Identification d'espèces végétales d'intérêt éco-hydrologiques dans les paysages agricoles : apport du Deep Learning et de l'Intelligence Artificielle

Loïc Lehnhoff

AGROCAMPUS OUEST

☒ CFR Rennes ☐ CFR Angers

Année universitaire : 2020-2021 Spécialité : Ingénieur Agronome Spécialisation : Master 2 de MODélisation en Écologie (MODE) de l'Université de Rennes 1 (master co-habilité)	**Mémoire de fin d'études** ☒ d'ingénieur d'AGROCAMPUS OUEST (École nationale supérieure des sciences agronomiques, agroalimentaires, horticoles et du paysage), école interne de L'institut Agro (Institut national d'enseignement supérieur pour l'agriculture, l'alimentation et l'environnement) ☒ de master d'AGROCAMPUS OUEST (École nationale supérieure des sciences agronomiques, agroalimentaires, horticoles et du paysage), école interne de L'institut Agro (Institut national d'enseignement supérieur pour l'agriculture, l'alimentation et l'environnement) ☐ de Montpellier SupAgro (étudiant arrivé en M2) ☐ d'un autre établissement (étudiant arrivé en M2)

Identification d'espèces végétales d'intérêt éco-hydrologiques dans les paysages agricoles : apport du Deep Learning et de l'Intelligence Artificielle

Par : Loïc Lehnhoff

Soutenu à Rennes le 15/06/2021

Devant le jury composé de :

Président : Cédric Wolf (MC U. Rennes 1)

Maîtres de stages : Fabrice Vinatier (CR INRAe) et Nicolas Parisey (IR INRAe)

Enseignant référent : Frédéric Hamelin (MC Agrocampus Ouest)

Autres membres du jury (Nom, Qualité)

Etienne Rivot (MC Agrocampus Ouest)

Etienne Sirot (MC U. Bretagne Sud)

Les analyses et les conclusions de ce travail d'étudiant n'engagent que la responsabilité de son auteur et non celle d'AGROCAMPUS OUEST

Remerciements

Je souhaite commencer par remercier mes maîtres de stage, Nicolas et Fabrice, pour leurs conseils, leur motivation et leurs encouragements. Malgré le distanciel, ils ont su maintenir une atmosphère de travail décontractée et positive, contribuant à faire de ce stage une excellente expérience. De nos réunions hebdomadaires à la relecture de ce travail, leur implication a été précieuse.

Les bureaux de l'IGEPP m'ont permis de rencontrer des personnes bienveillantes. L'accueil fait lors de ma semaine passée au LISAH pour mieux comprendre les données était également très agréable.

Merci à tous les professeurs du Master MODE, en particulier Frédéric Hamelin et Cédric Wolf, qui m'ont permis d'acquérir les connaissances nécessaires à ce stage et assurent la qualité des enseignements de ce Master.

Merci à Noémie d'avoir relu ce rapport et de son enthousiasme pour mes travaux. Enfin, merci à mes parents et à ma sœur, qui ont su supporter ma présence durant plus d'un mois de confinement et m'ont toujours soutenu dans mes études.

Table des matières

Introduction **1**

Matériel et Méthodes **2**

Présentation du jeu de données image 2

Obtention des images et des relevés botaniques 2

Ortho-image et vérité terrain 4

Modélisation 7

La stratégie de segmentation sémantique 7

Machine learning : caractéristiques et algorithme de classification 8

Deep learning, EfficientNetB0 et multi-vues 11

Un jeu de données déséquilibré 16

Échantillonnage par downsampling 16

Jeux d'entraînement, de test et de validation 17

Des métriques d'évaluation adaptées à des données déséquilibrées 18

Résultats **19**

Modèles Ortho-ML et Ortho-DL 19

Prédictions photographies brutes : mono-entrée-DL 21

Prédictions photographies brutes : multi-entrées-DL 22

Matrices de confusion 22

Discussion **23**

Interprétation des résultats 23

Limites des résultats 26

Perspectives 27

Conclusion **28**

Bibliographie **29**

Liste des illustrations

Liste des figures :

1 : Schéma des pratiques d'entretien pratiquées dans un fossé agricole d'Alignan du Vent dans le cadre d'une étude 3
2 : Schéma des prises de vue dans le fossé d'étude 3
3 : Données issues des campagnes de collecte de photographies 4
4 : Section de l'ortho-image correspondant à la zone M1 du fossé 4
5 : Données 'brutes' de notre jeu de données 6
6 : Illustration de différents niveaux de classification d'une image 7
7 : Schéma simplifié d'un réseau de neurones convolutionnel 12
8 : Précision de modèles faisant état de l'art sur ImageNet 13
9 : Représentation schématique du modèle *Ortho-DL* 14
10 : Architecture générale d'un modèle CNN multi-vues 15
11 : Prédictions *Ortho-ML* des répartitions des espèces végétales sur M2 20
12 : Prédictions *Ortho-DL* des répartitions des espèces végétales sur M2 21
13 : Matrices de confusion *Mono-entrée-DL* et *Multi-entrées-DL* 23
14 : Matrice de confusion pour *Ortho-DL* 23
15 : Vérités terrain pour une portion de M2 31
16 : Représentation simplifiée d'EfficientNet0 32
17 : Prédictions des répartitions des espèces végétales sur l'ortho-image de M4 34
18 : Matrices de confusions pour deux modèles sur M4 34

Liste des tableaux :

1 : Espèces végétales représentées dans nos données 18
2 : Scores de prédiction associés au modèle *Ortho-ML* 19
3 : Scores de prédiction associés au modèle *Ortho-DL* 20
4 : Scores de prédiction associés au modèle *mono-entrée-DL* 22
5 : Scores de prédiction associés au modèle *multi-entrées-DL* 22

Liste des annexes

Annexe I : Comparaison des vérités terrain 31
Annexe II : L'architecture d'EfficientNetB0 32
Annexe III : Calcul des scores 33
Annexe IV : Résultats supplémentaires 34

Glossaire

Accuracy (ENG) :	Mesure de précision des prédictions aux données.
Batch size (ENG) :	Hyper-paramètre utilisé en *deep learning*, c'est le nombre d'individus à prédire avant de corriger les poids du modèle.
Bounding Box (ENG) :	Plus petit rectangle délimitant un objet dans une image.
Deep Learning (ENG) :	Sous-catégorie de *machine learning* basée sur l'utilisation de réseaux de neurones artificiels.
Downsampling (ENG) :	Sous-échantillonnage d'une classe sur-représentée pour diminuer son poids dans le calibrage d'un modèle.
Epochs (ENG) :	Hyper-paramètre utilisé en *deep learning* pour définir le nombre d'itérations pour l'entraînement du modèle.
Features (ENG) :	Caractéristiques d'une image (texture, contraste, etc).
Fine-tuning (ENG) :	Réglage fin des dernières couches d'un réseau de neurones.
Learning rate (ENG) :	Vitesse d'apprentissage, hyper-paramètre utilisé en deep learning.
Machine learning (ENG) :	Apprentissage automatique, sous-catégorie d'IA.
Max-pooling (ENG) :	Échantillonnage par conservation de maxima locaux.
Nadir :	Direction verticale descendante, opposé de zénith.
Ortho-image :	Image en vue aérienne orthorectifiée et mosaïquée
Overfitting (ENG) :	Sur-apprentissage, qui correspond trop parfaitement à un échantillon de données et généralise donc mal.
Shapefile(ENG) :	Format de fichier pour le stockage de données vectorielles.
Training set (ENG) :	Jeu de données d'entraînement en modélisation statistique.
Transfer learning (ENG) :	Apprentissage par transfert, consiste à transférer les connaissances acquises sur un jeu de données à un autre.
Upsampling (ENG) :	Suréchantillonnage, opposé du *Downsampling*.
Rasteriser :	Action qui consiste à convertir une image vectorielle en image matricielle.

Liste des abréviations

API :	Application Programming Interface.
CCM :	Coefficient de corrélation de Matthews.
CLAHE :	Contrast Limited Adaptive Histogram Equalization.
CNN :	Convolutional neural network.
DL :	Deep Learning.
ENG :	Terme Anglais.
IGEPP :	Institut de Génétique, Environnement et Protection des Plantes.
KNN :	K-Nearest Neighbors.
LSTM :	Long Short-Term Memory.
ML :	Machine Learning.
NaN :	Not a Number.
LISAH :	Laboratoire d'étude des Interactions entre Sol-Agrosystème-Hydrosystème.
IA :	Intelligence Artificielle.
RAM :	Random Access Memory.
RF :	Random Forest.
RGB :	Red Green Blue.
SVM :	Support Vector Machine.
TIFF :	Tagged Image File Format.
GPU :	Graphics Processing Unit

1. Introduction

Durant la dernière décennie, l'utilisation des approches d'intelligences artificielles (IA), des algorithmes dont les objectifs sont de simuler des traits de l'intelligence humaine, a explosé dans le monde des sciences appliquées. Ce domaine étant très large, les IA sont divisées en plusieurs catégories dont une grande partie cherche à passer le test de Turing (Turing, 1950). C'est le cas des assistants vocaux de Google, Apple ou Amazon.

Néanmoins, c'est une autre catégorie d'IA qui concentre l'intérêt des scientifiques depuis quelques années : le *machine learning* (ML) et sa sous-catégorie, le *deep learning* (DL). Ces sous-catégories d'IA ont des applications dans tous les domaines : du marketing au transport en passant par nos téléphones, on les retrouve partout. Cet essor dans leur utilisation a trois origines : des ordinateurs plus puissants, des algorithmes efficaces pour tirer parti de cette puissance de calcul et la disponibilité de jeux de données de tailles colossales pour entraîner les modèles sur n'importe quel problème (Wäldchen and Mäder, 2018).

Récemment, la mise en avant du DL avec des performances lors de concours tels que ImageNet (Krizhevsky, Sutskever and Hinton, 2017) a permis le développement exponentiel du secteur de l'analyse d'image. La reconnaissance faciale, les voitures en conduite autonome ou la restauration de photographies anciennes : tous exploitent le DL.

En biologie, l'utilisation du DL s'est démocratisée avec la classification d'images, où de nombreuses applications sont possibles. En agriculture, il est possible d'automatiser une détection précoce des mauvaises herbes et leur traitement à partir d'un véhicule (Olsen *et al.*, 2019). A partir d'images aériennes de parcelles agricoles, des applications à un usage raisonné d'herbicides sont envisagées (Bakhshipour and Jafari, 2018). Et en écologie, les IA sont utilisées pour surveiller l'évolution de la distribution d'espèces végétales ou la propagation de maladies végétales (Safonova *et al.*, 2019).

En botanique, l'utilisation d'IA est particulièrement prometteuse pour pallier le déficit d'experts de terrain. En effet, à l'aide d'une méthode d'identification automatisée des communautés végétales à haut débit, on pourrait obtenir une meilleure exhaustivité des surfaces couvertes, une meilleure identification des espèces rares ou une réduction des biais d'observation liés à des expertises différentes. Or, il se trouve que l'analyse d'image permet déjà d'identifier des espèces végétales *in situ*, de les quantifier, et même, de les spatialiser.

A partir d'une telle méthode, il deviendrait alors possible de cartographier les propriétés hydrologiques d'un milieu (Rubol, Ling and Battiato, 2018), de calculer des indicateurs de biodiversité tel l'indice de Shannon, ou de surveiller l'évolution de distributions d'espèces plus facilement.

L'exemple de Pl@ntNet (Goëau *et al.*, 2014) montre qu'il est possible de prédire avec précision de nombreuses espèces végétales différentes, à partir de données

images variées. Cependant, la généralisation de cette technique à des photographies de plantes prises in situ pose problème. En effet, sans une attention particulière apportée à la prise de vue, tel que c'est le cas pour beaucoup de photographies de Pl@ntNet, il est beaucoup plus difficile d'exploiter les données.

Dans le cas de milieux à végétation dense comme les fossés agricoles méditerranéens, la différenciation d'un végétal d'intérêt avec l'arrière-plan peut s'avérer complexe et des cas d'occlusion de plantes par d'autres peuvent apparaître. La résolution de ces problèmes est nécessaire pour garantir une spatialisation des occurrences des espèces *in situ*, afin de les rapporter aux services écosystémiques qu'elles fournissent.

Dans ce contexte, nous étudierons la problématique suivante : peut-on prédire de manière satisfaisante la distribution d'espèces végétales dans les fossés agricoles méditerranéens par l'intelligence artificielle ? et, le cas échéant, existe-t-il une plus-value à utiliser des photographies multi-angles pour cela ?

Dans le cadre de cette problématique, on fait l'hypothèse que l'utilisation des photographies multi-angles permettra d'augmenter la précision des prédictions des modèles. Cette hypothèse est motivée par le fait que dans le cadre de données obtenues sur des plantes *in situ*, la multiplication des angles de vues permet de contrer les cas d'occlusion et d'avoir un effet de parallaxe.

Pour répondre à cette problématique, le travail s'est articulé autour d'une stratégie itérative afin de sélectionner la meilleure méthodologie d'IA pour l'identification végétale. A partir de construction de méthodes de segmentation sémantiques simples, utilisant uniquement des vues aériennes, une comparaison des performances entre ML et DL est réalisée. Le type d'IA le plus performant est conservé et sert de base à la construction d'une méthode d'identification utilisant des données multivues.

2. Matériel et Méthodes

2.1. Présentation du jeu de données image

2.1.1. Obtention des images et des relevés botaniques

Le jeu de données évoqué dans ce rapport a été collecté préalablement au stage, dans le cadre d'une étude des communautés végétales se développant spontanément dans les fossés agricoles méditerranéens suivant différentes pratiques d'entretien.

Les fossés permettent aux agriculteurs de drainer les eaux de pluies lors d'épisodes pluvieux, parfois extrêmes, en région méditerranéenne. Ces milieux remplissent également d'autres rôles et ils sont notamment de véritables réservoirs de biodiversité (Dollinger *et al.*, 2015). Le développement de communautés végétales spontanées affecte l'écoulement de l'eau et les capacités de drainage du fossé (Rudi *et al.*, 2020). Pour cette raison, les agriculteurs entretiennent régulièrement les fossés de leurs propriétés (Dollinger *et al.*, 2015). Diverses pratiques d'entretien existent : le curage, le fauchage, le désherbage chimique ou le brûlis.

C'est dans ce contexte qu'un jeu de données a été collecté au LISAH sur des espèces végétales spontanées dans un fossé agricole de la commune d'Alignan du Vent (34290). Le jeu de données est constitué d'un relevé botanique exhaustif d'environ 70 espèces végétales cartographiées. Il est accompagné de plusieurs milliers de photographies de la zone d'étude, prises selon des angles de vue différents. Un traitement par photogrammétrie de ces images a permis de constituer une ortho-image géolocalisée, d'une résolution sub-centimétrique du fossé. Ces données ont été relevées chaque année entre 2015 et 2017 (Vinatier *et al.*, 2018).

Le jeu de données ainsi obtenu est constitué d'une grande quantité de données de qualité, ce qui est un pré-requis nécessaire pour l'utilisation de modèles d'IA et notamment l'entraînement de modèles de DL.

Le but de cette précédente étude était donc de déterminer, suivant les pratiques d'entretien, quelles communautés végétales se développaient dans les fossés. Pour cela, un fossé agricole a été étudié pendant 3 ans, de 2015 à 2017. Le fossé a été découpé en 4 zones : M1, M2, M3 et M4 (nommées réplicats sur la figure 1). Chaque zone correspond à une répétition. Dans chacune, 5 pratiques d'entretien différentes ont été appliquées (voir figure 1). Sur les 3 années d'étude, un relevé botanique cartographié et un relevé photographique ont été réalisés au moins une fois par an.

Les relevés photographiques ont été réalisés à l'aide d'une caméra placée sur une perche, tenue à 2,5 mètres de hauteur au-dessus du fossé environ. La couverture photographique de chaque zone se fait en déclenchant l'appareil photo en rafale lors de 7 passages le long du fossé. A chaque passage, le fossé est photographié sous un angle différent, ainsi 7 angles de vue du même objet sont obtenus : 3 angles de vue de chaque côté et une vue du fossé au nadir (voir figure 2).

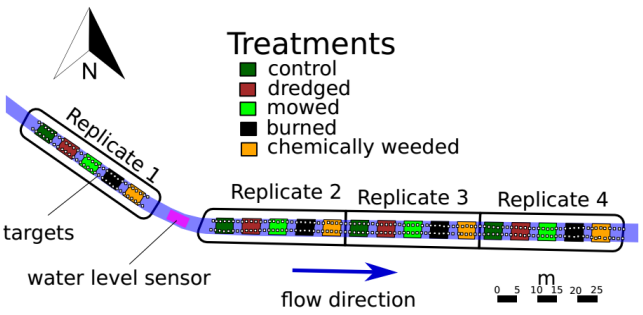


Figure 1 : Schéma des pratiques d'entretien pratiquées dans un fossé agricole d'Alignan du Vent dans le cadre d'une étude.
control = témoin, dredged = dragué, mowed = fauché, burned = brûlé, chemically weeded = désherbé chimiquement.
Source : Vinatier *et al.*, 2018.

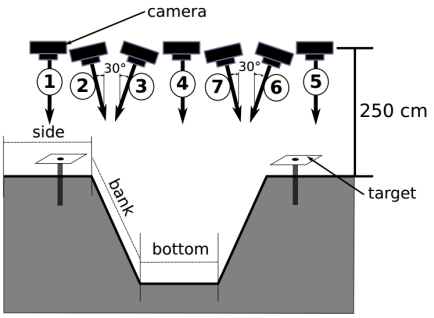


Figure 2 : Schéma des prises de vue du fossé d'étude.
Source : Vinatier *et al.*, 2018.

En ce qui concerne les relevés botaniques : ils ont été effectués sur place avec l'aide de botanistes (Yves Caraglio, CIRAD), qui ont tracé les distributions des espèces relevées sur une carte représentant le fossé. Les tracés sont par la suite numérisés et géoréférencés avec des *shapefiles* (voir Glossaire).

2.1.2. Ortho-image et vérité terrain

A la suite de ces campagnes, ce sont environ 750 photographies divisées en 7 angles de vue qui sont obtenues pour chaque zone. Ce qui constitue environ 3000 images par campagne, sur l'ensemble du fossé.

A partir de ces images il est possible de déterminer la distance à l'objectif de l'appareil photo de chaque pixel (voir figure 3). Grâce à cette information, et à la connaissance des localisations précises des balises (*target* sur les figures 1 et 2) posées autour du fossé, il est possible de calculer la position de chaque pixel dans le référentiel monde.

Figure 3 : Données issues des campagnes de collecte de photographies. (Gauche) Photographie prise en 2017 à Alignan du Vent par Fabrice Vinatier. (Droite) Distances normalisées à l'objectif de l'appareil photo. Unité arbitraire, plus la valeur est proche de 0 plus le pixel est proche de l'objectif.

Grâce à la technique de la photogrammétrie (Vinatier *et al.*, 2018), il est ensuite possible d'utiliser les données photographiques multi-vues afin de générer une représentation aérienne de chaque zone du fossé. C'est sur ces ortho-images que les relevés botaniques numérisés peuvent être superposés (voir figure 4).

Figure 4 : Section de l'ortho-image correspondant à la zone M1 du fossé, établie à partir de photographies de 2016. Les délimitations des distributions des espèces illustrent une partie des espèces végétales cartographiées sur la zone.

Pour chacune des 4 zones du fossé, on obtient une ortho-image de très grande qualité : avec une résolution d'environ 35 000 pixels par 5 000 pixels. Un mètre dans la réalité est représenté par 1000 pixels dans l'image. De plus, ces ortho-images sont enregistrées au format TIFF (Tagged Image File Format), format qui permet le stockage d'informations telles que la géolocalisation de l'image (ce qui donne le GeoTIFF). L'inconvénient pour des images

de si grande qualité est leur stockage : jusqu'à 9 Go d'espace de disque sont nécessaires pour une seule image.

Cette propriété est limitante pour le traitement des images et la modélisation qui utilise ces images comme donnée d'entrée. En effet, le calcul de convolutions (voir partie 2.2.2.) est gourmand en mémoire vive (ou RAM) et le chargement de plusieurs giga-octets d'images peut ralentir l'ordinateur. En conséquence, pour permettre leur utilisation les images ont été modifiées afin de minimiser l'espace nécessaire à leur stockage.

Pour commencer, la résolution des ortho-images a été dégradée, pour passer de 35 000 x 5 000 à 17 500 x 2 500 pixels. Cette diminution de la résolution se fait par interpolation bilinéaire (intégrée à la librairie *OpenCV* de Python). Malgré cette diminution, 500 pixels représentent un mètre, donc peu d'informations sont perdues par cette réduction. De plus, certaines ortho-images sont tournées en diagonale, notamment M1 (voir figure 1). Hors, une image est toujours un rectangle, ainsi de nombreux pixels sont inutiles autour du fossé. Pour trouver l'angle de rotation de l'ortho-image pour obtenir un fossé aligné à l'horizontale, il suffit de trouver les valeurs propres de la matrice de covariance de l'ortho-image. Ensuite, après rotation, il suffit de rogner les pixels trop éloignés du centre du fossé. Après enregistrement au format de fichier léger numpy (une librairie spécifique de Python), cette méthode permet de limiter la taille des ortho-images à 100 Mo.

Les formes dessinées sur la figure 4 correspondent aux relevés botaniques numérisés et constituent de ce fait la vérité terrain : ils seront utilisés pour l'entraînement et l'évaluation des modèles. Ces données doivent donc être les plus fidèles à la réalité possible. Pour cette raison, la totalité des distributions des espèces végétales n'a pas été utilisée dans nos modèles. Seules les espèces pour lesquelles les répartitions numérisées à partir du terrain étaient visibles facilement sur l'ortho-image ont été sélectionnées dans un premier temps.

De plus, dans la pratique les polygones ne s'excluent pas les uns les autres ce qui complique l'exploitation des données. En effet, certaines distributions se superposent du fait de la structure complexe du milieu étudié. *In situ*, une plante haute telle que le *Rumex crispus* peut tout à fait recouvrir une plante basse telle que la *Knautia arvensis*. Cette complexité se retranscrit dans ces données numérisées (voir figure 4).

Pour unifier les données en vue de leur exploitation, une phase de pré-traitement des images est réalisée. Un script R permet d'automatiser la rasterisation (voir Glossaire) des *shapefiles* : ils sont transformés en images de même résolution que les ortho-images. Puis, un script Python procède à la gestion des intersections entre les distributions de chaque espèce.

Afin de ne perdre aucune information, et pour ne pas favoriser l'apprentissage du modèle sur des données tronquées, les intersections sont considérées comme des nouvelles classes à part entières. Le script leur donne le nom "intersection X & Y" (où X et Y sont les noms des espèces dont les distributions se superposent). Cette stratégie permettra d'utiliser des matrices de confusion afin de déterminer avec quelles classes les intersections sont le plus souvent confondues. Cela facilitera l'interprétation des résultats.

Enfin, sur les ortho-images, des variations de luminosité et de contraste sont visibles. Une égalisation est donc réalisée. Elle se base sur la méthode CLAHE (voir Abréviations) dont l'avantage est la réduction des variations de contrastes entre les zones d'une même image, générant un type d'égalisation cohérent entre toutes les images du jeu de données.

En ce qui concerne les images brutes (c'est-à-dire les photographies), aucune vérité terrain n'a été établie lors des collectes de données. En effet, cartographier les relevés botaniques permet de les utiliser sur des images en vues aériennes mais pas sur des photographies brutes. Pour obtenir des vérités terrains parfaites sur ces photographies, il faudrait annoter manuellement les plantes de chaque image brute (ou tout du moins, une bonne partie d'entre-elles). S'il devait être mis en place, ce processus serait très chronophage.

La solution adoptée est de générer les vérités terrains des images brutes de manière automatisée. Puisque les coordonnées des pixels de chaque image sont connues dans le référentiel monde (ces coordonnées ont permis de construire l'ortho-image), alors il est possible d'utiliser ces coordonnées dans l'autre sens : connaissant les coordonnées des pixels dans l'ortho-image on peut en déduire les valeurs des pixels sur une photographie.

Un script Python permet de réaliser cette génération d'images. Les valeurs des pixels pour les vérités terrains des photographies dépendent ainsi des classes établies pour les ortho-images. (voir figure 5). Les inconvénients de cette méthode sont sa lenteur (en moyenne une minute par image appliquée à environ 3000 images) et la déformation des distributions, ce qui est dû à la structure des données utilisées. Une comparaison commentée avec les distributions sur l'ortho-image est disponible (voir Annexe I).

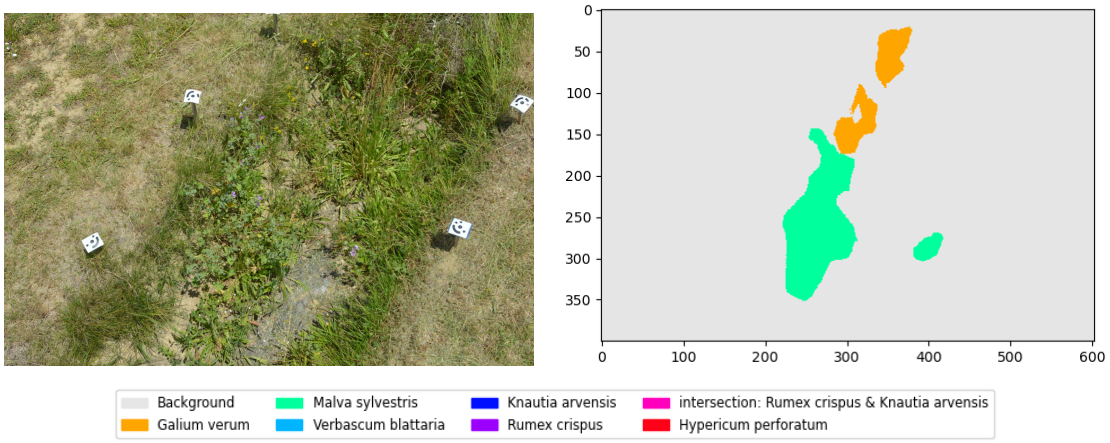


Figure 5 : Données 'brutes' de notre jeu de données.
(Gauche) Photographie prise en 2016 sur le fossé agricole d'étude à Alignan du Vent par Fabrice Vinatier.
(Droite) Vérité terrain associée, déduite de l'ortho-image (voir figure 4).

2.2. Modélisation

2.2.1. La stratégie de segmentation sémantique

Le *machine learning* est un domaine d'étude dont le but est de créer des algorithmes ayant la capacité d'apprendre, sans être explicitement programmés à apprendre. Parmi les sous-catégories du ML on retrouve des algorithmes de classification, des modèles de régression ou encore, le *deep learning*.

Ces méthodes peuvent être appliquées à des tâches très variées. Or, ce qui nous intéresse dans le cadre de ce travail, est d'évaluer la distribution des espèces végétales sur une image. Cet objectif peut être ramené à l'identification, pour chaque pixel de l'image, d'une classe à laquelle il doit appartenir, correspondante à un végétal.

La classification des pixels d'une image peut être effectuée de différentes manières. En vision artificielle, on peut compter 4 niveaux de classification (voir figure 6) :

- Classification : identification d'une classe pour l'image entière (e.g. ballons)
- Segmentation sémantique : identification d'une classe pour chacun des pixels d'une image, on ne cherche pas à faire de différences entre deux instances d'un même objet s'ils sont connexes.
- Détection d'objet : différenciation et identification de chaque instance de chaque objet dans une image, ce qui permet de les localiser avec des boîtes englobantes.
- Segmentation d'instance : combinaison de segmentation sémantique et de détection d'objet. Identification des pixels appartenant à chaque instance de chaque objet.

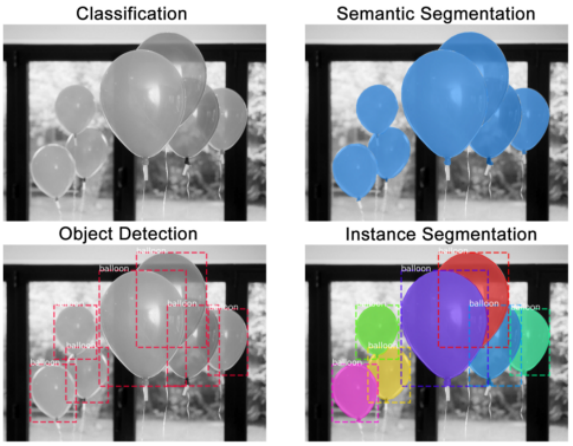


Figure 6 : Illustration de différents niveaux de classification d'une image. Source : Abdulla, 2018.

En ce qui concerne notre travail, nous cherchons à prédire, en premier lieu, la proportion des espèces végétales présentes dans une image, nous avons donc fait le choix d'utiliser des méthodes de segmentation sémantique. En effet, cette méthode permet la détection, l'identification et la quantification de la distribution des communautés végétales dans le fossé, nécessaires à la déduction de propriétés écologiques et hydrologiques.

Cette méthode a l'avantage de donner une classe à chaque pixel, ce qui nous permet de projeter les résultats des prédictions sur une image de même résolution que la vérité terrain d'origine. Cela facilitera la comparaison entre prédictions et réalité.

Avec la structure de nos données, il n'est pas possible d'effectuer de la segmentation d'instance. En revanche, la détection d'objets, avec des boîtes englobantes (*bounding box* en anglais) extraites des données, a été étudiée en parallèle de ce travail (voir partie 4.3.)

La segmentation sémantique est applicable au *machine learning* comme au *deep learning*. Le ML a longtemps été privilégié pour la classification d'images et il est parfois encore pertinent. En effet, le ML a été utilisé avant le DL. Il a fallu attendre d'avoir des ordinateurs assez puissants et les premiers succès dans des challenges tels ImageNet, pour que le DL passe au premier plan. C'est pourquoi nous avons comparé ML et DL dans un premier temps, bien que le DL soit en général plus adapté au traitement d'images aujourd'hui. Le but étant de sélectionner le meilleur modèle pour la classification des images brutes, sur la base de ses capacités à faire des prédictions sur les ortho-images.

2.2.2. Machine learning : caractéristiques et algorithme de classification

Tout d'abord, commençons par un rapide rappel du fonctionnement d'un modèle classique de ML. L'appellation *Machine Learning* regroupe un ensemble de méthodes très différentes, dont des méthodes de classification. Mais sa particularité réside dans l'utilisation de caractéristiques (ou *features*) pour réaliser ses prédictions.

Caractéristiques ou *features* :

Les *features* prennent la forme de tableaux de données faisant correspondre plusieurs individus à différentes variables. Pour des images, il suffit de considérer chaque pixel comme un individu et chaque canal de couleur comme une variable.

Cependant cette séparation des pixels en individus pose un problème : elle supprime toute relation spatiale des pixels entre eux. Pour caricaturer, une fleur rouge sur une plante verte porte une information différente d'une fleur rouge sur une plante rouge. Pourtant si on sépare les pixels de ces fleurs, il deviendra impossible de les différencier. C'est pour cette raison qu'il faut introduire le calcul de *features* supplémentaires, qui viendront s'ajouter à chaque pixel comme un nouveau canal, afin de conserver ce type d'information.

Ces calculs sont des convolutions. Discrète dans le cas du traitement de pixels, la convolution est le produit d'une fonction par une autre. Pour une image, cela signifie qu'une fonction est appliquée à l'ensemble d'une image, prenant en compte tous les pixels.

Il existe un nombre illimité de caractéristiques qui peuvent être extraites d'une image. Certaines sont communes et leur utilisation est facilitée via des librairies : *OpenCV* et *scikit-images* en Python. Mais il est également possible de construire ses propres *features* afin d'extraire des information spécifiques d'une image.

Par praticité, des caractéristiques communes ont été appliquées à nos images. Les données obtenues sont ensuite concaténées avec les canaux des couleurs rouge, vert et bleu (notés RGB) de l'image originale. Via un script python, les caractéristiques suivantes sont calculées :

- Floutage Gaussien : cette opération consiste à moyenner la valeur d'un pixel en fonction des pixels qui l'entourent. Plus un pixel est éloigné du point à flouter, moins il a de poids dans le calcul. La fonction *convolve* de la librairie astropy a été utilisée car elle permet la gestion de valeurs non numériques (ou NaN) comme des pixels transparents, par exemple.

- Filtres de Sobel : ce sont des filtres qui permettent la détection de contours dans une image. Leur fonctionnement passe par un calcul de gradient de l'intensité pour chaque pixel. Un premier calcule le gradient dans le sens de l'écoulement de l'eau, un autre dans le sens perpendiculaire et un dernier cumule les deux. Les fonctions *sobel* de scikit-image et OpenCV ont été utilisées pour ces opérations.

- Filtre Hessien : ce filtre permet également une détection de contours mais sa particularité est l'utilisation de dérivées secondes, ce qui permet la détection de contours continus. Pour cela, la fonction *hessian* de skimage a été utilisée.

Etant donné qu'il y a trois canaux de couleur RGB, ces caractéristiques sont calculées par multiples de trois : une par canal de couleur. De plus, ces opérations ne sont pas appliquées telles quelles à l'image. L'image est d'abord floutée, ce qui permet de "regrouper" les pixels en régions d'intensité, puis les filtres sont appliqués sur cette image floue.

Cependant, il ne faut pas oublier que l'opération de floutage s'appuie sur une fonction gaussienne dont la portée à 95% doit être définie. Étant donné que l'emprise au sol des plantes à détecter ne dépasse pas 20 cm de diamètre dans la réalité, soit environ 200 pixels sur les ortho-images originales, l'écart-type σ à utiliser est donc d'environ 102 pixels.

De plus, comme 20 cm correspond au diamètre maximal des structures à détecter, plusieurs portées sont définies afin d'appliquer les convolutions à des structures plus petites. Finalement, pour chaque canal de couleur, 5 portées sont appliquées pour lesquelles tous les filtres sont calculés. De plus, pour chaque couple de portée, on ajoute le calcul d'une différence des floutages gaussiens. Cela permet de générer des caractéristiques qui mettent en évidence des différences locales par région.

Au total, ce sont 105 canaux qui sont ajoutés à l'image originale. Étant donné la taille des images évoquée en partie 2.1.2., il a été nécessaire d'enregistrer les images et leurs caractéristiques concaténées sous un format plus léger que le format TIFF. Pour commencer, la résolution des images a été dégradée en divisant le nombre de pixels par 4 (soit en divisant par deux le nombre de pixels sur chaque dimension). Ce qui affecte grandement la taille des images mais permet de conserver une bonne qualité de représentation.

De plus, en Python, les images sont traitées comme des tableaux de données numpy. Ces tableaux peuvent être sauvegardés directement sur disque dur, ce qui permet d'exclure des informations inutiles pour la modélisation (telles que la géolocalisation). Grâce à cela, chaque ortho-image accompagnée de ses caractéristiques représente "seulement" 16 Go de stockage. En revanche, le temps de calcul nécessaire pour obtenir les *features* est très long : environ 2,5 jours sont nécessaires.

Algorithme de classification :

Une fois les caractéristiques obtenues, on peut passer au choix de l'algorithme à utiliser pour la classification de nos données. Les algorithmes de ML sont des modèles statistiques qui résolvent des problèmes d'optimisation complexes. Il existe de nombreux modèles, certains sont plus répandus que d'autres et parfois spécifiques à différentes tâches. Parmis les modèles disponibles dans la librairie *scikit-learn* de Python, les modèles suivant ont étés utilisés pour modéliser nos données :

- K-Nearest Neighbors (ou KNN) : c'est la méthode des K plus proches voisins. Elle repose sur la recherche des intervalles dans lesquels les individus d'une classe sont inclus. C'est l'un des algorithmes de classification le plus simple et le plus ancien du ML, mais aussi un des plus efficaces.

- Support Vector Machine (ou SVM) : c'est la méthode des séparateurs à vaste marge. Leur fonctionnement passe par la construction de fonctions qui divisent le plan de distribution des données. Ces séparations ne sont pas nécessairement linéaires. Cette complexité rend ces modèles très performants. Ils sont capables de rivaliser avec certains réseaux de neurones.

- Random Forest (ou RF) : c'est la méthode des forêt d'arbres décisionnels, ou forêt aléatoire. Le fonctionnement de cet algorithme repose sur la génération d'arbres de décisions formés à partir d'échantillons aléatoires des données d'entrée. Les arbres générés sont ensuite regroupés selon les partitions qui maximisent leur performance. C'est l'algorithme le plus répandu en ML, car il est très efficace et s'applique à tout type de données.

L'avantage de ces trois modèles est qu'ils constituent des boîtes blanches. Par opposition aux boîtes noires, dont le contenu est caché, ils sont relativement faciles à conceptualiser et simples à se représenter.

Parmi ces trois catégories, seuls les résultats du modèle RF seront présentés dans la partie résultat (3.1.) de ce rapport. En effet, les autres modèles testés ne permettaient pas d'obtenir des résultats assez satisfaisants pour être analysés. De plus, il est plus facile d'obtenir les probabilités liées à la confiance d'un modèle en ses prédictions avec un RF. Ce modèle sera nommé : modèle *ortho-ML,* dans la suite de ce rapport.

Les résultats obtenus en sortie de ces modèles sont des prédictions sur les classes à affecter à chaque pixel. Une fois entraînés sur nos données, le modèle *ortho-ML* permet de classer 10 millions d'individus (correspondant environ au nombre de pixels dans l'ortho-image) en moins de deux minutes. En effet, avec le ML, c'est la phase de génération des données qui prend du temps. Les prédictions sont extrêmement rapides puisqu'il suffit de suivre des règles établies durant la phase d'entraînement : quasiment aucun calcul n'est requis. C'est le contraire pour le DL.

2.2.3. *Deep learning*, EfficientNetB0 et multi-vues

***Deep Learning* :**

Le *deep learning* est une sous-catégorie du *machine learning*. Cependant, son fonctionnement est tout à fait différent. Pour commencer, le DL ne nécessite pas ou presque pas de pré-traitement des images avant de les utiliser dans un modèle. En effet, si en ML des convolutions sont appliquées à une image avant d'être fournies au modèle, en DL, les convolutions sont estimées (en apprentissage) et réalisées (en prédiction) au sein même du modèle. Ainsi, les images peuvent être données telles quelles en entrée du modèle. Néanmoins des pré-traitements d'images tels que des égalisations peuvent être effectués avant de fournir les données au modèle mais cela a souvent moins d'influence qu'en ML.

Souvent, une augmentation des données est tout de même réalisée. C'est une technique qui consiste à appliquer des rotations, des variations de contrastes, ou des translations à une image. Le but est de limiter le surapprentissage (ou *overfitting*) du modèle. Cette technique est presque systématique en *deep learning*.

Le *deep learning* s'est rendu célèbre après la victoire d'un CNN (ou réseau de neurones à convolution) nommé AlexNet lors de la compétition de classification d'images ImageNet, en 2012. La structure CNN est la plus utilisée en DL aujourd'hui et l'article de leur performance recense plus de 80 000 citations (Krizhevsky, Sutskever and Hinton, 2017).

Décomposons le fonctionnement d'un tel réseau de neurones (voir figure 7). Un CNN est composé classiquement de trois couches : une couche de convolution, une couche de *max-pooling* et une couche de neurones dense (tous connectés). La couche de convolution est un extracteur de caractéristiques, son but est de révéler des patterns précis dans l'image. La couche de *max-pooling* conserve les valeurs maximales de points spatialement proches, après sortie de la couche de convolution (ce qui condense les informations). Enfin la couche de neurones dense permet d'effectuer la classification à proprement parler.

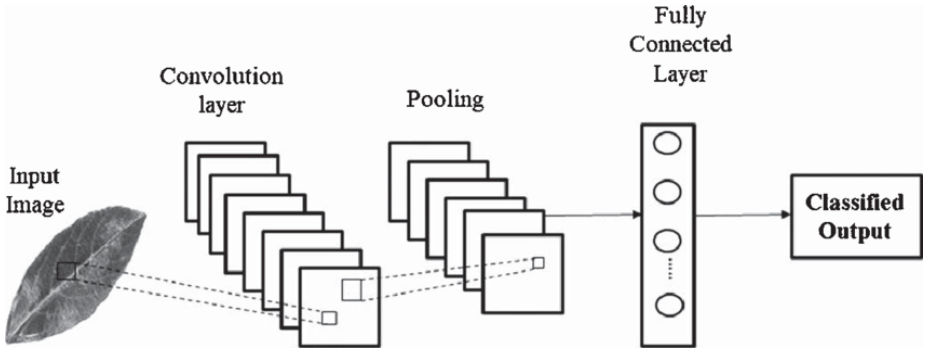


Figure 7 : Représentation schématique simplifiée d'un réseau de neurones convolutionnel.
Source : Anubha Pearline, Sathiesh Kumar and Harini, 2019.

Cette description est évidemment très simplifiée. D'autres couches peuvent être ajoutées au modèle. Et lorsque le nombre de couches d'un modèle augmente on dit alors que la profondeur du modèle CNN augmente. Depuis 2013, de nombreuses architectures se sont développées, et complexifiées. On parle d'architectures, car c'est le choix et l'agencement des couches d'un modèle qui définissent son fonctionnement.

Il existe de nombreuses architectures de DL différentes, chacune adaptée au traitement d'un problème différent. Le CNN est l'architecture la plus populaire et est utilisée pour la classification traditionnelle d'images, le LSTM (abbr.) pour la lecture d'écrits manuscrits, le RCNN (abbr.) pour la détection d'objets ou encore l'encodeur-décodeur pour la traduction de textes ou en segmentation sémantique.

Dans une stratégie de segmentation sémantique, l'utilisation d'un CNN ou d'un encodeur-décodeur sont donc possibles. C'est le CNN qui a été choisi ici car l'utilisation d'un encodeur-décodeur impliquerait de disposer d'un jeu de données contenant plus d'images.

Pour utiliser un CNN, qui attribue une classe à une image entière alors qu'on souhaite prédire la classe d'un seul pixel, on procède de la manière suivante. Une vignette de 20 cm par 20 cm est découpée autour du pixel central à prédire. Comme pour le ML, ces 20 cm correspondent à environ 200 pixels dans l'image, ce qui donne une vignette de 201 x 201 pixels. Cette vignette est ensuite fournie en entrée du modèle qui va renvoyer en sortie une classe, qui pourra être attribuée au pixel central de l'image.

L'inconvénient de cette méthode est qu'elle nécessite de produire autant de vignettes qu'il y a de pixels à prédire, ce qui peut s'avérer gourmand en capacité de calcul. Pour éviter de saturer la mémoire de l'ordinateur, les prédictions sur les ortho-images entières ont donc été faites sur un échantillon des pixels de l'image. 1 pixel sur 25 a été sélectionné de manière régulière, à raison d'un pixel sur 5 en hauteur et un pixel sur 5 en largeur sur l'ortho-image. Cet échantillonnage permet de reconstituer les prédictions sur toute la zone de l'ortho-image, en utilisant une interpolation pour les pixels manquants.

EfficientNetB0 :

En ce qui concerne le modèle, c'est EfficientNetB0 qui a été utilisé. Ce modèle a été développé par une équipe de chercheurs employés par Google comme une alternative plus rapide est plus performante que les modèles classiques utilisés sur ImageNet (Tan and Le, 2019). Sa particularité, outre son architecture générale (nous y reviendrons), est le faible nombre de paramètres dont il a besoin pour atteindre des résultats qui surpassent des modèles tels que Xception ou ResNet, qui font pourtant état de l'art en matière de classification d'images (voir figure 8).

EfficientNet existe en plusieurs versions (de B0 à B7). Nous avons utilisé la version B0 car c'est la base sur laquelle sont construites les autres versions d'EfficientNet. Elle contient donc le moins de paramètres à calibrer.

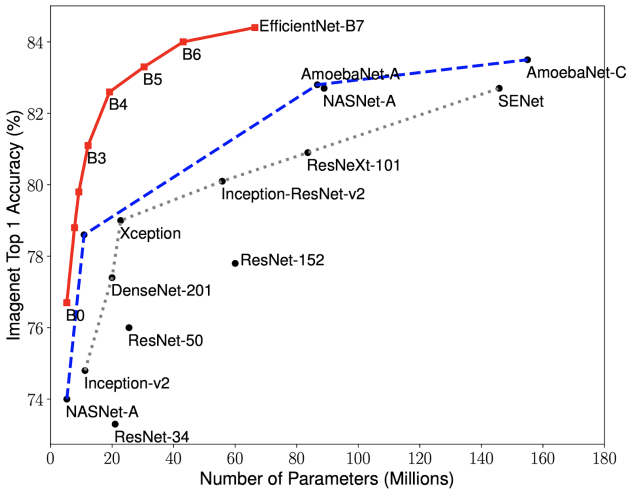


Figure 8 : Précision de modèles faisant état de l'art sur ImageNet en fonction de leur nombre de paramètres (en millions). Les lignes indiquent la progression des performances de modèles basés sur les mêmes architectures. Source : Tan and Le, 2019.

Nous avons choisi EfficientNetB0 pour plusieurs raisons : le faible nombre de paramètres à entraîner signifie que la phase d'entraînement sera relativement rapide et, à nombre de paramètres équivalents, plus performants que la plupart des autres modèles d'après la figure 8. De plus, ce modèle est inclus dans la librairie TensorFlow de Python ce qui facilite grandement son utilisation. TensorFlow permet également de créer soi-même des modèles . Il est même possible d'importer les paramètres qui ont été calibrés sur le jeu de données ImageNet, pour effectuer du *transfer learning*.

Le *transfer learning* consiste à entraîner un modèle de deep learning sur un jeu de données généraliste, comme ImageNet, puis à utiliser les valeurs des paramètres obtenues pour initialiser l'entraînement du modèle sur un nouveau jeu de données. Cette technique permet d'améliorer les résultats du modèle du fait de l'initialisation *a priori* des paramètres, qui ont appris à repérer des descripteurs généraux dans des images très différentes.

En pratique, pendant une première phase d'entraînement les poids des couches à convolutions du modèle pré-entraîné vont être conservés tels quels, on dit que les couches sont gelées. Durant cette phase de *transfer learning* seuls les paramètres du réseaux de neurones dense, utilisé pour la classification en fin de modèle, vont changer. Si nécessaire, une

seconde phase peut être ajoutée, pendant laquelle les poids sont tous dégelés et le *learning rate* (vitesse d'apprentissage) est réduit. Cette phase est appelée le *fine-tuning*, elle permet de changer légèrement les paramètres des couches de convolutions pour les adapter aux données.

Concernant la construction de notre modèle, nous importons les premières couches d'EfficientNetB0 et leurs poids. Un *transfer learning* a donc été effectué. Les poids importés proviennent d'un entraînement sur les données de ImageNet de 2019. De plus, une phase de *fine-tuning* a été ajoutée. Elle nous permet de stabiliser les prédictions et d'augmenter légèrement la précision des résultats.

L'architecture d'EfficientNetB0 est très complexe. Une description simplifiée est néanmoins disponible en Annexe II. A l'inverse du ML, où les modèles sont des boîtes blanches compréhensibles, les modèles en DL sont des boîtes grises, dans lesquelles des millions de paramètres rentrent dans les calculs des convolutions que contient l'architecture.

Pour la suite, il nous suffit de savoir qu'EfficientNetB0 est composé de trois parties :

- L'entrée : conversion d'une image de 224 x 224 pixels en un tenseur (format de travail de TensorFlow). Suivi d'une augmentation des données (ajoutée manuellement).
- Les convolutions : couches pré-entraînées organisées en 7 blocs, dont le but est de mettre en évidence des caractéristiques à des échelles de plus en plus fines.
- La sortie : à partir des *features* obtenues en sortie du dernier bloc de convolution, utilisation d'un réseau de neurones denses pour déterminer la classe en sortie.

A partir de l'entrée et des convolutions du modèle pré-entraîné EfficientNetB0, on construit le début de notre modèle (voir figure 9). Comme nos vignettes font 201 pixels de côté, il faut redimensionner les images à 224 pixels de côté pour que le modèle puisse utiliser ces vignettes. La sortie doit, elle, être construite à la main car ImageNet contient des centaines de classes à prédire (voiture, arbre, chat, etc.), alors que nous ne cherchons à prédire que 70 espèces végétales. La sortie est donc reconstruite pour limiter les prédictions à 70 classes.

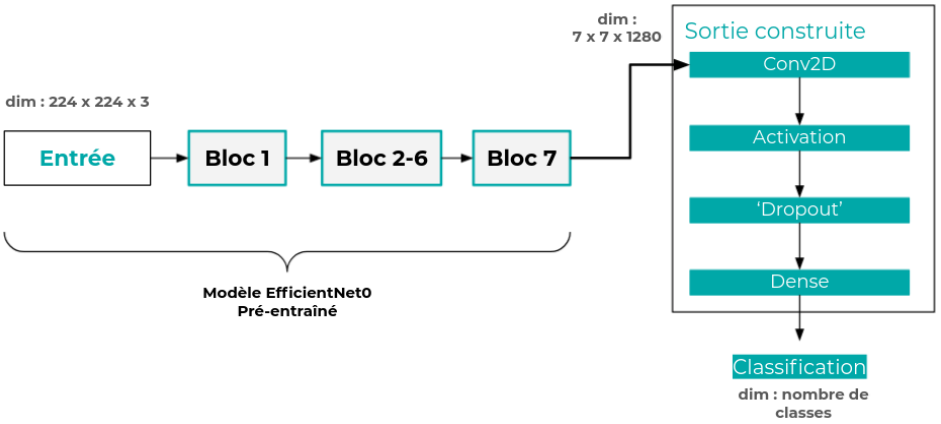


Figure 9 : Représentation schématique du modèle *Ortho-DL*. Source : personnelle

Ainsi, ce modèle que l'on nommera *ortho-DL*, peut être entraîné avec nos ortho-images. Cependant, le DL est également utilisé pour effectuer des prédictions à partir des photographies brutes des zones du fossé. Ces stratégies multi-vues s'appuient sur des versions modifiées du modèle *ortho-DL*.

Stratégies multi-vues :

Les modèles utilisant des données multi-vues ne se développent que depuis peu de temps. Ils consistent à utiliser plusieurs images, qui correspondent au même objet à identifier. Ces images sont données en entrée d'un modèle qui va alors effectuer des convolutions pour chaque image de manière séparée, jusqu'à un point de fusion des couches (voir figure 10) qui permet de compiler les informations obtenues en sortie.

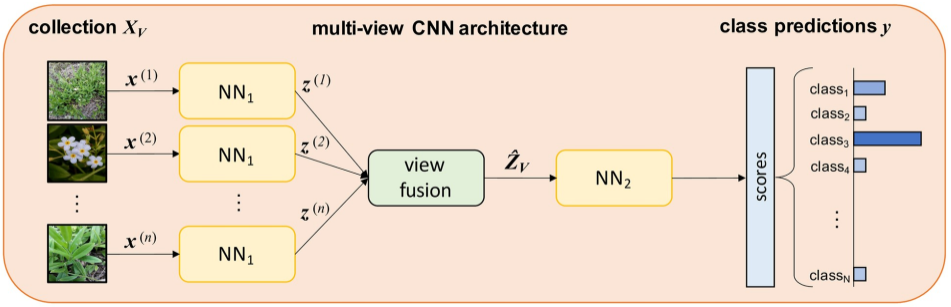


Figure 10 : Architecture générale d'un modèle CNN multi-vues.
NN = couche de convolution. Source : Seeland and Mäder, 2021.

Ces modèles peuvent être divisés en trois catégories selon la position où la fusion est effectuée dans le modèle (Seeland and Mäder, 2021).

- la fusion précoce : a lieu à n'importe quel point entre la première et la dernière couche de convolution. Elle consiste en une fusion de *features* intermédiaires.
- la fusion tardive : a lieu après la dernière couche de convolution, elle intervient juste avant ou juste après la classification par réseau de neurones denses. C'est une fusion des *features* finales obtenues avec les convolutions.
- la fusion des scores : a lieu après avoir obtenu les scores pour classer les images. C'est une fusion des résultats.

D'après les différentes stratégies comparées par Seeland and Mäder (2021), il semble qu'en général, les méthodes permettant d'obtenir les meilleurs résultats sont les stratégies où la fusion est effectuée le plus tardivement possible. C'est pourquoi nous avons choisi de construire deux modèles :

- Un modèle (nommé *mono-entrée-DL*) multi-vues : avec une stratégie naïve où toutes les images brutes sont données en entrée du modèle, sans distinction de leur angle de vue ou de leur orientation. Cela donne des données avec beaucoup de variabilité.

- Un modèle (nommé *multi-entrées-DL*) multi-vues : avec une stratégie de fusion tardive des *features*. Les deux branches du modèle se rejoignent avant l'entrée dans le réseau de neurones denses. La fusion correspond à une concaténation des *features* finales, suivie d'une réduction des dimensions du résultat de cette concaténation pour la rendre utilisable en entrée du réseau de neurones.

L'hypothèse de départ pour justifier l'utilisation de ces modèles se base sur la complexité des données à utiliser. Dans un fossé agricole, les plantes *in situ* ne sont pas aussi faciles à détecter ni à identifier que lorsque les photographies sont prises dans un champ, ou en laboratoire. La densité de la végétation, ainsi que sa diversité, rendent les prédictions difficiles à réaliser.

Les avantages hypothétiques à utiliser plusieurs vues d'une même plante, sont les suivants : réduction des occlusions de plantes dues à la densité de végétation et meilleure identification en vue de profil, notamment avec la possibilité de mieux différencier le port des plantes. C'est pourquoi nous espérons obtenir de meilleurs résultats avec les modèles multi-vues. Cependant, les photographies multi-angles ne sont pas toutes exploitables : les photographies prises au nadir du fossé sont, par exemple, souvent mal cadrées. Pour cette raison, seules deux orientations sont retenues pour le modèle *multi-entrées-DL* : la 2 et la 6 (voir figure 2), car ce modèle nécessite deux vues d'une exacte même coordonnée.

2.3. Un jeu de données déséquilibré

2.3.1. Échantillonnage par *downsampling*

Le jeu de données est composé de 4 ortho-images. Au total, ce sont quasiment 50 millions de pixels qui peuvent être considérés comme des individus statistiques à utiliser pour construire et entraîner nos modèles. Cependant, sur les vérités terrain on retrouve une surreprésentation de certaines classes. Par exemple, *Equisetum ramosissinum* est l'espèce végétale la plus représentée : elle couvre toute la zone d'étude et peut être considérée comme l'arrière-plan de l'image car presque 50% de la vérité terrain peut être attribuée à cette espèce.

Le jeu de données contient donc des classes avec des effectifs très variables. Cette situation est problématique pour l'entraînement des modèles. Pour caricaturer, il suffit de prédire toujours la même classe (l'arrière-plan) pour que la précision du modèle soit de 50%. Pour prédire des proportions d'espèces végétales, cela biaise fortement les résultats.

Pour résoudre ce souci, trois solutions sont courantes : le *downsampling* (sous-échantillonnage), l'*upsampling* (suréchantillonnage) ou l'utilisation de pondérations des paramètres selon les classes.

Pour la pondération, on peut par exemple affecter un poids inversement proportionnel à la fréquence de chaque classe pour gérer l'influence des différentes classes dans le calibrage du modèle. Mais cette méthode souffre d'un grand défaut : en DL et en ML, ces poids sont des hyper-paramètres supplémentaires. Ils augmentent la complexité du problème et donc amoindrissent souvent les capacités prédictives. C'est pourquoi nous ne l'avons pas choisie.

En ce qui concerne l'*upsampling* et le *downsampling*, ces techniques consistent à sélectionner un échantillon de données qui contient respectivement plus d'individus d'une classe peu représentée ou moins d'individus d'une classe sur-représentée. Cela permet de modérer la variabilité dans le nombre d'individus de chaque classe directement dans la phase de création des jeux de données.

Sachant que l'arrière-plan, dans notre jeu de données, est surreprésenté par rapport à toutes les autres espèces végétales, nous avons opté pour la méthode de *downsampling*. Le script Python de sélection des données applique donc cette méthode pour sélectionner 10 000 individus, aléatoirement, dans chaque ortho-image.

Pour les photographies, la sélection des pixels à utiliser se fait par correspondance des coordonnées géographiques des pixels sélectionnés sur l'ortho-image, aux coordonnées des pixels dans les images brutes. Cette sélection est effectuée en utilisant de l'algèbre relationnelle, ce qui nous assure que nos requêtes sont cohérentes et relativement rapides.

2.3.2. Jeux d'entraînement, de test et de validation

Les modèles de ML se basent sur l'utilisation de jeux de données divisés usuellement en deux catégories : un jeu de données d'entraînement (ou *training set*) et un jeu de données de test (ou *test set*). Pour les modèles de DL spécifiquement, un jeu de données de validation (ou *validation set*) est ajouté.

Ces jeux de données sont obtenus en sélectionnant un échantillon d'individus aléatoires parmi toutes les données disponibles. Un jeu de données d'entraînement est l'échantillon utilisé pour calibrer les paramètres du modèle afin d'optimiser ses prédictions. Une fois les combinaisons optimales de paramètres trouvés, le modèle est calibré. Pour le tester, on utilise le jeu de données de test : c'est un échantillon prélevé dans le jeu de données original, sans répétition avec le *training set*. Il permet d'évaluer le modèle sur des données différentes, ce qui permet de tester sa capacité à généraliser.

Pour le DL, le jeu de données validation remplit un rôle mixte. Il permet à la fois de tester le modèle, mais il permet aussi de calibrer les hyper-paramètres du modèle : son architecture, le *learning rate*, les *epochs*, le *batch size* (ou taille du lot). C'est également un indicateur de l'*overfitting* du modèle aux données d'entraînement.

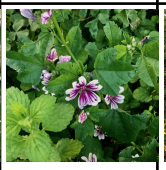
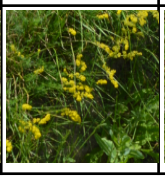
Parmi toutes nos données, nous utilisons uniquement les photographies et vérités terrain collectés lors de la campagne de 2016, car ce sont les plus fiables. Après construction des modèles, d'autres campagnes pourront être utilisées si besoin.

Pour nos données, après *downsampling*, les individus sont permutés aléatoirement avant d'être répartis dans les différents *sets*. Sur 2 500 pixels sélectionnés par zone, 50% sont sélectionnés pour faire partie de *training set*, et 50% pour le *test set*. Dans le cas du DL, le *validation set* est constitué à partir d'un échantillon de 25% du *training set*. Cela permet de s'assurer que les jeux de données de test comparent les mêmes coordonnées, afin d'évaluer les modèles sur les mêmes données.

De plus, seules les zones M1, M2 et M3 participent à la création du jeu de données d'entraînement. Les 2 500 pixels sélectionnés dans M4 sont affectés à un quatrième jeu de données nommé *excluded set*. Ce jeu de données totalement inconnues des modèles permettra de les tester dans un scénario complexe. En effet, M4 contient des variations de contraste et de luminosité rendant la reconnaissance d'espèces végétales plus difficile que pour M1-3.

Enfin, les modèles ont été testés sur deux vérités terrain différentes. La première contient la distribution d'une seule espèce végétale : *Knautia arvensis*, très reconnaissable à l'œil nu sur les images, cette vérité terrain permet de valider le fonctionnement des modèles. La seconde vérité terrain contient la distribution de 7 espèces végétales (voir tableau 1) : la multiplication du nombre de classes est un défi qui permet de tester les capacités des modèles à généraliser. La vue aérienne des plantes est également un challenge pour l'identification, comparé à des données centrées et vues de côté telles celles de Pl@ntNet (tableau 1).

Tableau 1 : Espèces végétales représentées dans nos données.

Nom vernaculaire de l'espèce	Gaillet blanc	Gaillet jaune	Knautie des champs	Mauve sylvestre	Millepertuis perforé	Molène blattaire	Rumex crépu
Image Pl@ntNet							
Image jeu de données							

Sources : Consortium Pl@ntNet (2014) et photographies brutes de notre jeu de données

2.3.3. Des métriques d'évaluation adaptées à des données déséquilibrées

La métrique classiquement utilisée afin de tester les modèles de ML et DL est la précision (ou *accuracy*). Cependant, malgré le *downsampling* des classes surreprésentées, il reste une certaine variabilité dans les effectifs des classes de nos données.

Pour éviter un biais dans nos interprétations, nous utilisons des métriques prenant en compte les effectifs de chaque classe. Ces métriques sont :

- Le Kappa de Cohen : mesure l'accord entre les observateurs.
- La précision équilibrée : pourcentage de précision pondéré.
- Le F1-score : évalue la précision sans compter le nombre de vrais négatifs.
- Le score de Jaccard : mesure le pourcentage de similarités entre deux jeux de données.
- Le coefficient de corrélation de Matthews (ou CCM) : résume les coefficient d'une matrice de confusion entre les jeux de données.

Ces scores varient tous entre 0 et 1, où 1 est un score signifiant un parfait accord des prédictions avec les données. L'utilisation de plusieurs scores mesurant les résultats de manière relativement identique permet de dégager une tendance, qui permettra de valider les comparaisons de modèle. Plus d'informations sur le calcul de ces scores en annexe III.

De plus, les résultats peuvent également être comparés sur la base de leurs matrices de confusions. Cela permettra de raisonner sur la précision des résultats de chaque classe.

Enfin, pour *Ortho-ML* et *Ortho-DL*, les prédictions sont projetées sur l'emprise des ortho-images et peuvent être comparées avec les vérités terrain. Ceci permet ensuite d'appliquer des transformations morphologiques aux prédictions, afin de transformer pour les objets/régions obtenus par segmentation sémantique (et donc d'améliorer les résultats). Ces transformations sont obtenues grâce à des dilatations ou des érosions : des opérations qui permettent de supprimer le bruit autour des formes dessinées. Ce sont des opérations de traitement d'image réalisées grâce à la librairie OpenCV.

3. Résultats

3.1. Modèles *Ortho-ML* et *Ortho-DL*

Pour ces modèles, les données d'entrées sont les ortho-images. *Ortho-ML* s'appuie sur des *features* générées à partir des ortho-images entières pour extraire des informations par pixels, qui sont données entrée d'un RF. *Ortho-DL* s'appuie sur des vignettes découpées dans l'ortho image avec au centre, le pixel dont la classe est à prédire. Ces vignettes sont directement données en entrée d'EfficientNetB0.

En ce qui concerne les paramètres utilisés : le nombre d'arbres est limité à 250 pour le RF. Pour *Ortho-DL*, on considère 75 epochs en *transfer learning* puis 25 epochs en *fine-tuning*, un batch size de 32 et un *learning rate* planifié. Le *learning rate* est de 10^{-3} sur les 25 premiers epochs puis il diminue progressivement pour atteindre 10^{-5} en *fine-tuning*.

Tableau 2 : Scores de prédiction associés au modèle *Ortho-ML*.

Type de données	Binaire		7 espèces	
Jeu de donnée	Test	Excluded	Test	Excluded
Précision équilibrée	**0.929**	0.533	0.885	0.211
Kappa de Cohen	**0.859**	0.07	0.772	0.115
F1-score	**0.931**	0.513	0.801	0.228
Score de Jaccard	**0.870**	0.374	0.689	0.140
CCM	**0.858**	0.099	0.772	0.123

Avec ces premiers résultats (tableau 2), on constate que le modèle *Ortho-ML* fait de bonnes prédictions pour les jeux de données test (quel que soit le type de données) mais que les scores sur les données *excluded* sont beaucoup plus faibles.

Les scores en gras dans les tableaux mettent en évidence les meilleurs scores entre tableau 2 et tableau 3.

Tableau 3 : Scores de prédiction associés au modèle *Ortho-DL*.

Type de données	Binaire		7 espèces	
Jeu de donnée	Test	Excluded	Test	Excluded
Précision équilibrée	0.905	**0.826**	**0.902**	**0.367**
Kappa de Cohen	0.819	**0.666**	**0.826**	**0.303**
F1-score	0.911	**0.834**	**0.852**	**0.364**
Score de Jaccard	0.838	**0.720**	**0.751**	**0.269**
CCM	0.822	**0.674**	**0.826**	**0.331**

Les scores obtenus avec ce modèle *Ortho-DL* (tableau 3) sont presque aussi bons que *Ortho-ML* pour les données de test. En effet, pour les données de test binaires, le DL obtient de moins bons scores que le ML. Néanmoins, les scores de ce second modèle sont meilleurs que ceux du ML pour toutes les autres données, dont les données *excluded*. On remarque surtout une différence de scores entre test et *excluded* (pour chaque type de données) moins grande avec le modèle *Ortho-DL*.

Dans les tableaux, on remarque qu'individuellement, tous les scores suivent les mêmes variations en fonction du jeu de données sur lequel le modèle est évalué.

Pour accompagner les scores, qui sont une représentation abstraite des distributions des espèces végétales à prédire. On peut considérer des matrices de confusion ou, pour encore mieux visualiser les résultats, les reprojections des prédictions sur l'ortho-image.

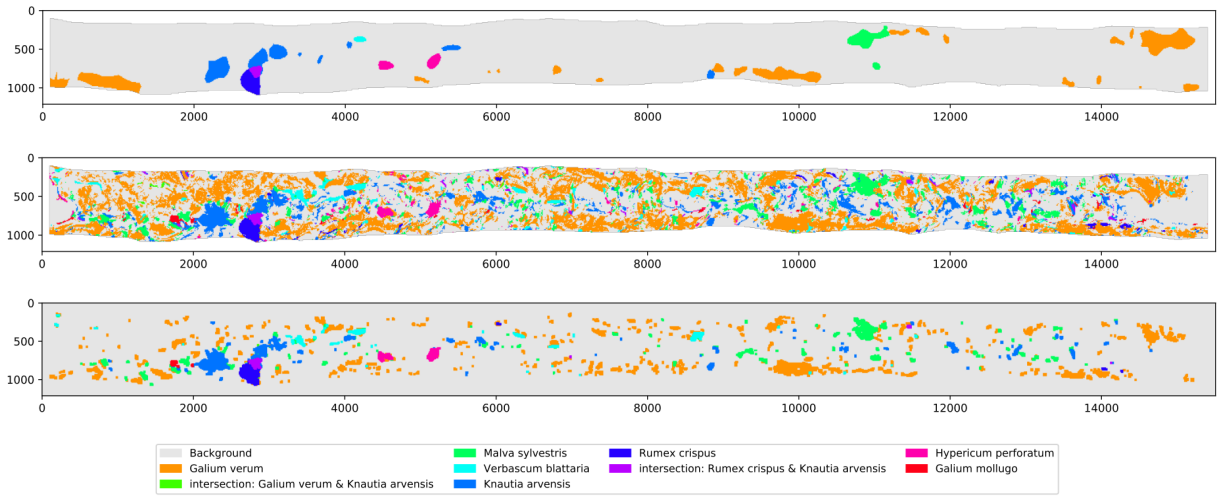


Figure 11 : Prédictions *Ortho-ML* des répartitions des espèces végétales sur l'ortho-image de M2 (*test set*). (Haut) Vérité terrain. (Milieu) Prédiction brute. (Bas) Prédiction après traitement morphologique.

Sur la représentation brute (figure 11) on remarque que la prédiction *Ortho-ML* est désorganisée. Certaines distributions sont prédites avec précision : 97% pour le *Rumex crispus*

et 98% pour son intersection avec *Knautia arvensis*. Cependant, la prédiction brute contient au total 25.6% de pixels mal classés.

Le traitement morphologique de l'image est efficace : il permet de se rapprocher des distributions de la vérité terrain mais il est brutal et produit des formes carrées sur l'image. Ces représentations ne sont pas aussi propres que celles obtenues avec le modèle *Ortho-DL*.

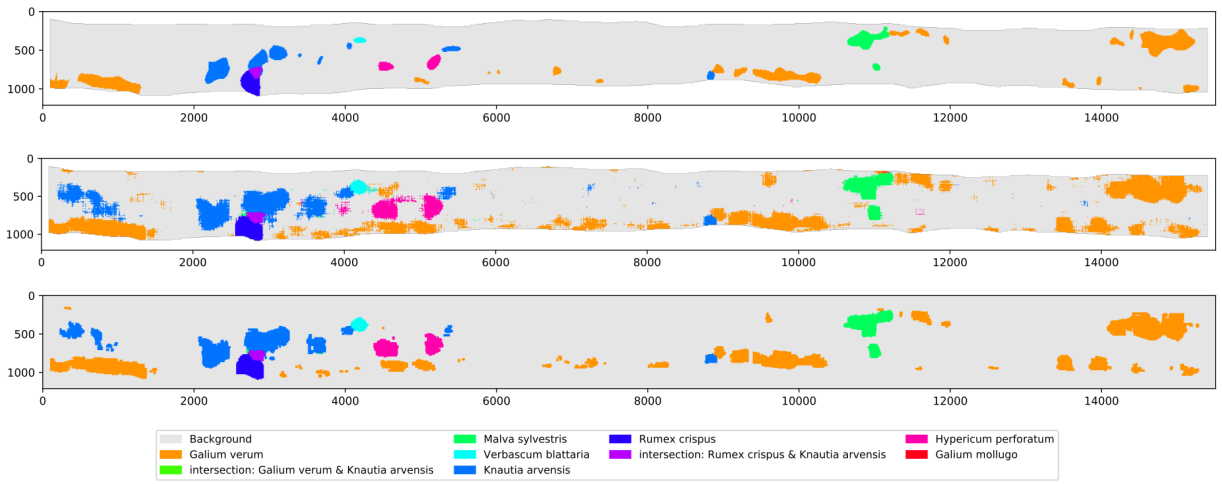


Figure 12 : Prédictions *Ortho-DL* des répartitions des espèces végétales sur l'ortho-image de M2 (*test set*). (Haut) Vérité terrain. (Milieu) Prédiction brute. (Bas) Prédiction après traitement morphologique.

Le modèle *Ortho-DL* obtient un meilleur score que le modèle *Ortho-ML* sur les jeux de données avec 7 espèces (voir tableau 2). Cela se retranscrit dans la représentation figure 12, qui montre que le classement de chaque pixel s'approche bien de la vérité terrain. Sans artefacts comme s'était visible sur la figure 11. De plus, le traitement des formes sur l'image ne modifie pas énormément les prédictions mais enlève le "bruit", permettant d'obtenir un rendu plus propre. D'autres images issues des prédictions sont disponibles en Annexe IV.

Il est à noter que, pour le DL, le temps de prédiction est beaucoup plus long. Pour une ortho-image, il faudrait plusieurs jours si tous les pixels devaient être classés. Pour contrer cela, seul 1 pixel sur 5 a été classé dans chaque direction, soit 1 pixel sur 25 au total. Cela permet de réduire le temps de prédiction à quelques heures pour chaque ortho-image. Cependant, cela ne rivalise pas avec les 2 minutes 30 nécessaires au ML pour prédire une image entière.

3.2. Prédictions photographies brutes : mono-entrée-DL

Pour ce modèle de DL, on considère 50 epochs en *transfer learning* puis 25 epochs en *fine-tuning*, un batch size de 8 et un *learning rate* planifié. Le *learning rate* est de 5.10^{-4} sur les 25 premiers epochs puis il diminue progressivement pour atteindre 10^{-5} en *fine-tuning*. Ces hyperparamètres sont différents du modèle *Ortho-DL* car les données utilisées sont différentes et il a été remarqué que le modèle convergeait plus vite vers des prédictions stables avec ces changements.

Pour ce modèle, aucune représentation projetée sur l'ortho-image n'a pu être réalisée, principalement par manque de temps. C'est pourquoi seuls les scores (tableau 4) sont montrés. Ils suivent la même tendance que les tables précédentes avec des scores légèrement inférieurs.

Tableau 4 : Scores de prédiction associés au modèle *mono-entrée-DL.*

Type de données	Binaire		7 espèces	
Jeu de donnée	Test	Excluded	Test	Excluded
Précision équilibrée	0.853	0.753	0.760	0.336
Kappa de Cohen	0.698	0.517	0.710	0.248
F1-score	0.853	0.765	0.770	0.328
Score de Jaccard	0.744	0.622	0.633	0.244
CCM	0.700	0.523	0.715	0.302

3.3. Prédictions photographies brutes : multi-entrées-DL

Pour ce dernier modèle de DL, les mêmes hyper-paramètres que le modèle *mono-entrée-DL* sont conservés. De même, seuls le tableau des scores est montré ici (tableau 5). Les scores suivent la même tendance que pour le modèle *mono-entrée-DL* cependant les scores de test sont supérieurs et les scores *excluded* sont, eux, inférieurs.

Tableau 5 : Scores de prédiction associés au modèle *multi-entrées-DL.*

Type de données	Binaire		7 espèces	
Jeu de donnée	Test	Excluded	Test	Excluded
Précision équilibrée	0.884	0.575	0.860	0.346
Kappa de Cohen	0.750	0.123	0.755	0.100
F1-score	0.879	0.452	0.796	0.171
Score de Jaccard	0.784	0.298	0.670	0.104
CCM	0.754	0.187	0.758	0.123

3.4. Matrices de confusion

Les matrices de confusion permettent de résumer beaucoup d'informations. Les figures 13 et 14 représentent les matrices de confusion de chaque modèle sur le jeu de données *excluded* avec 7 espèces. C'est le jeu de données le plus difficile à prédire. Des résultats complémentaires, non nécessaires pour l'analyse, sont disponibles en Annexe IV.

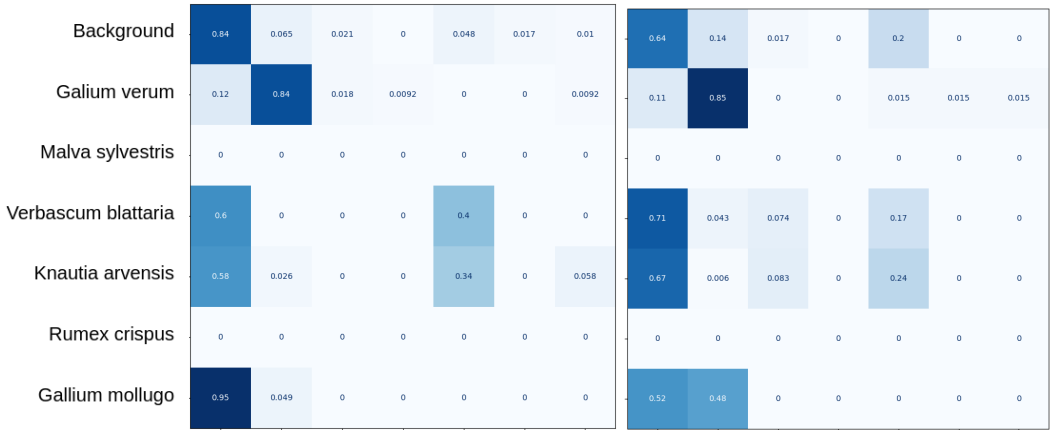


Figure 13 : Matrices de confusion *Mono-entrée-DL* (Gauche) *Multi-entrées-DL* (Droite).
Produites à partir de prédiction sur le jeu de données *excluded* à 7 espèces.

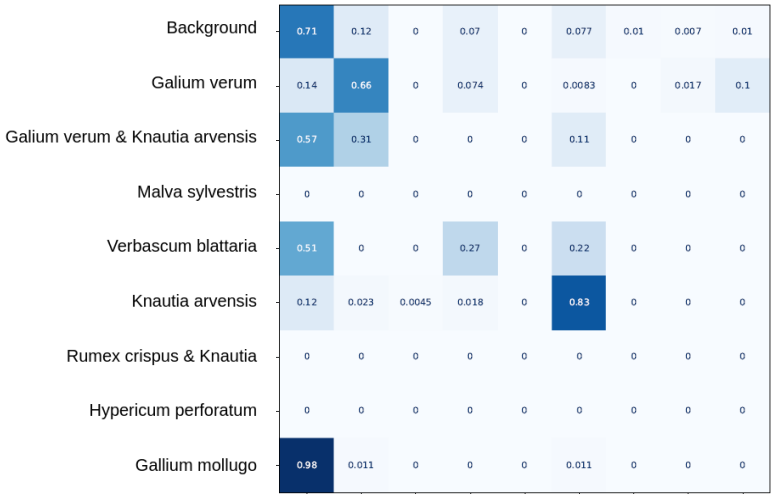


Figure 14 : Matrice de confusion pour *Ortho-DL*.
produites à partir de prédiction sur le jeu de données *excluded* à 7 espèces.

4. Discussion

4.1. Interprétation des résultats

Comparaison *Ortho-ML* et *Ortho-DL* :

D'après les scores résumés dans les tableaux 2 et 3, il semble que le ML soit moins bon que le DL sauf sur les données de test binaire. Cela illustre deux choses : la première c'est un *overfitting* de notre modèle de RandomForest aux données d'entraînement. Cela amène *Ortho-ML* à obtenir de très bons scores avec notamment une précision équilibrée de près de 93%. Mais cet *overfitting* le rend totalement inutile sur des données inconnues et plus complexes : celles du jeu de données *excluded* binaire. En effet, le kappa de Cohen des prédictions du modèle *Ortho-ML* sur ces données est de 7% contre 67% pour les prédictions du modèle *Ortho-DL*.

Ensuite, on remarque une supériorité des scores du DL sur le ML pour les jeux de données contenant 7 espèces. Les scores du DL sur le jeu de données de test sont supérieurs d'en moyenne 0,05 points à ceux du ML, et supérieurs d'environ 0,16 points en moyenne sur

les données de M4. Cette observation illustre la malédiction des dimensions : lorsque l'espace dans lequel les données sont confinées grandit, il devient plus difficile de trouver des règles pour les regrouper. Les modèles de DL ont étés créés pour gérer de grands jeux de données et il est donc attendu qu'ils performent mieux que des modèles de ML lorsque le nombre de classes, et donc la complexité d'un jeu de données, augmente.

Ainsi, *Ortho-ML* est capable de faire des prédictions d'une grande précision. Mais il souffre d'*overfitting*, et ne parvient pas à maintenir de bons résultats lorsque le nombre de classes à prédire augmente. De son côté *Ortho-DL* est capable de mieux généraliser, les scores varient moins lorsque le jeu de données change : ils diminuent de moins de 20% sur le jeu de données binaire contre minimum 40% pour *Ortho-ML*.

Enfin, une comparaison rapide des ortho-images prédites montre que le ML a tendance à prédire des espèces là où il n'y en a pas (Figure 11). Malgré une étape de post-traitement des images lourdes (c'est la raison pour laquelle les formes finissent par ressembler à des carrés), les distributions des espèces restent peu en accord avec la réalité. En revanche pour le DL (figure 12) les distributions ressemblent déjà beaucoup, avant post-traitement, aux répartitions à prédire. Cet avantage a un prix : le temps prédiction est beaucoup plus long pour *Ortho-DL*, et passe en plus par une interpolation.

Sachant que ces modèles sont voués à être utilisés sur de grands jeux de données comprenant jusqu'à 70 classes d'espèces végétales dans notre cas, *Ortho-ML* semble être le modèle avec le plus de défauts. C'est donc *Ortho-DL* qui va permettre de construire les modèles multi-vues.

Comparaison *mono-entrée-DL*, *multi-entrées-DL* :

Les résultats des modèles *mono-entrée-DL* et *multi-entrées-DL* disponibles dans les tableaux 4 et 5 montrent des comportements tout à fait différents de ces modèles sur chaque jeu de données.

Le modèle *mono-entrée-DL* performe bien sur les jeux de données *test* et *excluded* binaires, et on note notamment que les scores de ce modèle ne perdent en moyenne que 0,14 points pour des prédictions entre un jeu de données relativement simple (test) et le jeu de données totalement inconnu *excluded*. De même, moins de 0,05 points sont perdus lors du passage du jeu de données de test binaire à celui contenant plusieurs espèces.

Ces résultats témoignent d'une très bonne capacité du modèle à généraliser. Ils étaient attendus compte-tenu du fait que les données d'entrées de ce modèle proviennent de photographies avec beaucoup de variabilité : en plus des variations de contraste et luminosité des photographies brutes, il y a des variations dans l'angle de la prise de vue. L'entraînement d'un modèle de DL sur de telles données amène à une bonne résistance à la variabilité.

Le modèle *multi-entrée-DL* performe mieux que le *mono-entrée-DL* dans deux situations : sur les jeux de données de test binaire et 7 espèces avec respectivement 0,04 et

0,05 de précision en plus par rapport au *mono-entrée-DL* en moyenne. C'est l'inverse pour les jeux de données *excluded*, avec une réduction de 0,31 et 0,12 soit une diminution des scores de 48% et 42%.

Cet effondrement des prédictions suite au passage au jeu de données *excluded* alors que les résultats semblaient prometteur sur les données de test révèle encore une fois un *overfitting*. La raison pour cela pourrait être les contraintes imposées aux données. En effet, le modèle est entraîné sur une sélection de photographies brutes dont les angles de prise de vue sont toujours les mêmes.

De plus, contrairement à ce que nous pouvions penser, les modèles ne parviennent pas à gérer efficacement les occlusions. En effet, d'après la matrice de confusion calculée sur des données issues de M4 (Figure 13), *Verbascum blattaria* et *Knautia arvensis* sont toutes deux plutôt mal prédites. Or la première s'étend en hauteur, bien visible, alors que la seconde s'étale sur le sol, parfois cachée par d'autres plantes. On espérait donc avoir de meilleures prédictions sur la knautie avec le modèle *multi-entrées-DL*, pouvant avoir deux angles de vue sur cette plante, afin de confirmer notre hypothèse, ce qui n'est pas le cas.

Dans le détail, les trois modèles de DL peuvent être comparés à partir de leurs matrices de confusion issues des prédictions le jeu de données *excluded* (Figure 13 et 14). Pour commencer, le nombre de classes prédites par chaque modèle est différent. Concernant *Ortho-DL*, la matrice présente des valeurs de confusion pour 9 classes alors que seules 7 classes sont présentes dans les matrices de confusion des modèles multi-vues. Cela démontre que le modèle *Ortho-DL* généralise beaucoup pour effectuer ses prédictions, ce qui peut amener à la production de faux positifs.

En regardant chaque espèce, on remarque que les modèles ont des difficultés à prédire les mêmes classes. Plus précisément, les modèles multivues prédisent par exemple le *Galium Verum* correctement 84% à 85% du temps, mais seulement 66% du temps pour le modèle *Ortho-DL*. A l'inverse, les modèles multi-vues ont beaucoup de mal à reconnaître *Knautia arvensis* avec moins de 35% de prédictions correctes alors que *Ortho-DL* permet de reconnaître cette espèce 83% des fois où elle est présente sur l'image. Les données d'entrée jouent donc un rôle, suivant les espèces, sur les capacités de prédiction. Il peut aussi s'agir du fait qu'un angle de vue au nadir (ortho-image) rend plus difficile l'identification de plantes dont le port s'étend en hauteur.

Finalement, ces résultats révèlent que le modèle *Ortho-DL* est celui qui obtient les meilleurs scores, pour détecter, identifier et quantifier les espèces végétales globalement. Il semble donc être le meilleur pour effectuer prédire les distributions spatialisées d'espèces végétales. L'*overfitting* est un problème récurrent pour tous les modèles qui ont été construits. Il peut être résolu de plusieurs manières : en augmentant la variabilité des données en entrée, ce qui est possible en utilisant les photographies de plusieurs campagnes de collecte de données, ou en faisant varier la structure ou les hyper-paramètres des modèles.
Cependant dans le détail, l'identification de certaines espèces peut-être facilitée par l'utilisation de différents types de données, par différents modèles.

4.2. Limites des résultats

Les modèles construits peuvent être évalués de nombreuses manières : par leurs scores, leurs matrices de confusion ou encore par la comparaison des distributions des communautés végétales prédites aux vérités terrain cartographiées. Cependant, ce dernier résultat n'est pour l'instant disponible que pour les modèles *Ortho-ML* et *Ortho-DL*. Il serait très simple de poser des règles permettant d'obtenir des distributions projetées sur les ortho-images pour les modèles multi-vues. Ce résultat permettrait également de mieux comparer les performances des modèles, car les modèles multi-vues disposent par définition de plusieurs vues pour prédire un seul pixel sur l'ortho-image, ce qui pourrait améliorer leurs performances réelles. Cependant, par manque de temps, cela n'a pas pu être mis en place.

Concernant ces mêmes distributions, au vu de l'aspect très désordonné des prédictions brutes du modèle *Ortho-ML* (Figure 11), on est en droit de se demander pourquoi les scores associés (Tableau 2) ne sont pas encore moins bons. La raison est simple, elle réside dans l'échantillonnage : le *downsampling* réduit l'effectif de la classe contenant l'arrière-plan dans le jeu de données test. Ainsi, seul un échantillon de cette classe est mal classé dans le tableau des scores. Alors que sur la vérité terrain 50% de l'image est constituée de cette classe.

Également par contrainte de temps, les modèles présentés ici n'ont pas pu être calibrés parfaitement. En ce qui concerne le ML, des caractéristiques moins généralistes auraient pu être utilisées afin d'extraire de meilleures informations des images. Notamment, avec des extractions de couleurs et de formes plus spécifiques (Bakhshipour and Jafari, 2018). Mais cela aurait ralenti d'autant plus la génération des *features* durant le pré-traitement de l'image.

En ce qui concerne le DL, les modèles manquent d'un choix plus fin des valeurs des hyper-paramètres. Un algorithme d'optimisation simple peut-être lancé pour déterminer le *batch size*, les itérations ou le *learning rate* les plus adaptés à chaque modèle. Mais chaque phase d'entraînement nécessitant plusieurs heures, cette méthode n'a pas été appliquée.

Comme mentionné en partie 2.2.2., les prédictions faites avec *Ortho-ML* sont effectuées en quelques minutes alors qu'il faut plusieurs heures aux modèles de DL pour prédire les mêmes images. Cependant puisque le pré-traitement des images prend 2,5 jours pour le ML afin de générer les *features*, les modèles de DL restent plus rapides. De plus, le DL rend possible la réalisation de calculs en parallèle sur une carte graphique (GPU) d'un ordinateur. L'utilisation de GPU plus performants, ou de plusieurs GPUs pourrait donc accélérer considérablement la vitesse de prédiction.

Concernant les données, parmi les 7 angles de vue (voir figure 2), certains étaient inutilisables. En effet, l'utilisation des photographies prises au nadir peut sembler intéressante pour comparaison avec l'ortho-image mais cela n'a pas pu être possible. Avoir deux angles de vue très différents : un incliné à 20° et un autre perpendiculaire au sol en entrée d'un modèle multi-vues permettrait peut-être d'obtenir de meilleurs résultats. Les prédictions d'un modèle dépendent beaucoup de son jeu de données d'entraînement. Et nos données n'ont pas encore été totalement exploitées.

4.3. Perspectives

Réutilisation des résultats :

Au vu de l'efficacité de la détection en vue aérienne (avec l'ortho-image), on peut imaginer adapter la récolte de données pour ne plus avoir à la faire manuellement. En effet, il est tout à fait possible d'utiliser un drone, pour automatiser le processus de collecte des données, ce qui permet une spatialisation des distributions exhaustive, et non plus réduite à un fossé agricole méditerranéen.

On peut alors imaginer une prédiction séquentielle pour automatiser partiellement les relevés botaniques sur le terrain : collecte de photographies par drone, prédiction des résultats et évaluation des incertitudes (e.g. basée sur les probabilité d'occurrence des différentes classes). Puis, un utilisateur peut retourner sur le terrain prendre des photos au plus près des régions incertaines, qui seront identifiées par un algorithme de DL ou via Pl@antNet.

L'avantage de la prédiction par pixels est que des post-traitements sont possibles pour extraire des métriques mono-espèces ou d'assemblage. A partir de ce travail, de nombreux estimateurs écologiques peuvent être calculés comme des indices de diversité. La spatialisation des données permet aussi l'étude de concurrences spatiales.

Partenariat avec Pl@ntNet :

Les données utilisées pour ce travail ne sont pas si différentes des données photographiques de la base de données de Pl@ntNet. Via un accès à leur API (Interface de Programmation Applicative), il est possible de fournir les vignettes de nos ortho-images au modèle de DL de Pl@ntNet afin d'obtenir une prédiction des distributions des espèces sur l'ortho-image. Cela permet ainsi de d'utiliser un modèle déjà entraîné à reconnaître plusieurs milliers d'espèces différentes, une idée s'approchant d'un *transfer learning*.

Quelques tests avec des photographies de *Rumex crispus* et *Malva sylvestris* ont démontré que l'utilisation de cette API était prometteuse. Cependant, le passage à la prédiction d'une ortho-image entière implique de devoir effectuer une interrogation massive de leur API, et donc plus de discussions avec leur équipe.

Boîtes englobantes :

La génération des vérités terrain permet d'extraire les *bounding box* autour de chaque polygone présent dans les *shapefiles*. Cette information permet d'effectuer de la détection d'objet (Figure 6), notamment avec des algorithmes tels que YOLO (Redmon *et al.*, 2016). Quelques tests nous ont permis de confirmer que ce type de modèles était applicable à nos données, mais un approfondissement est encore nécessaire. La détection d'objet permet d'obtenir des informations telles que la localisation d'une communauté végétale dans l'image et son nombre d'instances. Ces informations peuvent permettre de compléter des résultats obtenus par segmentation sémantique.

5. Conclusion

L'objectif de ce travail était de produire des IA capables de prédire la distribution d'espèces végétales dans les fossés agricoles méditerranéens. Il concernait aussi l'exploration des apports de l'utilisation de données multi-vues. Dans ce cadre, une stratégie de prédiction par segmentation sémantique a été explorée, elle permet d'attribuer une classe à chaque pixel d'une image et permet ainsi de cartographier des distributions d'espèces végétales.

Après la production de vérités terrain pouvant être exploitées par des modèles de *machine learning*, 4 modèles ont étés construits : *Ortho-ML* et *Ortho-DL*, utilisant seulement une ortho-image pour réaliser leurs prédictions, ainsi que *mono-entrée-DL* et *multi-entrée-DL*, deux modèles multi-vues qui utilisent deux photographies d'une même coordonnée terrain pour effectuer leurs prédictions.

Ces modèles permettent de prédire la distribution d"espèces végétales dans les fossés de manière satisfaisante. Le modèle *Ortho-DL* est le plus performant avec une précision équilibrée de plus de 90% pour détecter et identifier une espèce végétale dans une ortho-image. Ce modèle est également capable de généraliser son apprentissage et donne de bons résultats lorsque le nombre d'espèces à prédire augmente.

Les modèles multi-vues, utilisant des images multi-angles, sont moins performants globalement, et peuvent présenter un certain *overfitting*. Ils montrent cependant des avantages pour la prédiction de la distribution de certaines espèces. Le couplage des prédictions de ces deux types de modèles peut permettre de mettre en place des règles de décision pour améliorer les prédictions. De même, un jeu de données mixte couplant des photographies en vue aérienne et vue au sol pourrait être utilisé dans un modèle multivues, permettant ainsi de coupler les avantages des modèles construits durant ce stage.

Ce travail a ouvert de nombreuses perspectives et applications. Les modèles sont généralistes, facilitant leur application à d'autres jeux de données. Les résultats sous forme d'ortho-images projetées permettent de spatialiser les espèces végétales et déduire avec précision des propriétés hydrologiques et écologiques du fossé.

Bibliographie

Références :

Anubha Pearline, S., Sathiesh Kumar, V. and Harini, S. (2019) 'A study on plant recognition using conventional image processing and deep learning approaches', *Journal of Intelligent & Fuzzy Systems*. Edited by S. M. Thampi and E.-S. M. El-Alfy, 36(3), pp. 1997–2004. doi: 10.3233/JIFS-169911.

Bakhshipour, A. and Jafari, A. (2018) 'Evaluation of support vector machine and artificial neural networks in weed detection using shape features', *Computers and Electronics in Agriculture*, 145, pp. 153–160. doi: 10.1016/j.compag.2017.12.032.

Dollinger, J. *et al.* (2015) 'Managing ditches for agroecological engineering of landscape. A review', *Agronomy for Sustainable Development*, 35(3), pp. 999–1020. doi: 10.1007/s13593-015-0301-6.

Goëau, H. *et al.* (2014) 'PlantNet Participation at LifeCLEF2014 Plant Identification Task', *CLEF: Conference and Labs of the Evaluation Forum*, pp. 724-737. Available at: https://hal.archives-ouvertes.fr/halsde-01064569

Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2017) 'ImageNet classification with deep convolutional neural networks', *Communications of the ACM*, 60(6), pp. 84–90. doi: 10.1145/3065386.

Olsen, A. *et al.* (2019) 'DeepWeeds: A Multiclass Weed Species Image Dataset for Deep Learning', *Scientific Reports*, 9(1), p. 2058. doi: 10.1038/s41598-018-38343-3.

Redmon, J. *et al.* (2016) 'You Only Look Once: Unified, Real-Time Object Detection', *arXiv:1506.02640 [cs]*. Available at: http://arxiv.org/abs/1506.02640

Rubol, S., Ling, B. and Battiato, I. (2018) 'Universal scaling-law for flow resistance over canopies with complex morphology', *Scientific Reports*, 8(1), p. 4430. doi: 10.1038/s41598-018-22346-1.

Rudi, G. *et al.* (2020) 'Multifunctionality of agricultural channel vegetation : A review based on community functional parameters and properties to support ecosystem function modeling', *Ecohydrology & Hydrobiology*, 20(3), pp. 397–412. doi: 10.1016/j.ecohyd.2020.03.004.

Safonova, A. *et al.* (2019) 'Detection of Fir Trees (Abies sibirica) Damaged by the Bark Beetle in Unmanned Aerial Vehicle Images with Deep Learning', *Remote Sensing*, 11(6), p. 643. doi: 10.3390/rs11060643.

Seeland, M. and Mäder, P. (2021) 'Multi-view classification with convolutional neural networks', *PLOS ONE*. Edited by R. Damaševičius, 16(1), p. e0245230. doi: 10.1371/journal.pone.0245230.

Tan, M. and Le, Q. (2019) 'EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks', in *International Conference on Machine Learning*. *International Conference on Machine Learning*, PMLR, pp. 6105–6114. Available at: http://proceedings.mlr.press/v97/tan19a.html

Turing, A. M. (1950) 'I.—COMPUTING MACHINERY AND INTELLIGENCE', *Mind*, LIX(236), pp. 433–460. doi: 10.1093/mind/LIX.236.433.

Vinatier, F. *et al.* (2018) 'The Use of Photogrammetry to Construct Time Series of Vegetation Permeability to Water and Seed Transport in Agricultural Waterways', *Remote Sensing*, 10(12), p. 2050. doi: 10.3390/rs10122050.

Wäldchen, J. and Mäder, P. (2018) 'Machine learning for image based species identification', *Methods in Ecology and Evolution*. Edited by N. Cooper, 9(11), pp. 2216–2225. doi: 10.1111/2041-210X.13075.

Sitographie :

Abdulla, W. (2018) *Splash of Color: Instance Segmentation with Mask R-CNN and TensorFlow*, *engineering.matterport.com*. Available at: https://engineering.matterport.com/splash-of-color-instance-segmentation-with-mask-r-cnn-and-tensorflow-7c761e238b46 (Accessed: 26 May 2021).

Consortium Pl@ntNet (2014) *Pl@ntNet Identify*, *Pl@ntNet Identify*. Available at: https://identify.plantnet.org/ (Accessed: 26 April 2021).

Annexe I : Comparaison des vérités terrain

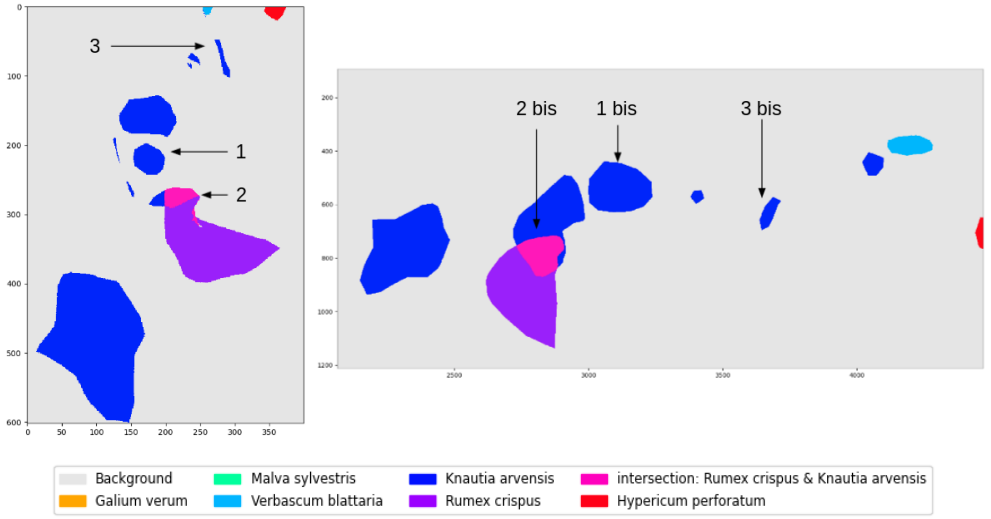


Figure 15 : Vérités terrain pour une portion de M2. (Les flèches indiquent des régions d'intérêt). Classe projetées sur une (Gauche) photographie, (Droite) ortho-image.

Quelques détails sur le calcul de la vérité terrain des images brutes : les fichiers utilisés pour connaître les coordonnées de chaque pixel dans le référentiel monde ne référencent qu'un pixel sur 10 dans chaque direction. Pour obtenir une image de la vérité terrain de même résolution que les photographies brutes, il faut donc procéder par interpolation. C'est une interpolation par plus proches voisins qui a été choisie, elle permet de conserver les classes déjà existantes sur l'image.

Entre les région 1 et 1 bis (Figure 15) on remarque une forme homogène sur l'ortho-image qui devient morcelée sur la photographie. Cela est dû au calcul des coordonnées par photogrammétrie : certaines zones de l'image sont trop complexes à analyser pour en déduire des coordonnées et sont donc affectées à des NaN, ce qui peut provoquer des "trous" dans la vérité terrain. La région 2 montre une intersection, qui apparaît très déformée sur la vérité terrain photographique (2 bis). C'est également une conséquence des calculs de coordonnées dans le référentiel monde des pixels de cette image.

Enfin la région 3, devient allongée lorsqu'on la compare à la région 3 bis. Cette transformation est totalement normale et vient du fait que les photographies sont prises avec un certain angle. Cet angle induit une perspective, et donc un allongement (ou affinement) des objets les plus distants de l'objectif de l'appareil photo. En ce qui concerne le reste de l'image, en prenant en compte la perspective, toutes les distributions sont relativement bien conservées lors de la conversion.

Ces 3 régions d'intérêt montrent les différences que l'on peut voir entre les deux images. Cette portion de M2 a été choisie pour montrer les différences que l'on peut voir, mais il est plutôt rare d'avoir des artefacts comme les régions 1 bis et 2 bis. La vérité terrain pour les photographies n'est pas exacte, mais elle s'approche au mieux de la réalité.

Annexe II : L'architecture d'EfficientNetB0

EfficientNet0 est composé de blocs, dans lesquels on trouve plusieurs couches de convolution. Ce système en blocs est très répandu en *deep learning* car il permet de construire des architectures de manière simple en empilant des blocs afin d'augmenter la profondeur d'un modèle. Lorsque l'on importe EfficientNetB0 avec la libraire *Tensorflow* de Python, c'est l'architecture de la figure 16 que l'on retrouve.

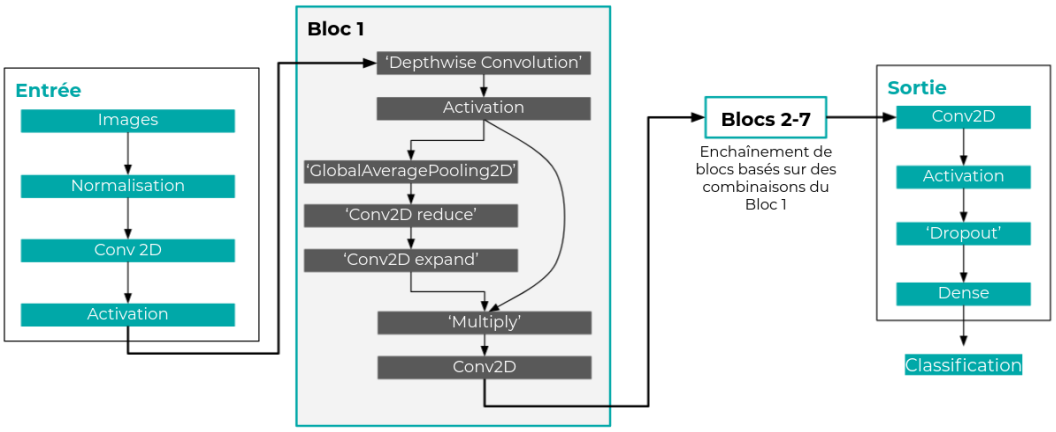


Figure 16 : Représentation simplifiée d'EfficientNet0, avec décomposition des couches des premiers blocs.

EfficientNetB0 est composé d'une couche classique d'entrée qui transforme les images de 224 par 224 pixels en tenseurs, puis réalise une première convolution (Conv 2D) afin d'extraire des features générales dans l'image. La couche d'activation est une pondération des sorties (utilisant la fonction y = x*sigmoid(x)). L'entrée mène ensuite à un premier bloc.

Le bloc 1 à une structure tout à fait particulière. En le suivant de haut en bas on remarque l'utilisation de couches de convolution classiques : 'Depthwise convolution' est un type de convolution particulier qui permet d'appliquer des convolutions à chaque canal en entrée (3eme dimension d'une image) et de conserver le même nombre de canaux en sortie. Les autres couches sont classiques jusqu'à la couche 'Multiply'.

Il s'agit d'une couche qui prend en entrée le résultat des convolutions du Bloc 1 mais aussi la sortie de la couche d'activation (au début du bloc 1). Cette particularité permet d'augmenter les performances du modèle. On appelle ce type de bloc un *squeeze and excitation block* (en français : bloc de compression et excitation). Car la couche résiduelle de l'activation est utilisée en fin de bloc pour activer la sortie des convolutions qui ont compressé l'image.

Le modèle utilise également d'autres astuces mais c'est principalement sur cette méthode de fonctionnement que sont basés les autres blocs.

En fin de modèle la sortie est aussi classique que l'entrée : une dernière convolution et activation des *features* finales avant de déterminer la classe de l'image via un réseau dense. Le 'dropout' est une couche qui élimine aléatoirement des neurones pour diminuer l'*overfitting*.

Annexe III : Calcul des scores

Utilisation des notations suivantes :

- Vrais positifs : TP
- Faux positifs : FP
- Vrais négatifs : TN
- Faux négatifs : FN
- Nombre d'échantillons : n

Précision équilibrée :

$$\frac{1}{2}\left(\frac{TP}{TP+FN}+\frac{TN}{TN+FP}\right)=\frac{1}{2}\left(Sensibilité+Spécificité\right)$$

Kappa de Cohen :

$$\frac{P(accord)-P(hasard)}{1-P(hasard)}$$

avec P(accord) la probabilité d'accord entre deux observateurs et P(hasard) la probabilité d'accord au hasard :

P(accord) = $\frac{TP+TN}{n}$ et P(hasard) = $\frac{(TP+FN)\times(TP+FP)+(TN+FP)\times(TN+FN)}{n}$

F1-score :

$$2\times\frac{(Spécificité\times Sensibilité)}{(Spécificité+Sensibilité)}=\frac{2TP}{2TP+FN+FP}$$

Score de jaccard :

$$\frac{TP}{n-TN}$$

Coefficient de corrélation de matthews :

$$\frac{(TN\times TP)-(FP\times FN)}{\sqrt{(TN+FN)(FP+TP)(TN+FP)(FN+TP)}}$$

Annexe IV : Résultats supplémentaires

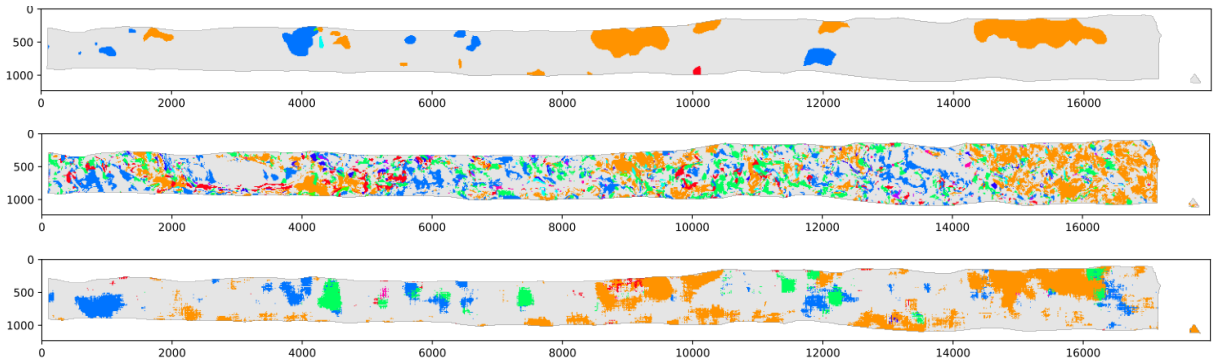


Figure 17 : Prédictions des répartitions des espèces végétales sur l'ortho-image de M4 (*excluded set*). (Haut) Vérité terrain. (Milieu) Prédiction *Ortho-ML*. (Bas) Prédiction *Ortho-DL*.

La figure 17 montre la projection des prédictions effectuées par *Ortho-ML* et *Ortho-DL* sur M4. Comme attendu au vu des scores plus faibles pour les jeux de données *excluded* (tableau 2), les prédictions sont moins bonnes que sur M2 (figures 11 et 12). Cependant on remarque deux choses : le modèle *Ortho-ML* produit toujours des distributions désorganisées et le modèle *Ortho-DL*, même si moins précis que sur M2, s'approche très bien de la vérité terrain. Cela prouve à nouveau la meilleure capacité du modèle *Ortho-DL* à généraliser sur des données inconnues.

Pour se rendre compte de l'avantage à prédire sur un jeu de données connu, le *test set*, il suffit de regarder les matrices de confusion pour ces données (Figure 18). Les erreurs sont rares, mais plus de confusions sont faites pour les prédictions avec ML qu'avec DL.

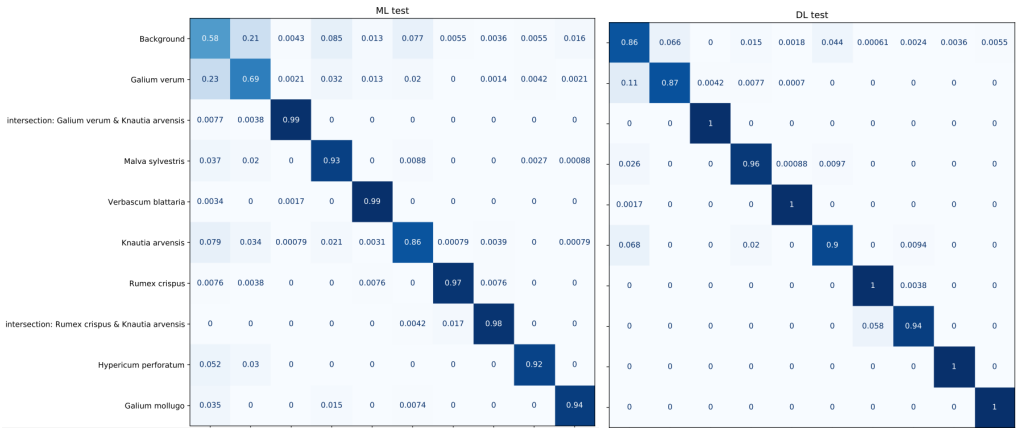


Figure 18 : Matrices de confusions pour deux modèles sur M4 (excluded set). (Gauche) *Ortho-ML*. (Droite) *Ortho-DL*.

l'institut Agro agriculture • alimentation • environnement / AGRO CAMPUS OUEST / UNIVERSITÉ DE RENNES 1	Diplôme : Ingénieur Agronome et Master 2 Spécialité : MODE (MODélisation en Ecologie) Spécialisation / option : --- Enseignant référent : Frédéric Hamelin
Auteur(s) : Loïc Lehnhoff Date de naissance* : 08/09/1998	Organisme d'accueil : IGEPP et LISAH Adresse : Domaine de la motte, Le Rheu (35) et Campus SupAgro, Montpellier (34)
Nb pages : 35 Annexe(s) : 4	
Année de soutenance : 2021	Maître de stage : Nicolas Parisey et Fabrice Vinatier

Titre français : Identification d'espèces végétales d'intérêt éco-hydrologiques dans les paysages agricoles : apports du Deep Learning et de l'Intelligence Artificielle.
Titre anglais : Identification of plant species with eco-hydrological importance in agricultural landscapes : contribution of Deep Learning and Artificial Intelligence.

Résumé (1600 caractères maximum) :

L'identification automatisée d'espèces végétales à partir de photographies est un enjeu majeur pour la détermination de propriétés éco-hydrologiques des milieux. Les modèles d'intelligence artificielle sont déjà capables d'identifier des plantes sur des photos avec une composition simple. Pour appréhender les capacités de prédiction de ces modèles sur des images plus complexes, l'hypothèse de l'apport de données multi-vues dans l'objectif de faciliter ces prédictions, est étudié. A partir de données photographiques et de relevés botaniques issus de l'étude d'un fossé agricole méditerranéen, quatre modèles de segmentation sémantique sont construits et comparés. Leur but étant de produire des prédictions spatialisées des espèces végétales d'une image. Deux modèles sont basés sur l'utilisation d'ortho-images : *Ortho-ML* utilisant le machine learning et *Ortho-DL* utilisant le deep learning (DL). Et deux autres en deep learning, utilisant des photographies au sol : *mono-entrée-DL* et *multi-entrées-DL*. Les modèles de DL sont basés sur EfficientNetB0, une architecture faisant état de l'art en DL. Les résultats de la comparaison entre *Ortho-ML* et *Ortho-DL* montrent plusieurs avantages à l'utilisation du DL, c'est pourquoi les modèles multivues sont basés sur la même architecture. Les prédictions des modèles multivues sont moins performants que *Ortho-DL*, mais peuvent être compétitifs dans le détail en fonction des espèces. Ces résultats ne valident pas notre hypothèse de départ mais sont encourageants et ouvrent la voie à de futurs travaux.

Abstract (1600 caractères maximum) :

The automatic identification of plant species using photographs is important for the determination of an environments' eco-hydrological properties. Artificial intelligence models are already capable of identifying plants on photographs with simple complexity. To apprehend the capabilities of such models on more complex images, we study the hypothesis of an advantage in using multiview data to facilitate predictions. Using photographs and botanic reports collected from a study of a mediterranean agricultural pit, we built and compared four models that use semantic segmentation, in the goal of producing spatialized prediction of plant species from images. Two models use ortho-images : *Ortho-ML* in machine learning and *Ortho-DL* in deep learning (DL). And two others in DL, using photographs taken from the ground : *mono-entrée-DL* et *multi-entrées-DL*. DL models are based on EfficientNetB0, a state-of-the-art architecture in DL. A Comparison of the results between *Ortho-ML* and *Ortho-DL* shows several advantages in using DL. That is why we based multiview models on the same architecture. Multiview models make less efficient predictions than *Ortho-DL* but can be competitive when used for specific species. Those results do not validate our base hypothesis but are encouraging and open the path for future works.

Mots-clés : Intelligence Artificielle, Deep Learning, Multi-vues, Segmentation, Plantes.

Key Words: Artificial Intelligence, Deep Learning, Multiview, Segmentation, Plant.

* *Elément qui permet d'enregistrer les notices auteurs dans le catalogue des bibliothèques universitaires*

PS : Idées supplémentaires

Possible ajouts concernant les pré-traitements des images:
-> Explications de l'égalisation plus en détails
-> Explication du fonctionnement de Unit Outputs (très facultatif vu la complexité)

Expliquer la division des scripts en 4:
pre-process
create-dataset
modelization
prediction-analysis

Description de l'oridnateur utilisé

Matériel et Méthode, vérité terrain.

Une seconde stratégie a été implémentée concernant la gestion de ces intersections. Un classement des espèces selon l'importance de leur influence sur l'écoulement de l'eau a été réalisé. Les espèces sont classées en 3 catégories : L, M et H (pour des niveaux d'importance faibles, moyens et élevés respectivement). Ainsi, dans le cas d'une superposition, seule l'espèce avec le niveau d'importance le plus élevé est conservée. Il est nécessaire d'utiliser cette méthode lorsque beaucoup d'espèces sont présentes sur une image. En effet, avec le nombre d'espèces, le nombre d'intersections de leurs distributions augmente et peut rapidement devenir plus important que le nombre d'espèces de base. Cette stratégie permet de limiter le nombre de classes à prédire en introduisant une sélection raisonnée des données. Dans les faits, les modèles ont été évalués sur moins d'une dizaine d'espèces parmi les 70 cartographiées afin d'accélérer les temps d'entraînement des modèles. Cette seconde stratégie n'a donc pas été exploitée mais elle pourra être appliquée à d'autres jeux de données.