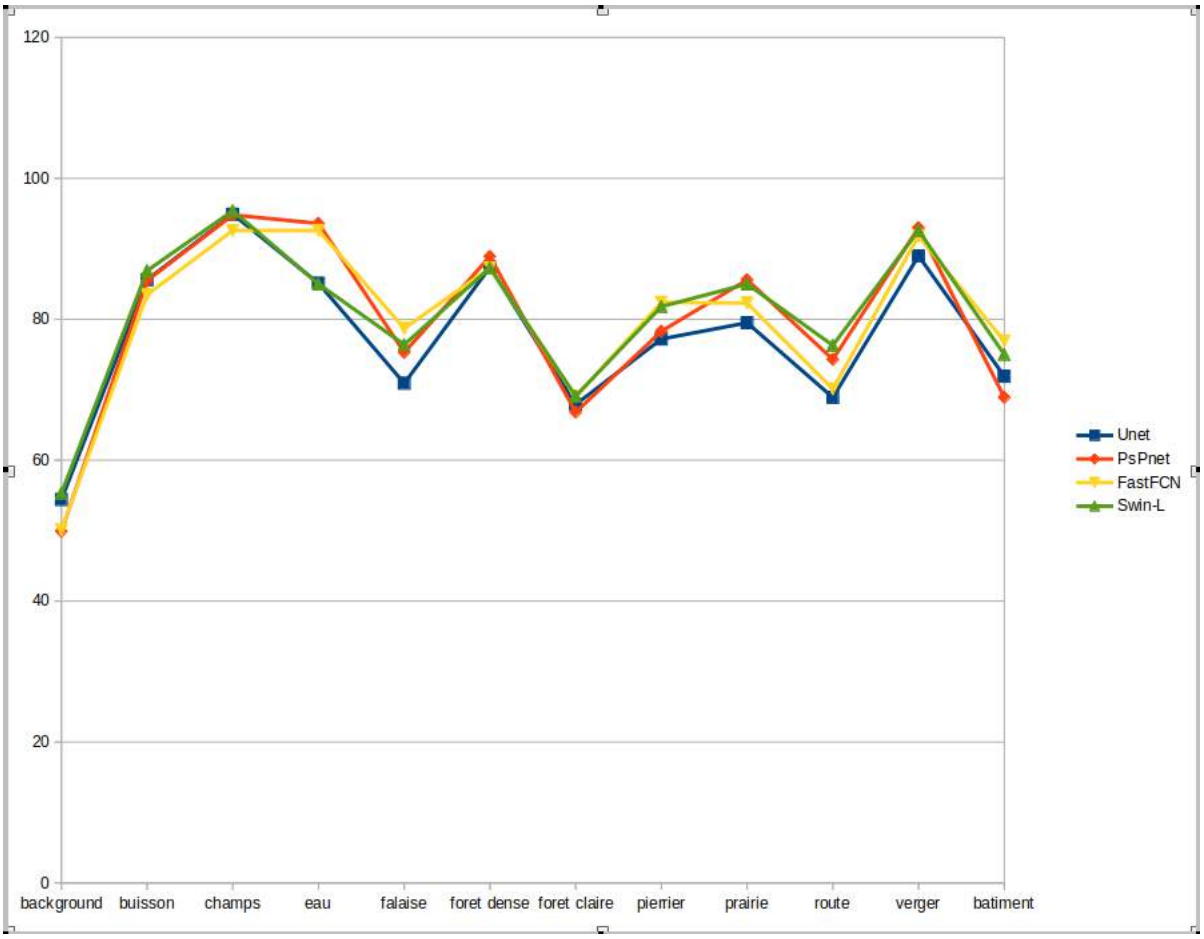


Figure 20 – Accuracy sur le jeu réel en fonction des classes

On peut voir ci dessus l'accuracy par classes des différents réseaux sur le jeu réel. Les meilleures scores s'observent pour les classes qui caractérisent les mi-

lieux agroforestiers: buissons, champs, forêt dense, prairie, verger. En ce sens il s'agit d'une bonne nouvelle puisque le cadre de ce stage s'inscrivait dans un contexte de caractérisation du milieu agroforestier. Globalement ces classes offrent de bons résultats car elles sont très caractéristiques et offrent peu de confusion avec les autres classes. Les classes verger et champs présentent les meilleurs résultats, forêt dense est moins bonne en partie dû à la grande confusion avec la classe forêt claire dont la limite entre ces deux est parfois difficile à établir pour un humain. De plus le réseau acquiert souvent un niveau trop précis de détails ce qui pénalise certaines classes comme la classe forêt. Par exemple, les petites haies d'arbustes bien denses ou encore un groupe d'arbres isolés dans un champ sont classées en forêt dense. La classe forêt claire est souvent confondue avec la classe forêt et possède un rappel assez faible. Ceci n'est pas étonnant car nous avions eu nous même du mal à décider si nous annotions telle ou telle zone en forêt claire ou pas. Cette classe n'est pas évidente à définir sur certaines images et apporte beaucoup de confusion sur le réseau. Les classes pierrier et falaise sont elles aussi assez souvent confondues car le contenu de ces zones est souvent assez proche, et le réseau peine à apprendre le contexte entourant ces zones. Les pierriers étant souvent situés dans un lit de rivière et les falaises dans des zones environnées de buissons. Pour ce qui est de la classe route et bâtiment, ces classes possèdent peu de pixels et une forte hétérogénéité dans les contenus surtout pour la classe bâtiment ce qui explique en partie les résultats plus mauvais.

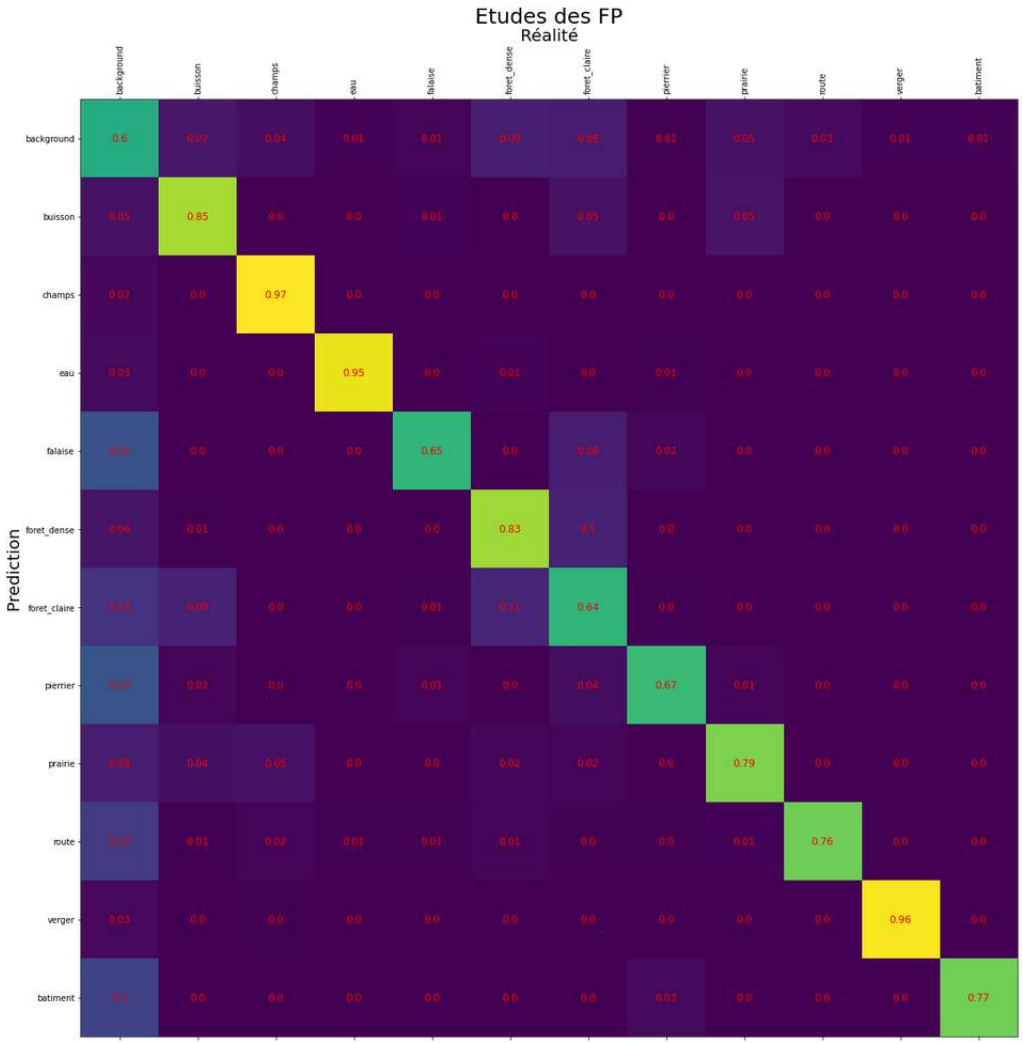


Figure 21 – Matrice de confusion normalisé sur les lignes pour Unet sur le jeu réel

Les confusions les plus fréquentes se situent entre la classe forêt et la forêt claire, la forêt claire et les buissons ou encore pierrier et falaise. Ces classes sont en effet assez proches dans leur contenus. Elle présente certainement le plus d'erreurs dans leurs annotations puisqu'étant difficile parfois à distinguer.

7.2 Résultats sur le jeu réel

	background		buisson		champs		eau		falaise		foret dense	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Unet	0,49	0,54	0,78	0,86	0,9	0,95	0,95	0,85	0,71	0,7	0,88	0,88
PsPnet	0,48	0,51	0,86	0,85	0,93	0,95	0,94	0,92	0,75	0,75	0,9	0,87
FastFCN	0,5	0,5	0,83	0,84	0,89	0,93	0,93	0,93	0,8	0,79	0,88	0,87
Vit-L	0,4	0,55	0,86	0,77	0,88	0,9	0,87	0,77	0,76	0,67	0,85	0,85
Swin-L	0,5	0,55	0,85	0,86	0,92	0,95	0,94	0,95	0,85	0,77	0,88	0,87
Mean	0,474	0,53	0,836	0,836	0,904	0,936	0,926	0,884	0,774	0,736	0,878	0,868
Variance*100	0,178	0,055	0,113	0,143	0,043	0,048	0,103	0,548	0,283	0,248	0,032	0,012

	foret claire		pierrier		prairie		route		verger		batiment		Mean	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Unet	0,72	0,68	0,84	0,77	0,82	0,8	0,79	0,7	0,95	0,89	0,8	0,72	0,80	0,78
PsPnet	0,67	0,67	0,82	0,77	0,86	0,86	0,8	0,74	0,94	0,93	0,71	0,68	0,81	0,79
FastFCN	0,68	0,69	0,88	0,82	0,8	0,82	0,78	0,7	0,94	0,92	0,81	0,77	0,81	0,80
Vit-L	0,61	0,62	0,85	0,7	0,72	0,7	0,71	0,59	0,87	0,72	0,78	0,75	0,76	0,72
Swin-L	0,72	0,71	0,88	0,81	0,85	0,85	0,77	0,76	0,95	0,93	0,86	0,76	0,83	0,81
Mean	0,68	0,674	0,854	0,774	0,81	0,806	0,77	0,698	0,93	0,878	0,792	0,736		
Variance*100	0,205	0,113	0,068	0,223	0,31	0,408	0,125	0,432	0,115	0,807	0,297	0,133		

Figure 22 – Résultats sur jeu réel par classe

Nous avons pu tester 5 réseaux différents classés dans un ordre chronologique d'apparitions.

— U-Net

— Pyramid Scene Parsing Network

— FastFCN

— Vision Transformer

— Swin Transformer

Ce que l'on observe globalement est un léger gain de performance sur les nouveaux réseaux et peu de variations des résultats sur les classes ce qui est rassurant. Les réseaux rencontrent à peu près les mêmes difficultés à noter cependant que certaines classes ont une plus forte hétérogénéité que d'autres: eau, prairie, verger.

Les Vision Transformer n'ont pas pu être entraînées avec les mêmes paramètres (batch size, crop size...) que les autres, les résultats sont donc beaucoup moins bons sur celui-ci à cause d'une limitation matérielle. Par la suite, on établira les benchmarks seulement sur les 4 réseaux restants.

7.3 Résultats sur le jeu réel avec Augmentation sur le jeu de données artificiel

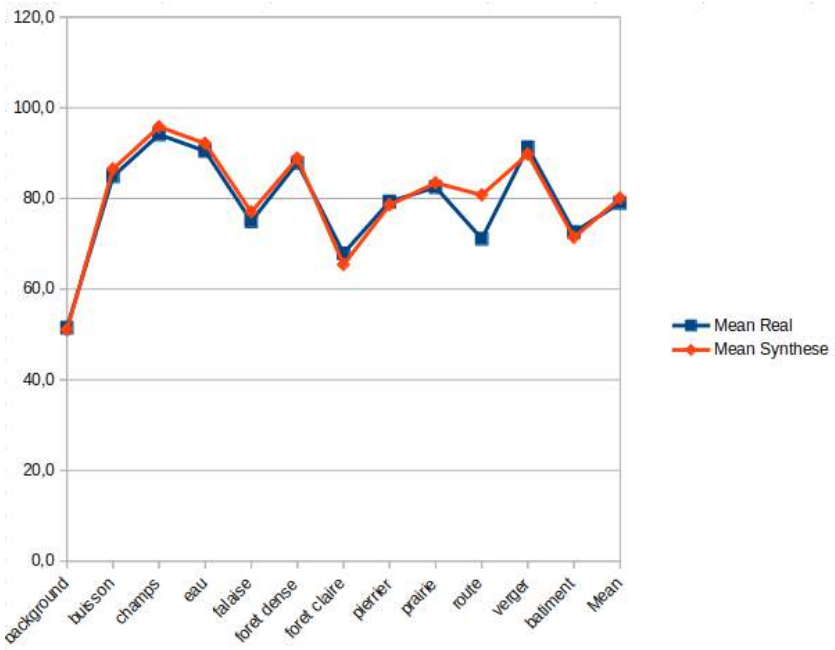


Figure 23 – Accuracy par classes avec ou sans augmentation du jeu de données artificiel

L'augmentation de données par le jeu de données artificiel n'apporte pas de différence notable hormis sur la classe route. La différence de résultat pour la classe route est très remarquable et peut s'expliquer par le manque de précision du jeu réel sur la classe route. En effet on a souvent omis lors de l'annotation de

classer les zones de chemins et sentiers en route induisant ainsi une confusion pour le réseau entre la classe route et forêt par exemple.

7.4 Résultats sur le jeu réel avec Augmentation sur le jeu GAN

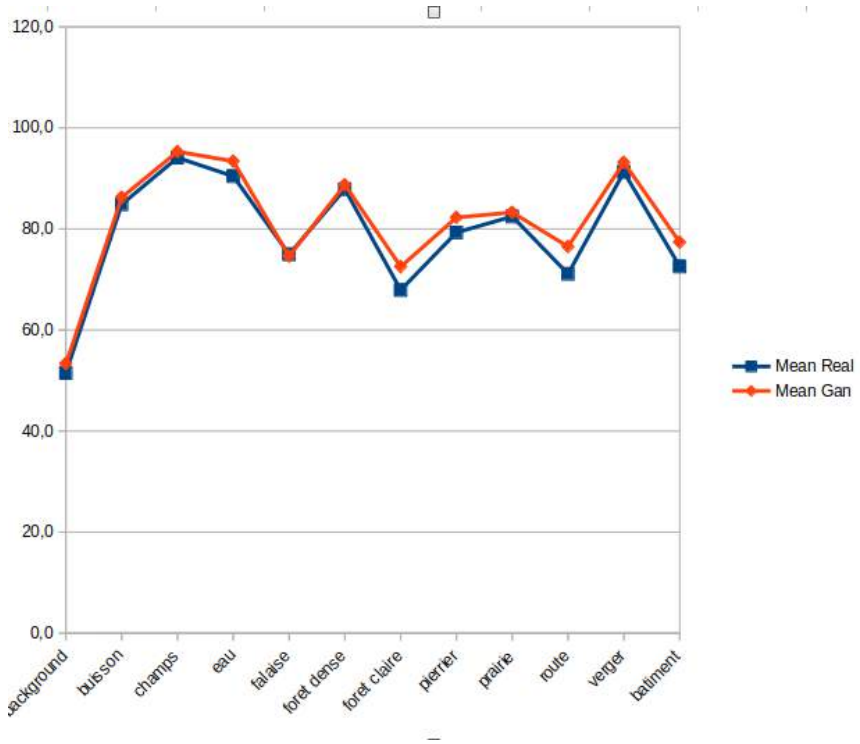


Figure 24 – Accuracy par classes avec ou sans augmentation du jeu Gan

L'augmentation de données par le jeu de données générées par GAN apporte quelques améliorations sur de nombreuses classes. Notamment sur les classes de grande confusion comme forêt claire, pierrier. Le gain moyen est de 2% d'accuracy.

7.5 Vue d'ensemble

Jeu réel						
Method	backbone	year	val mIoU	val mAcc	crop size	Lr SChd
Unet-S5-D16	FCN + Unet	2015	66.1	77.1	512*512	60K
PSPnet	r50-d8	2017	68.3	79.6	512*512	60K
FastFCNN	r50-d32	2019	68.6	79.8	512*512	80K
ViT-L	SETR-PUP	2020	59.7	72.0	256*256	80K
Swin-L	UperNet	2021	71.1	81.5	512*512	80K
Jeu réel + artificiel						
Unet-S5-D16	FCN + Unet	2015	66.6	78.4	512*512	60K
PSPnet	r50-d8	2017	70.6	81.6	512*512	60K
FastFCNN	r50-d32	2019	68.5	80.2	512*512	80K
Swin-L	UperNet	2021	-	-	-	-
Jeu réel + jeu GAN						
Unet-S5-D16	FCN + Unet	2015	67.5	78.9	512*512	60K
PSPnet	r50-d8	2017	70.9	82.4	512*512	60K
FastFCNN	r50-d32	2019	72.1	82.6	512*512	80K
Swin-L	UperNet	2021	-	-	-	-

Le tableau ci-dessus résume l'intégralité de nos expériences effectuées avec l'outil mmsegmentation. Finalement, nous n'avons pas eu le temps de faire l'apprentissage sur les Swin Transformer pour le jeu de données artificiel et le jeu de données généré par GAN.

7.6 Discussions

Nous observons en augmentant notre jeu de données par des images générées par GAN des résultats meilleurs d'environ 2% sur nos deux métriques utili-

sées (IoU et l'accuracy). On observe aussi un gain de performance léger sur les réseaux plus récent pour les 3 jeux. D'après nos observations on peut en déduire que les résultats dépendent de l'âge du réseau (généralement relié à sa complexité) et de l'augmentation par GAN ou pas. Ces résultats sont cependant à relativiser pour diverses raisons: L'impact de la qualité de nos annotations sur ces résultats est grandement discutable. Nous n'avons pas effectué de réelle analyse sur les erreurs engendrées ni de vérification par différents experts de leur qualité. Les écarts sont très faibles pour conclure d'un véritable gain entre une méthode et une autre; un jeu et un autre. Par ailleurs, ces faibles différences nous amènent à questionner l'intérêt d'utiliser des méthodes plus récentes offrant de faibles gains de performances mais multipliant souvent les temps d'apprentissage GPU par 5 voir 10.

8 Perspectives d'amélioration

Nous avons pus identifier 3 pistes d'amélioration:

— L'amélioration de la qualité des annotations notamment en réalisant un protocole rigoureux de leurs création. Idéalement faire effectuer les annotations par 2-3 personnes à 2 reprises avec un intervalle de temps assez important. On pourrait essayer de réentraîner plusieurs fois le réseau sur ses propres prédictions de façon à obtenir les meilleures annotations possibles. Si l'on veut aller plus loin pour évacuer l'étape d'annotation, il faut explorer le domaine de l'apprentissage non supervisé. L'expert devra tout de même valider les résultats finaux.

— Réaliser une segmentation d'images par une approche de GAN. Il est possible de faire le procédé inverse que nous faisions avec les GAN. Donner au réseau des images satellites en entrée et lui demander de produire la

carte en sortie.

Figure 25 – Exemple d'image du jeu de données Maps. L'image de droite est donnée en entrée au réseau et l'image de gauche est celle produite par le GAN.

— Réduire à nouveau la taille de nos imagettes: Travailler avec des images de 256*256 pixels au lieu de 512*512. Cela nous permettra d'augmenter au moins par 4 tous nos batch size pour les entraînements sur les réseaux de segmentation sur mmsegmentation et pour l'entraînement du GAN. On a pu observer des résultats parfois sensiblement meilleurs (surtout pour le GAN) lorsque le batch size etait plus gros. En revanche la taille de 512*512 avait été choisie pour avoir des zones assez grosses et conserver du contexte entres chaques classes. Le risque avec des images de 256*256 est d'avoir des images couvrant des zones trop petites composé par exemple d'une seule classe.

9 Conclusion

Nous souhaitions évaluer et comparer différents réseaux de segmentation sur un jeu de données d'images satellites des Hautes Alpes. Le jeu de données à notre disposition n'était pas annoté ce qui nous a permis de définir

nos propres classes. L'accent a été mis sur la définition de classes caractérisant au mieux les différents systèmes agroforestiers rencontrés. Nous avons donc cherché à faire de l'apprentissage supervisé une fois notre jeu d'apprentissage constitué. Les résultats obtenus varient beaucoup entre les classes mais peu entre les réseaux, de manière générale les classes d'agroforesterie sont celles qui offrent les meilleurs résultats. La difficulté à mettre en œuvre un jeu d'apprentissage doté d'annotations précises et correctes à ensuite motivé la constitution de jeux d'apprentissage augmenté avec des images aux annotations plus précises: un jeu de données artificiel et un jeu de données générées par GAN. À la fin de ce travail, nous avons montré que l'augmentation par images artificielles n'apporte pas un réel gain de résultat sur nos classes, en revanche en complétant notre jeu de données par des images générées par GAN nous avons réussi à améliorer un peu les performances des réseaux sur le jeu de validation.

10 Code Source

L'intégralité du code des experiences effectuées se trouve à cette adresse.

Bibliographie

Barthélémy, D., Caraglio, Y. 2007. Plant Architecture: A Dynamic, Multilevel and Comprehensive Approach to Plant Form, Structure and Ontogeny. Annals of Botany, 99 (3) : pp. 375-407 19

Jaeger, Marc, and P. H. De Reffye. "Basic concepts of computer simulation of plant growth."Journal of biosciences 17.3 (1992): 275-291.

Philippe de Reffye, Baogang Hu, Mengzhen Kang, Véronique Letort, Marc Jaeger. Two decades of research with the GreenLab model in Agronomy. Annals of Botany, Oxford University Press (OUP), 2021, 127 (3), pp.281-295. ⟨10.1093/aob/mcaa172⟩. ⟨hal-02950606⟩

Joly, A., Bonnet, P., Goëau, H., Barbe, J., Selmi, S., Champ, J., Dufour-Kowalski, S., Affouard, A., Carré, J., Molino, J.F. and Boujemaa, N., 2016. A look inside the Pl@ ntNet experience. Multimedia Systems, 22(6), pp.751-766

Frederic Borne, Gaëlle Viennois. Texture-based classification for characterizing regions on remote sensing images. Journal of applied remote sensing, Bellingham, WA : SPIE, 2017, 11 (03), pp.036028.

Pol Kennel, Christophe Fiorio, Frederic Borne. Supervised image segmentation using Q-Shift Dual-Tree Complex Wavelet Transform coefficients with a texton approach. Pattern Analysis and Applications, Springer Verlag, 2017, 20 (1), pp.227-237

You Only Need Adversarial Supervision for Semantic Image Synthesis, Edgar Schonfeld and Vadim Sushko and Dan Zhang and Juergen Gall and Bernt Schiele and Anna Khoreva,International Conference on Learning Representations,2021