

FloraFlow

Cultivez mieux avec l'IA

Projet Fil Rouge

Data Scientist - Promotion Août 2023



Equipe du projet

Gilles de Peretti

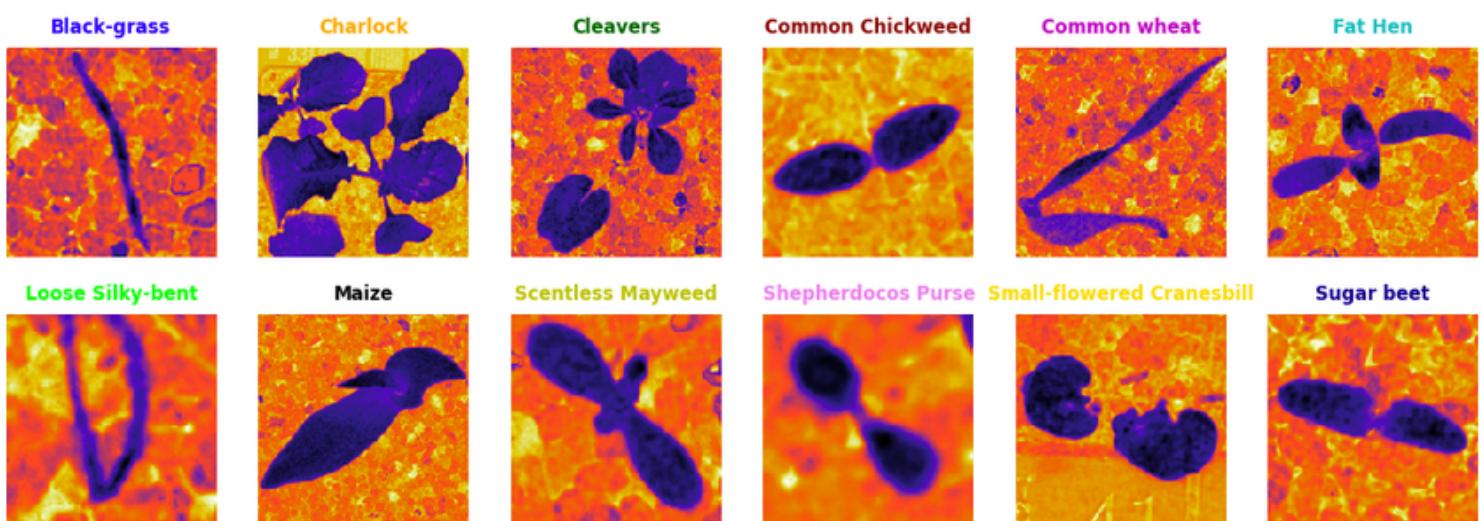
Hassan ZBIB

Dr. Iréné Bérenger AMIEHE ESSOMBA

Olivier MOTELET

Table des Matières

4	Introduction	23	Modélisation - Réglages
5	Analyse exploratoire des données (EDA)	35	Modélisation - Résultats finaux
18	Modélisation - Préparation des données	38	Modélisation - Conclusion & pistes d'amélioration
19	Modélisation - Choix des modèles	40	Annexes



Equipe derrière ce projet



Gilles de PERETTI

ACTUELLEMENT EN RECONVERSION PROFESSIONNELLE. JE POURSUIS LE CURSUS MACHINE LEARNING ENGINEER. LA COMPUTER VISION M'ATTIRE CAR J'AI UNE IDÉE QUE JE SOUHAITE TRANSFORMER EN SOLUTION FONCTIONNELLE. C'EST POURQUOI J'AI CHOISIS CE PROJET FIL ROUGE.



Hassan ZBIB

ATTRIÉ PAR LE MONDE DE L'INTELLIGENCE ARTIFICIELLE J'AI DÉCIDÉ DE FRANCHIR LE PAS EN ME FORMANT CHEZ DATASCIENTEST. ET C'EST PAR INTÉRÊT PERSONNEL POUR LES PLANTES QUI M'A POUSSÉ A CHOISIR CE PROJET.



Dr. Iréné AMIEHE ESSOMBA

DOCTEUR EN COMPUTER SCIENCE, JE ME CONSACRE À RÉSOUDRE DES PROBLÈMES COMPLEXES EN UTILISANT DES TECHNIQUES AVANCÉES DE TRAITEMENTS DES DONNÉES. CE PROJET FIL ROUGE M'AIDERA À ATTEINDRE MES OBJECTIFS .



Olivier MOTELET

INGÉNIER LOGICIEL ET UX RESEARCHER, CE PROJET ET CETTE FORMATION ME PERMET D'AFFINER MES CONNAISSANCES EN DATA SCIENCE.



Maxime MICHEL

NOTRE MENTOR POUR CE PROJET, IL NOUS ACCOMPAGNE DANS LE DÉVELOPPEMENT ET L'ABOUTISSEMENT D'UN PROJET FONCTIONNEL.

Introduction

Les plantes représentent un défi complexe en termes de classification et de reconnaissance des espèces, notamment en ce qui concerne les mauvaises herbes nuisibles, un enjeu pour lequel l'apprentissage automatique et la vision par ordinateur offrent des solutions prometteuses pour les agriculteurs de demain.

Problématique

La réussite de la culture du maïs, par exemple, dépend en grande partie de l'efficacité de la lutte contre les mauvaises herbes, en particulier au cours des six à huit premières semaines après la plantation. Les mauvaises herbes peuvent entraîner d'importantes pertes de rendement, allant de 10 à 100 %, en fonction de divers facteurs tels que le type de mauvaises herbes et les conditions environnementales. Lutter efficacement contre ces mauvaises herbes nécessite une identification précise, un processus souvent long et sujet à des erreurs.

C'est d'autant plus véridique dans le contexte actuel, où une partie du monde agricole est en train de se réinventer afin de devenir plus durable, et moins consommateur d'agents chimiques. Le développement de la robotique peut permettre l'arrachage mécanique automatique des plantes envahissantes, à condition de développer des modèles d'intelligences artificielles à même de les reconnaître avec certitude.

Objectif

Ce projet a pour objectif de développer un modèle de deep learning capable de classer avec précision différentes espèces de plantes, en vue d'améliorer la gestion des cultures. Nous utiliserons le jeu de données Kaggle V2 Plant Seedlings Dataset, qui comprend une collection d'images de semis de plantes appartenant à diverses espèces.

Les principales étapes de notre projet comprendront

- l'exploration approfondie du jeu de données,
- la préparation des données,
- la conception et la mise en œuvre d'un modèle d'apprentissage,
- ainsi que l'analyse de ses performances.

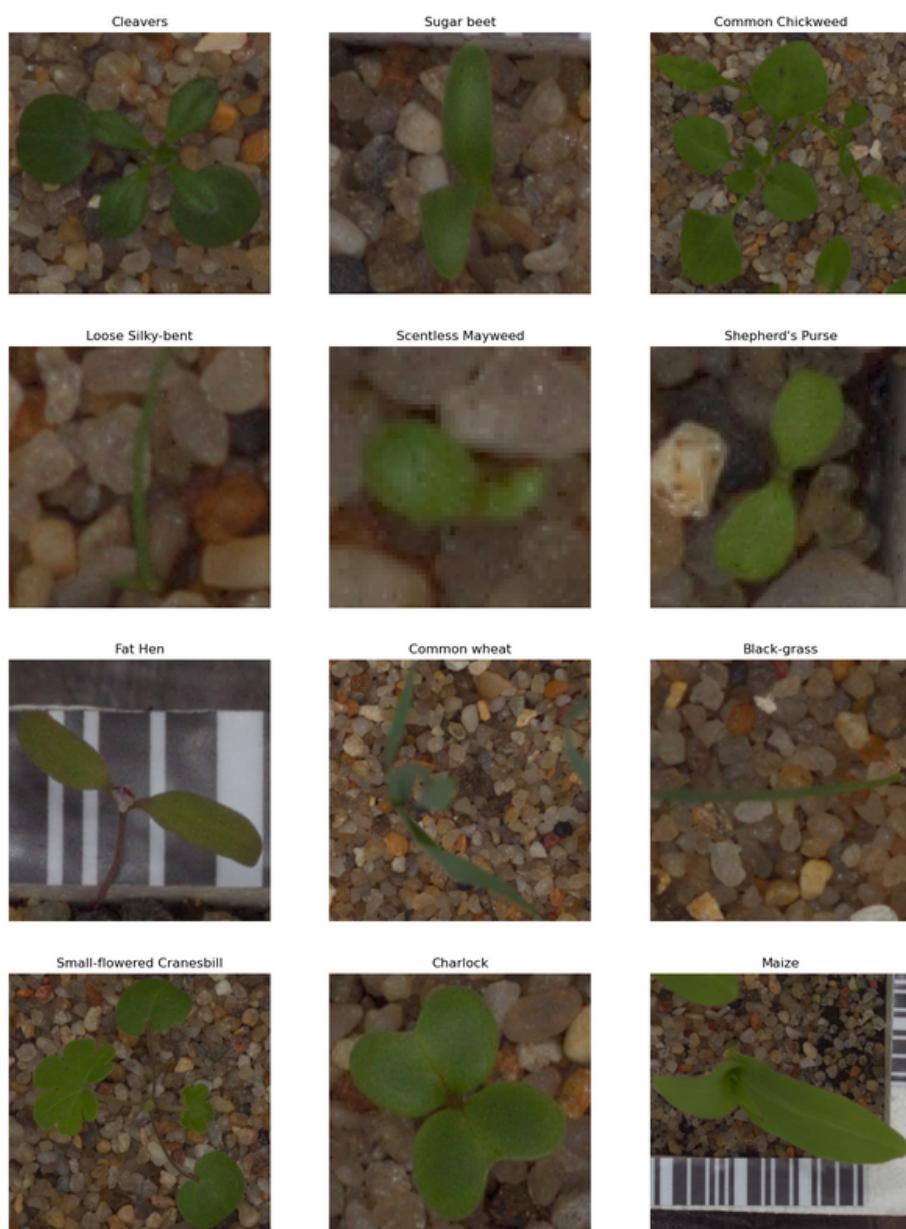
Notre objectif final est de fournir les bases d'un outil robuste pour la classification des plantes, offrant ainsi aux agriculteurs des moyens plus efficaces de gérer leurs cultures.

Exploration du jeu de données

Le jeu de données utilisé est intitulé [V2 Plant Seedlings Dataset disponible sur Kaggle](#) est une base de données d'images publiques pour l'étalonnage des algorithmes de classification des semis de plantes.

Le dataset V2 Plant Seedlings est composé de 5539 images représentant des plants au stade de la germination. Elles sont regroupées en 12 classes, représentant chacune une variété / espèce de plantes.

Exemple d'images pour chaque espèce



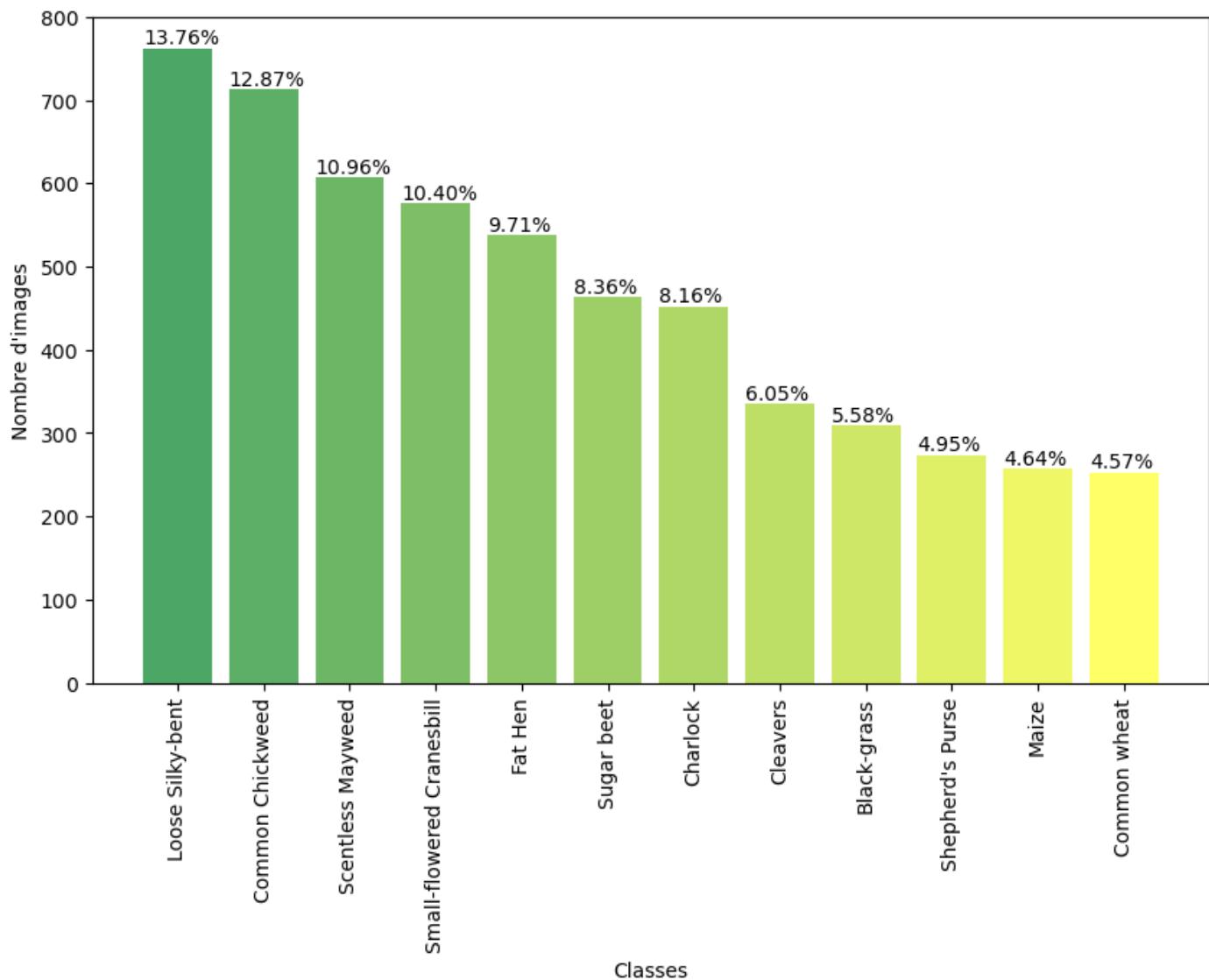
L'exploration de ce jeu de données comprendra les étapes suivantes :

- Analyse des caractéristiques du jeu de données.
- Exploration du contenu des images.

Analyse des caractéristiques du jeu de données

Nombre d'images dans le dataset

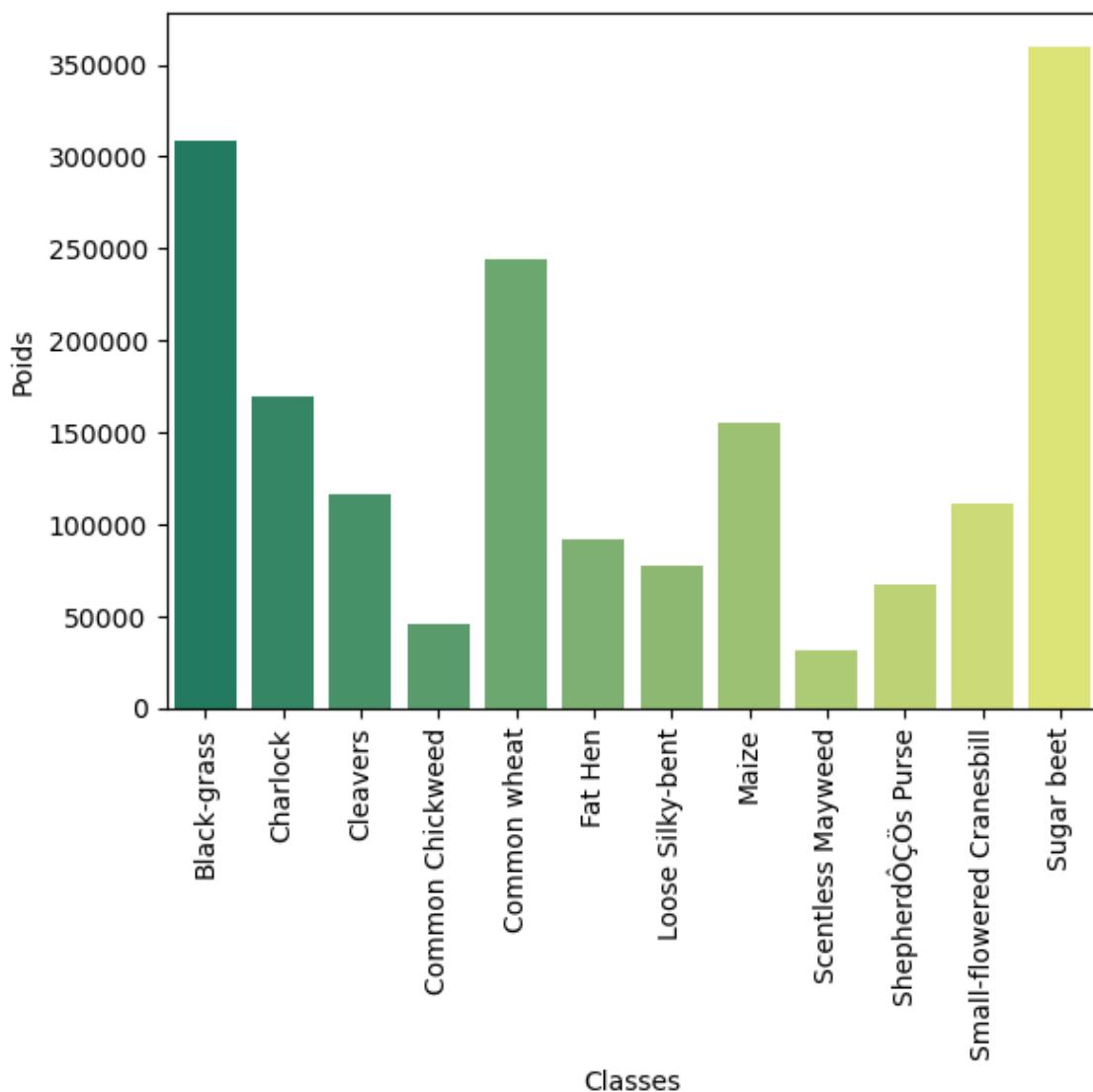
Distribution des classes (espèces) dans le dataset



On remarque une disparité entre le nombre de photos qui composent chaque classe, cette différence va presque jusqu'au triple dans certains cas. Il faudra donc que nous prenions en compte le déséquilibre des classes dans nos modélisations.

Taille des images du dataset

Poids médian des images en fonction de l'espèce



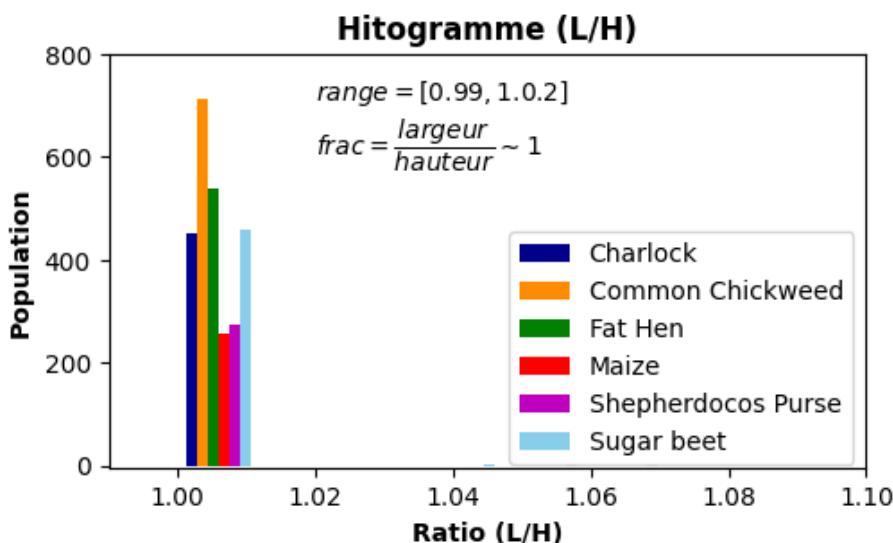
Le poids moyen des photos qui nous informe sur leur taille, dénote une différence de qualité de photos suivant les classes. Dans le cas de la classe « Black-grass » il y a peu de photos, mais celles-ci sont de grande qualité, tandis que pour « Scentless Mayweed » c'est l'inverse beaucoup de photos mais de qualité médiocre.

Les dimensions des images sont très variées, allant de 49x49 à 3457x3652 pixels. Une étape de normalisation sera nécessaire. Très peu d'images au-delà de 1500x1500 pixels. La moyenne est d'environ 355x355 pixels et la valeur médiane est de 267x267 pixels.

Un travail d'homogénéisation de la taille des images devra être fait en prenant en compte les disparités entre les classes.

Rapport Hauteur/Largeur des images du dataset

Rapport hauteur\largeur par espèces

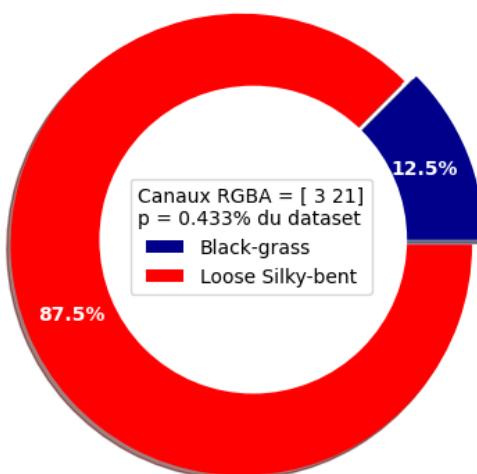


Bien que 98,77% des images aient un ratio hauteur-largeur égal à 1 (c'est-à-dire qu'elles sont carrées), certaines présentent un ratio qui s'approche de 1 sans toutefois y être égal. Nous re-dimensionnerons les images pour avoir largeur = hauteur.

Espace de couleur

La grande majorité des images est en format RGB. Seules quelques unes (0,433%) sont en format RGBA. Nous mettrons toutes les images au format RGB

Répartition par espèces des images RGBA en pourcentage du total



Histogrammes de couleur

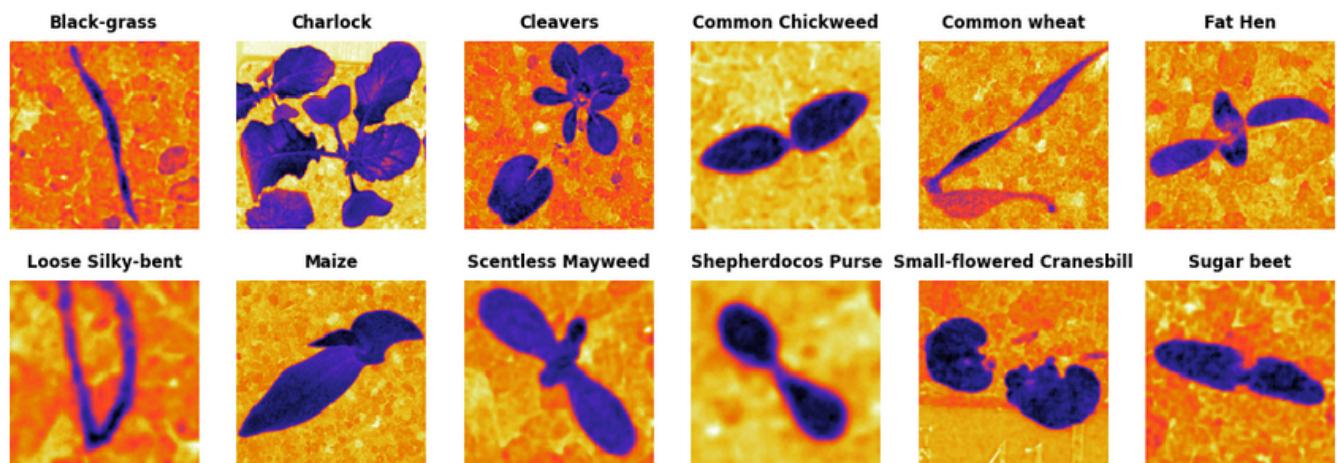
Les histogrammes de couleurs sont utilisés pour réaliser la segmentation sémantique d'une image, c'est-à-dire pour créer des filtres spécifiques permettant de différencier les objets présents dans une image. Dans le cadre de ce projet, cette méthode est employée pour séparer un objet (une plante) de son arrière-plan (un fond bruité).

Nous utilisons l'espace colorimétrique LAB qui tente de représenter de manière plus précise la perception humaine des couleurs par rapport au RGB.

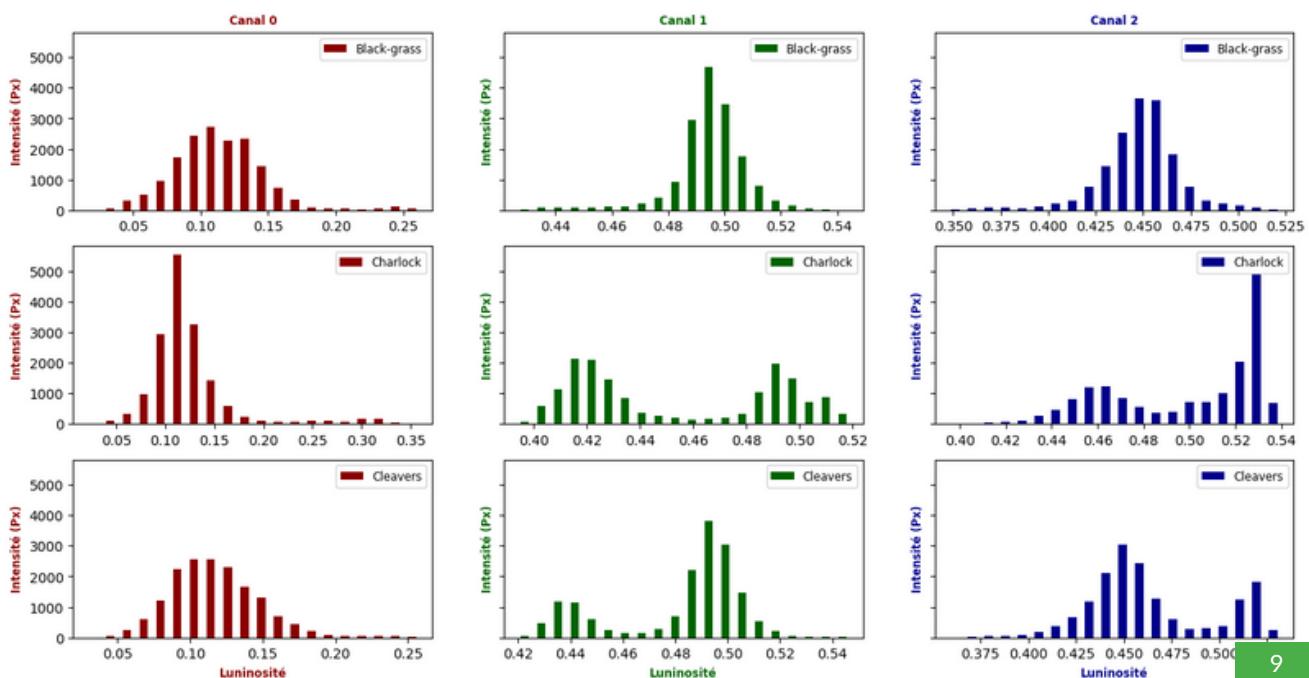
Il se compose de trois composantes principales:

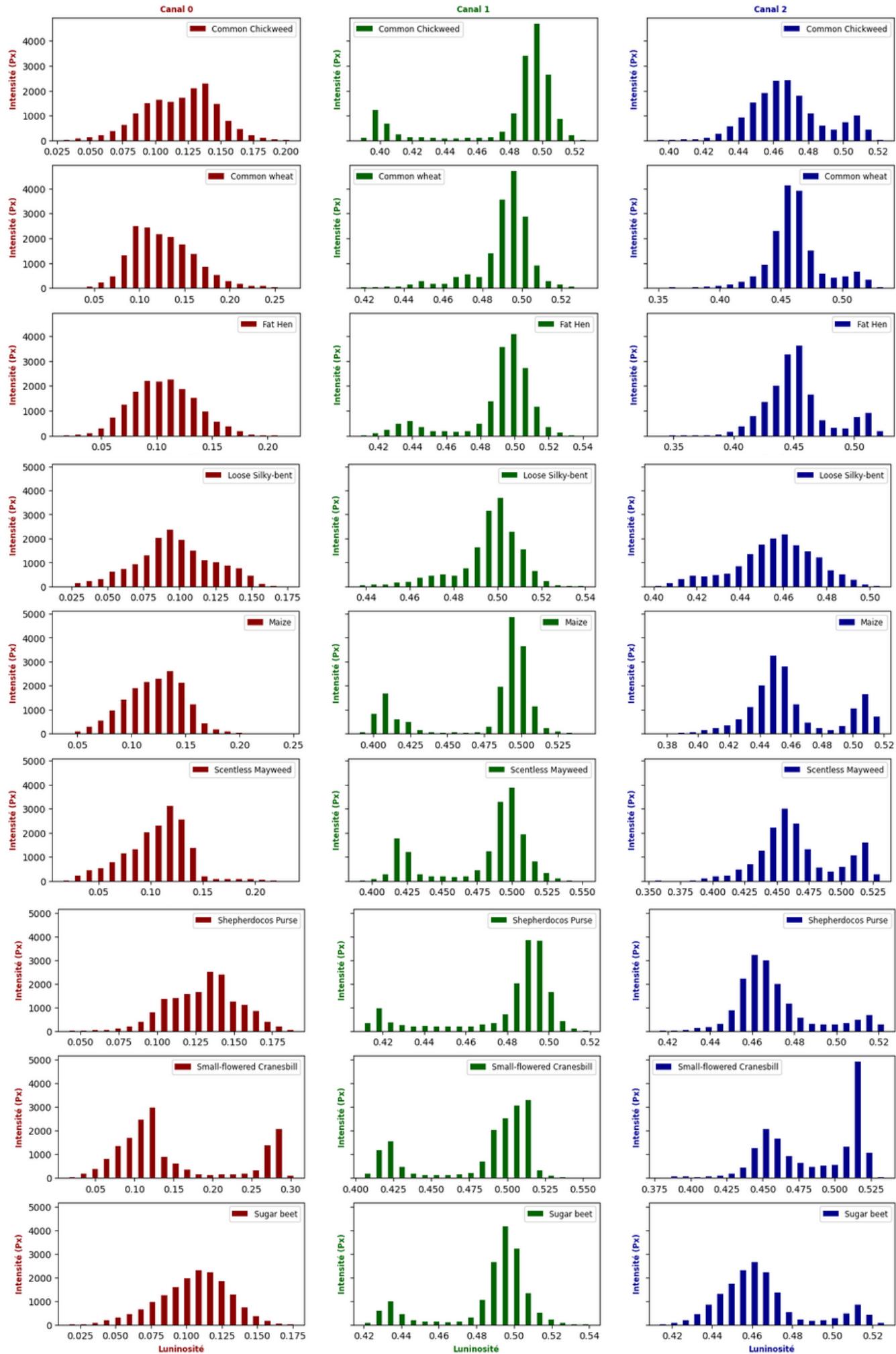
- L (Luminance) : représente la luminosité ou la clarté de la couleur.
- A : représente la gamme de couleurs de vert à magenta.
- B : représente la gamme de couleurs de bleu à jaune.

Exemple d'images dans l'espace colorimétrique LAB



Histogramme des couleurs par canal et par espèce.





Les résultats obtenus permettent de visualiser la distribution des couleurs de l'image dans ses trois canaux, avec des variations d'intensité de pixellisation d'un canal à l'autre. Toutefois, il est essentiel de noter que seul le canal 1 permet une distinction optimale de la couleur verte des plantes, avec un intervalle de valeurs compris entre [0.39, 0.46]. Dans les deux autres cas, soit le canal 0 ne présente pas de plage distincte, soit le canal 2 ne montre pas de pics significatifs. Nous utiliserons le canal 1 pour différencier la couleur verte de la plante du reste de l'image.

Des informations plus détaillées seront fournies dans la suite du rapport, notamment :

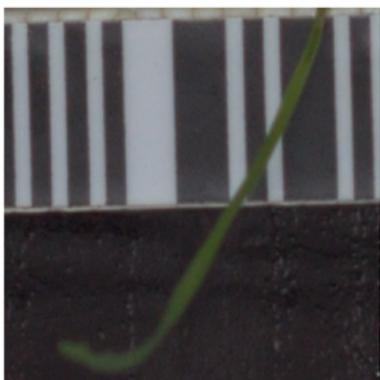
- Comment sélectionner le canal optimal en se basant sur l'histogramme de couleurs pour créer une segmentation efficace de l'image.
- Comment utiliser ce canal pour créer un filtre extrêmement puissant.
- Comment obtenir une image finale nette en appliquant ce filtre à l'image originale.

Exploration du contenu des images

Différence d'apparence en fonction de l'espèce

Voici une illustration de chaque espèce à partir de trois images prises aléatoirement :

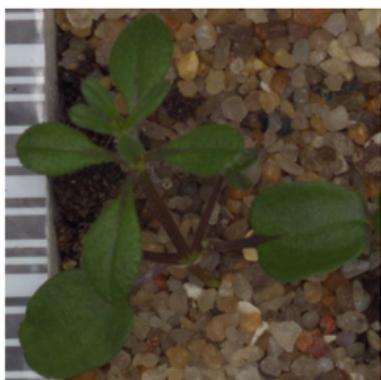
Black-grass



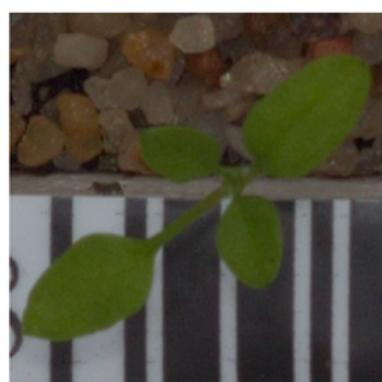
Charlock



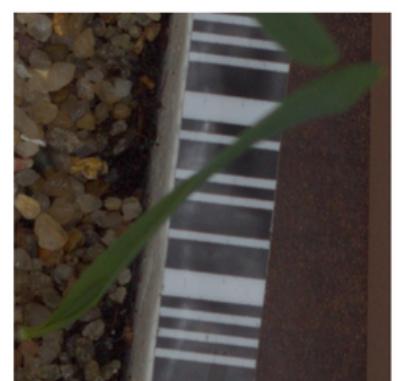
Cleavers



Common Chickweed



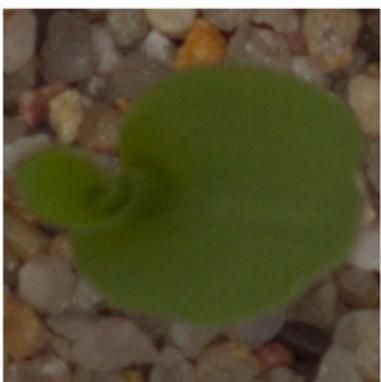
Common wheat



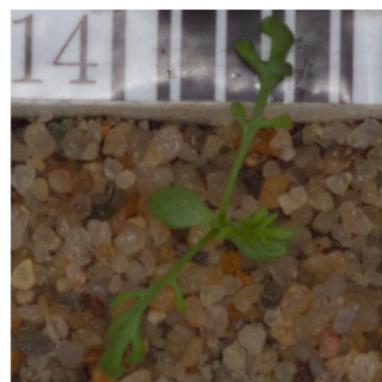
Fat Hen



Maize



Scentless Mayweed



Sheperd's Purse



Small-flowered Cranesbill



Loose Silky-bent



Sugar Beet



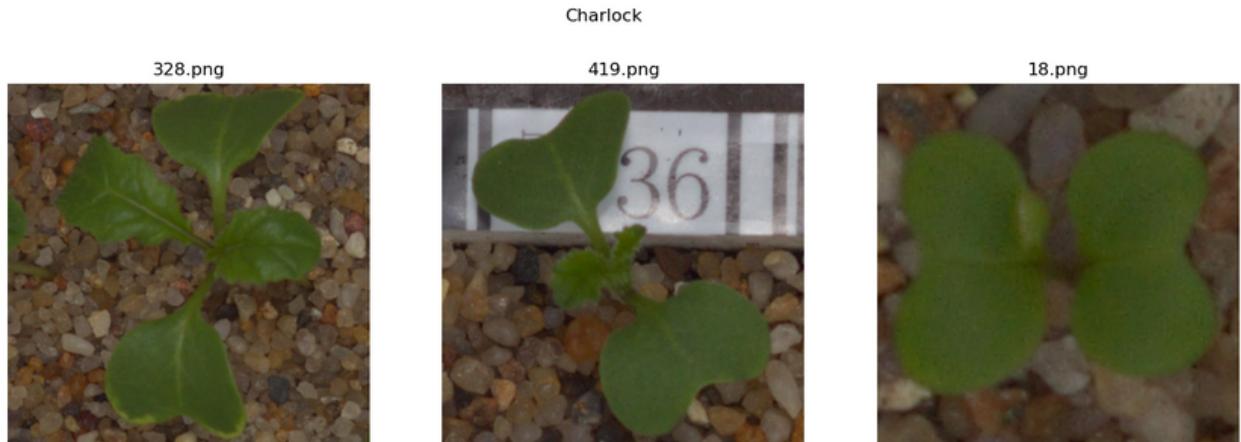
On constate en visualisant aléatoirement notre dataset, bien que les plantes soient toujours centrées au sein de l'image, la qualité de l'image n'est pas toujours au rendez-vous ce qui nous renvoie au poids des images selon les classes.

De plus, il ressort de cette visualisation que certaines espèces sont très différentes, tandis que d'autres se ressemblent beaucoup, comme par exemple : Loose Silky-bent et Black-grass. Cela soulève la question de savoir si un regroupement en "familles" pourrait être utile au futur modèle.

De même, nous voyons que les plants ont été photographiés à différents stades de croissance. Pour une même espèce, une plante peut ne pas avoir le même nombre de feuilles et la forme des feuilles peut aussi varier.

Arrières plans

Nous avons constaté sur les échantillons que le fond des images n'est pas toujours "uniforme". Il est soit constitué de graviers soit d'une toile noire soit des deux et des instruments peuvent s'ajouter.



D'autres part, du stade de croissance le plus avancé à gauche au stade le plus précoce à droite sur la figure précédente, on constate que la taille des graviers ainsi que le flou augmentent dans le même sens de visualisation. Si nous analysons les propriété extraites des images et regroupées dans un dataframe, nous pouvons constater que :

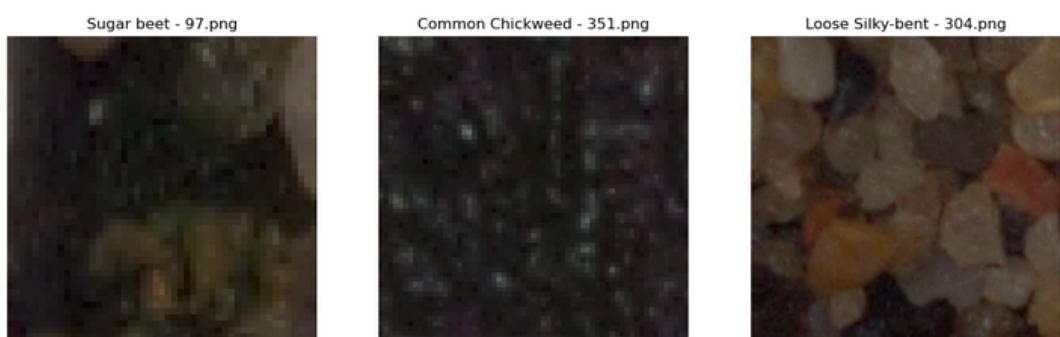
	Classe	Nom du fichier	Chemin	Hauteur	Largeur	Forme	Résolution	Ratio
4963	Charlock	328.png	/Users/morph/Desktop/DS/projet/dataset_V2_plan...	514	514	(514, 514, 3)	264196	1.0
5106	Charlock	18.png	/Users/morph/Desktop/DS/projet/dataset_V2_plan...	159	159	(159, 159, 3)	25281	1.0
5204	Charlock	419.png	/Users/morph/Desktop/DS/projet/dataset_V2_plan...	358	358	(358, 358, 3)	128164	1.0

Les dimensions de nos images augmentent selon le stade de croissance observé visuellement. Une explication plausible sur la disparité dans la taille des images et le flou peut être que les images aient été prises toujours avec le même appareil mais recadrées pour centrer le plant au sein de l'image.

Une autre piste de segmentation des stades de croissance se faire en fonction du nom des images et de leur hiérarchie au sein d'une classe. Cependant, l'analyse du lien entre taille et nom d'image montre qu'il n'existe pas une telle corrélation. Il sera peut être intéressant de conserver la taille originale de l'image comme feature caractéristique de nos données et représentant le taux de croissance.

Anomalies

Sur l'ensemble du jeu de données, nous avons détecté trois images qui ne contiennent pas de plantes mais seulement le fond que l'on peut détecter sur les autres images :



Lors du téléchargement du dataset, on observe un dossier intitulé "nonsegmentedv2" qui contient des sous-dossiers portant le nom des 12 espèces. Bien que les images semblent identiques, certaines comportent plusieurs espèces dans une seule image. Cela pourrait poser un problème pour notre modèle qui travaille sur la base de probabilités. Il est donc préférable de ne pas utiliser ces images, mais plutôt de se fier aux 12 premiers dossiers qui contiennent uniquement les images adéquates.

I have also learned not to take glory in the difficulty of a proof: difficulty means we have not understood.
The ideal is to be able to paint a landscape in which the proof is obvious.

— Pierre Deligne
Notices of the American Mathematical Society, Volume 63, Number 3, pp. 250, March 2016

Modélisation

Sur les dix dernières années, les CNN (Convolutional Neural Networks) se sont positionnés comme les modèles de référence pour les tâches de reconnaissance d'image. Leur structure hiérarchique leur permet d'apprendre des schémas complexes. Et même si depuis quelques mois, les Vision Transformer (ViT) semblaient dépasser les CNN, les tous derniers modèles de CNN (ConvNeXt v2) atteignent des performances similaires avec un coût de traitement bien plus raisonnable que pour les ViT.

Étant donné nos capacités de traitement des données (machines personnelles), nous nous sommes concentrés sur des modèles de type CNN. Cette phase de modélisation comprend une série d'étapes et de choix :

- préparation des données
- choix du CNN
- réglage fin
- résultats finaux et pistes d'amélioration

Ces choix se baseront non seulement sur les résultats de nos expérimentations, mais aussi sur :

- notre contexte de développement – nos machines sont relativement peu puissantes et ne peuvent entraîner des modèles très gourmands
- le contexte d'usage – il est raisonnable d'imaginer que les robots de désherbage automatiques pouvant appliquer ce genre de modèle seront des machines avec des ressources limitées

Préparation des données

Data augmentation

Notre phase d'exploration nous a montré que nous avons un jeu de données réduit et déséquilibré. Nous traiterons la partie équilibrage dans un second temps. Dans un premier temps, nous nous concentrerons sur l'augmentation des données. Pour ce faire, nous appliquerons les filtres suivants :

- rotations aléatoires : `rotation_range=360`,
- zooms aléatoires : `zoom_range=0.2`,
- translations aléatoires : `width_shift_range=0.2, height_shift_range=0.2`,
- remplissage des parties manquantes : `fill_mode="nearest"`

Ces choix ont été faits pour générer le plus de diversité possible.

Nous avons aussi essayé d'utiliser un cisaillement (shear) mais nos tests ont montré que cette déformation avait un impact négatif sur nos résultats. Nous présumons que c'est parce que la forme des feuilles est importante dans la reconnaissance d'une plante.

Création du jeu de test

Dans un premier temps, nous avons utilisé la fonction `image_dataset_from_directory` de Keras, simple et amenant de bonnes performances avec une optimisation du prefetch.

Mais nous avons finalement utilisé un `ImageDataGenerator` avec la fonction `flow_from_dataframe` afin de garder plus de contrôle sur nos données de tests et pouvoir retester nos modèles plus facilement en association avec une random seed constante. D'autant plus que certaines méthodes, utiles à l'implémentation du Grad-CAM, ne sont pas accessible avec la première fonction.

Afin de garder un équilibre entre la quantité de données d'entraînement et un jeu de test statistiquement raisonnable, nous avons choisi le découpage suivant :

- 10% pour le jeu de test (soit 553 images)
- 90% pour le jeu d'entraînement incluant une part de validation de 11%

Redimensionnement des images

Nous avons choisi 2 tailles d'entrée dans nos modèles pendant la phase d'expérimentation toutes 2 multiples de 32 :

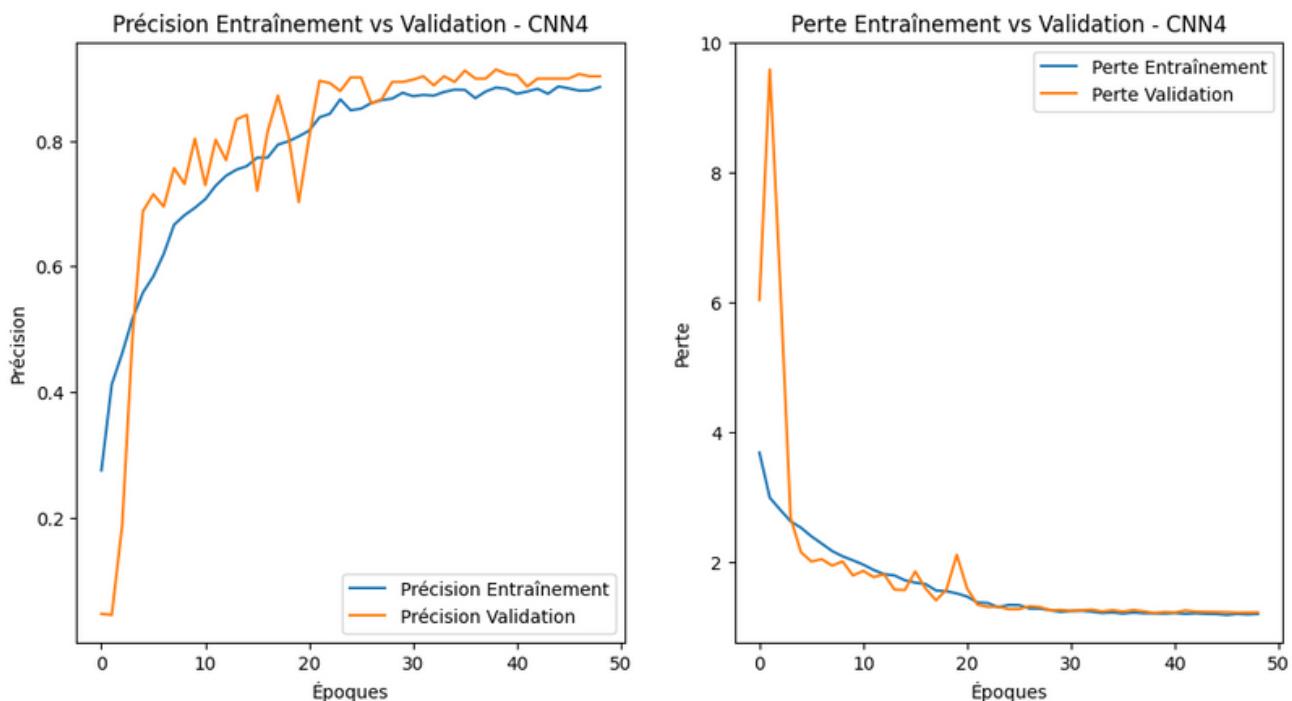
- 128x128 : pour la légèreté que nous permet ce format réduit
- 224x224 : car cela se rapproche de la taille médiane (267x267) de notre dataset

Choix des modèles

Au cours de nos expérimentations individuelles, nous avons essayé différents modèles, des constructions propres aux modèles pré-entraînés. Bien qu'un de nos CNN maison ait obtenu une précision de 91%, les modèles pré-entraînés dépassent largement le CNN maison.

Le CNN maison est composé de couches convolutives qui augmentent en profondeur de filtres de 32 à 256 et diminuent en taille de noyau de (6, 6) à (3, 3), toutes utilisant ReLU pour l'activation et une régularisation L2 pour limiter le surajustement. Chaque couche convulsive est suivie d'une normalisation par lots et d'une réduction de dimensionnalité via MaxPooling. Les caractéristiques apprises sont aplatis et passent par une couche dense de 512 neurones avec ReLU et régularisation L2, puis par une étape de dropout à 50% avant d'atteindre la couche de sortie avec 12 neurones et une activation softmax pour la classification en plusieurs classes.

Précision vs Perte sur les ensembles d'entraînement et de validation du CNN maison

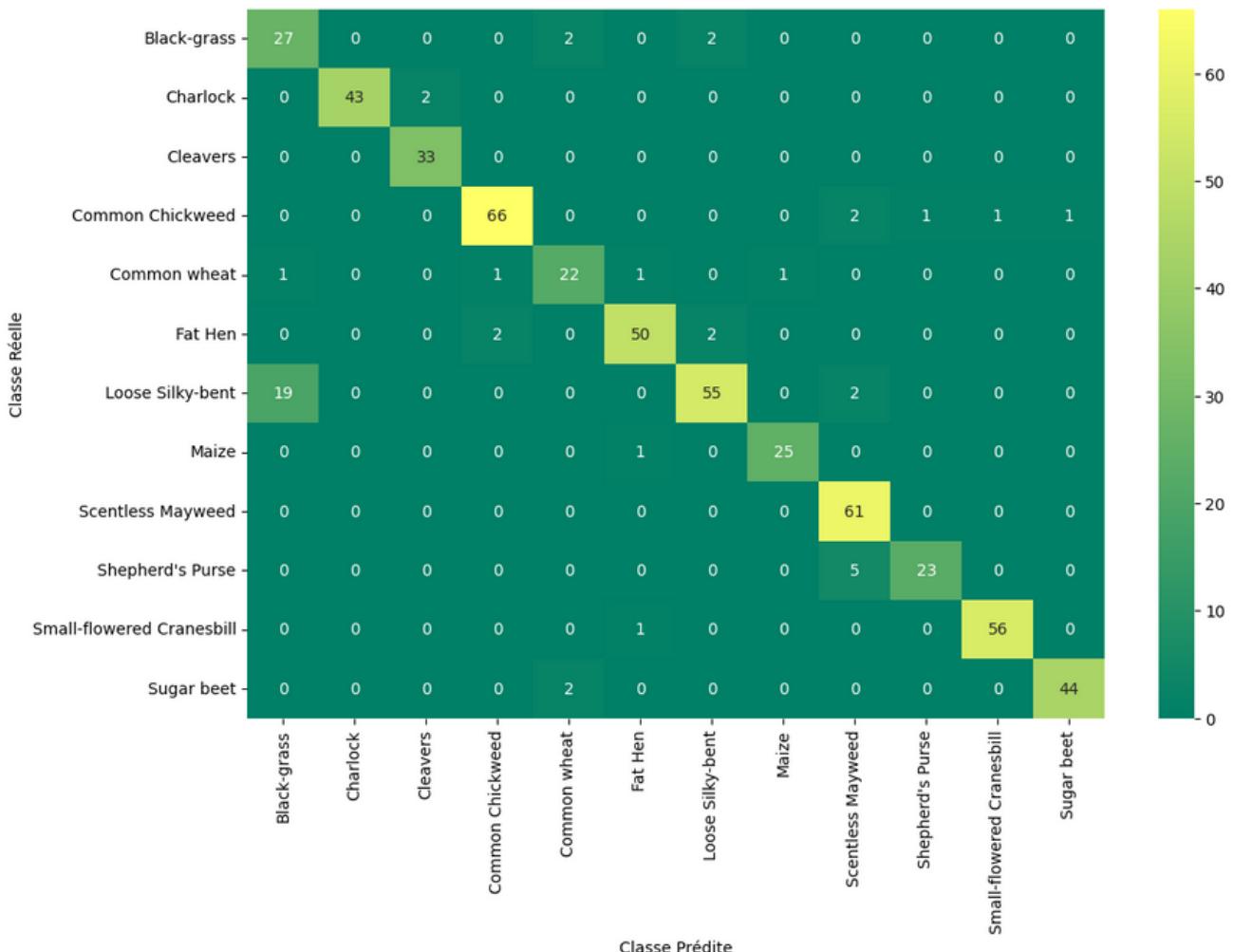


Le CNN affiche une convergence satisfaisante, illustrée par une amélioration progressive de la précision et une diminution de la perte au fil des époques. Ce résultat a été obtenu grâce à un entraînement de 50 époques, ce qui est plus important que les modèles pré-entraînés. Cela est normal car le CNN doit apprendre à partir de rien et sur un jeu de données de taille réduite.

Rapport de classification du CNN Maison

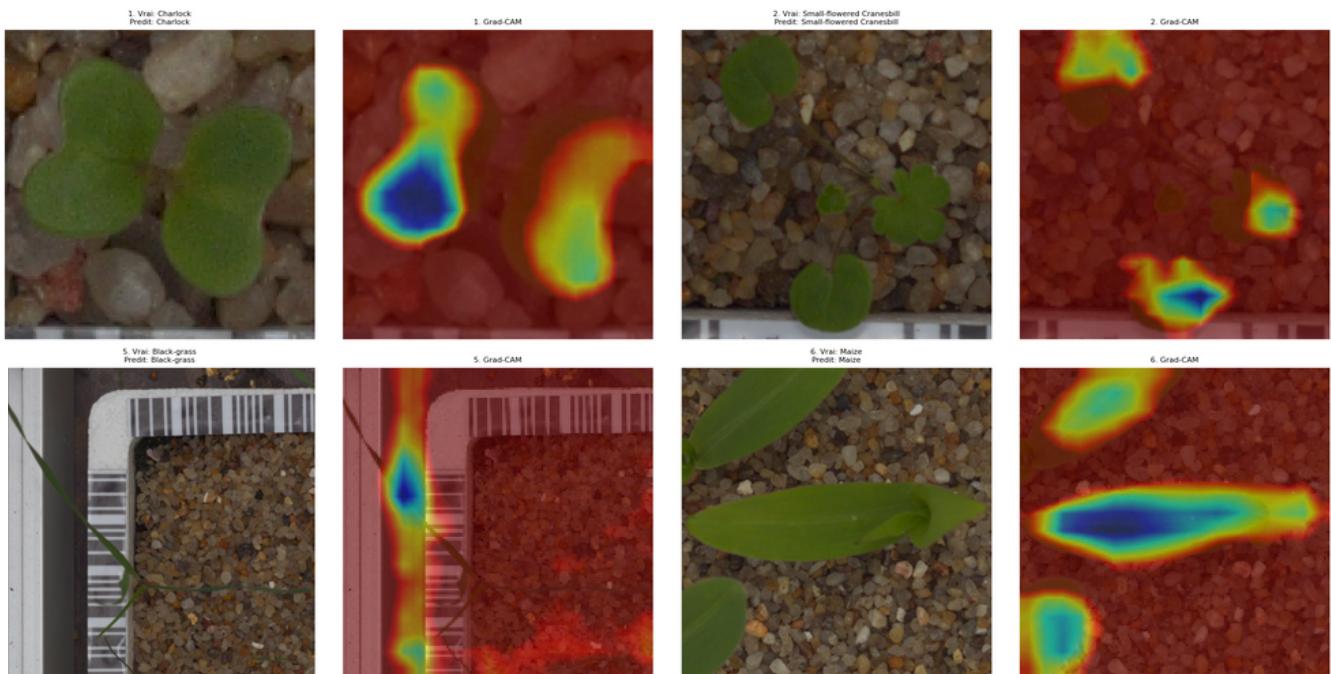
	precision	recall	f1-score	support
Black-grass	0.57	0.87	0.69	31
Charlock	1.00	0.96	0.98	45
Cleavers	0.94	1.00	0.97	33
Common Chickweed	0.96	0.93	0.94	71
Common wheat	0.85	0.85	0.85	26
Fat Hen	0.94	0.93	0.93	54
Loose Silky-bent	0.93	0.72	0.81	76
Maize	0.96	0.96	0.96	26
Scentless Mayweed	0.87	1.00	0.93	61
Shepherd's Purse	0.96	0.82	0.88	28
Small-flowered Cranesbill	0.98	0.98	0.98	57
Sugar beet	0.98	0.96	0.97	46
accuracy			0.91	554
macro avg	0.91	0.91	0.91	554
weighted avg	0.92	0.91	0.91	554

Matrice de confusion du CNN Maison

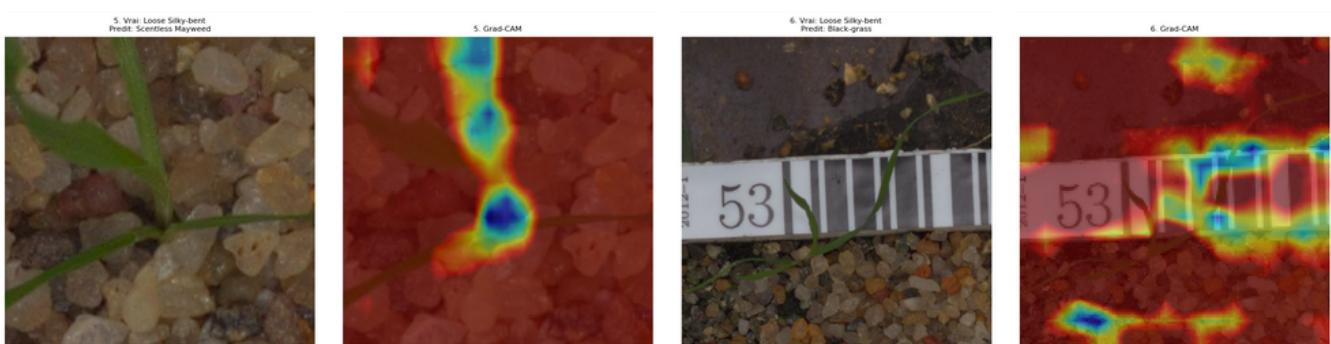


Ce CNN démontre une forte performance globale avec une précision moyenne de 0.91, indiquant une capacité efficace à classifier correctement les différentes espèces de plantes. Malgré une confusion sur les espèces comme Black-grass et le Loose Silky-bent, la majorité des espèces sont bien distinguées, comme en témoignent les scores élevés de précision et de rappel. Le modèle présente un bon équilibre entre la précision et la capacité à identifier la majorité des vrais positifs, ce qui est validé par de solides scores F1.

Interprétabilité du CNN Maison - Grad-CAM sur les bonnes prédictions



Interprétabilité du CNN Maison - Grad-CAM sur les mauvaises prédictions

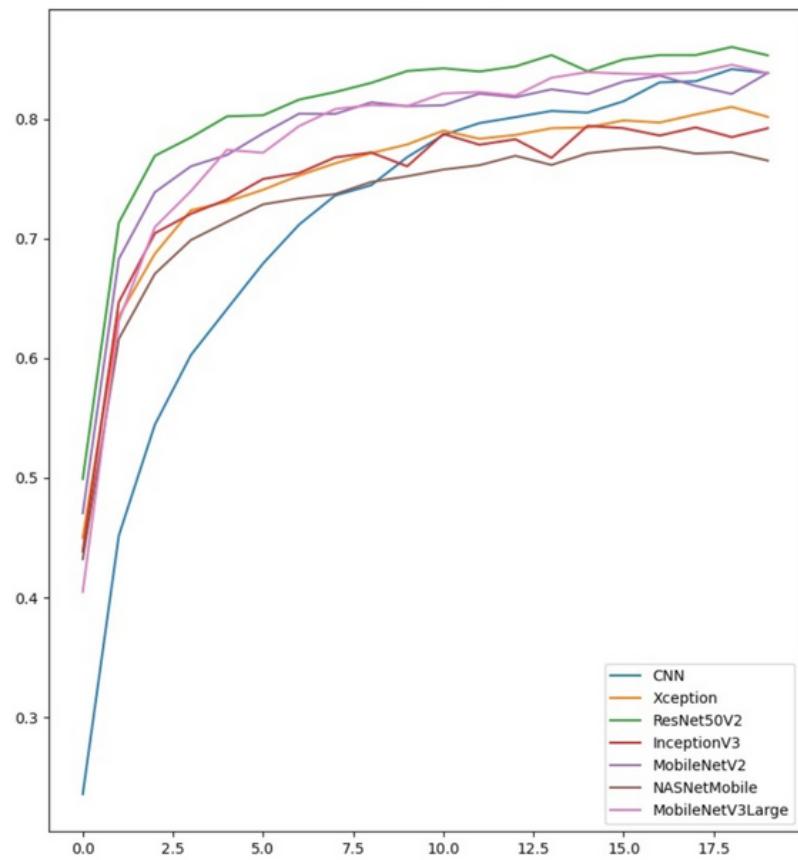


Le Grad-CAM est une méthode où l'on recherche à analyser quelles parties ont été utilisées par le réseau neuronal convolutif pour prendre sa décision finale. Cette méthode produit en sortie des cartes thermiques qui mettent en évidence les régions importantes d'une image.

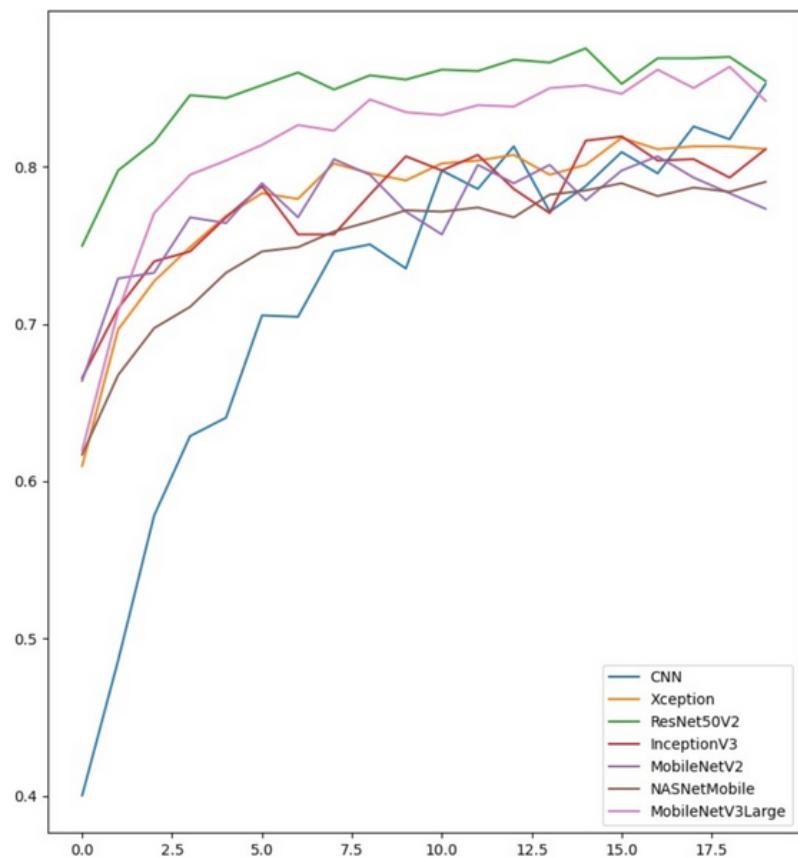
Ici, pour les prédictions correctes, le modèle se concentre sur les contours des plantes, tandis que pour les prédictions incorrectes, le modèle se base sur des zones inappropriées de l'image, ce qui conduit à des erreurs de classification.

Voici les résultats comparatifs obtenus au cours de nos expérimentations individuelles sur notre jeu de données :

Comparaison de la précision des modèles sur l'ensemble d'entraînement



Comparaison de la précision des modèles sur l'ensemble de validation



Nous nous sommes donc concentrés sur les modèles qui offrent les meilleures performances sur un temps d'entraînement réduit : ResNet50 et MobileNet.

Même si ces premiers tests ont utilisé MobileNetV2, des études ont montré que MobileNetV3 est encore plus performant.

[Searching for MobileNetV3, Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, Hartwig Adam; Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2022, pp. 1314-1324]

Pour la suite de cette étude, nous nous concentrerons donc sur les deux modèles : ResNet50v2 et MobileNetV3Large.

Néanmoins, il est important de noter que les temps d'entraînement sont bien différents entre ces deux modèles : il y a presque un rapport de 2.5 dans les temps de traitement. (62 min pour ResNet50v2 vs. 25 min sur MobileNetV3Large sur un Mac M1).

La taille de stockage de ces deux modèles est aussi très différente avec un rapport de x5 (ResNet50V2 \simeq 100 à 150 Mo / MobileNetV3Large \simeq 20 à 30 Mo)

Notons aussi que l'utilisation de MobileNetV3Large nous oblige à utiliser une taille d'image de 224x224.

Réglages fins

Le réglage fin de notre modèle s'est basé sur plusieurs pistes :

- équilibrage du jeu de données et suppression des incohérences
- traitement des images
- fine tuning du modèle pré-entraîné
- fine-tuning des couches d'apprentissage

Nous utiliserons un processus itératif : nous allons définir des étalons de départ et modifierons le minimum de paramètres à chaque fois afin de pouvoir quantifier l'effet de ces changements. Cette démarche ne nous assurera pas de trouver le meilleur modèle mais elle nous assurera de mieux comprendre l'effet de nos actions.

Dans cette logique d'être capable de comparer les résultats nous avons construit un moteur de test permettant de lancer plusieurs tests et de générer les rapports associés. Des outils existent déjà pour réaliser cela mais la construction de ce moteur a permis de mieux comprendre et s'approprier le fonctionnement de TensorFlow.

Ce moteur est disponible sur le GitHub :

https://github.com/DataScientest-Studio/AU23_Plantes

Les notebooks [1- Trainers Demo](#) et [2- Campaign Demo](#) expliquent et illustrent le fonctionnement du moteur.

Définition de l'étalon

Nous prendrons les hyper-paramètres suivants pour notre étalon – les mêmes seront utilisés pour MobileNetV3Large et ResNet50V2.

Seeds pour la reproductibilité des résultats :

- numpy : random.seed(777)
- tensorflow : random.set_seed(777)
- génération des datasets : seed=42

Compilation et entraînement

- batch_size = 32
- epochs = 32
- ReduceLROnPlateau(monitor='val_loss',
 factor=0.1,
 patience=3,
 verbose=1)
- EarlyStopping(monitor='val_loss',
 patience=5,
 restore_best_weights=True,
 verbose=1)
- ModelCheckpoint(filepath=file,
 monitor="val_categorical_accuracy",
 verbose=1,
 save_best_only=True)
- optimizer=Adam(learning_rate=1e-3),
- loss=CategoricalCrossentropy(),
- metrics=CategoricalAccuracy()
- class_weight=défini à partir du jeu de données

Définition du modèle

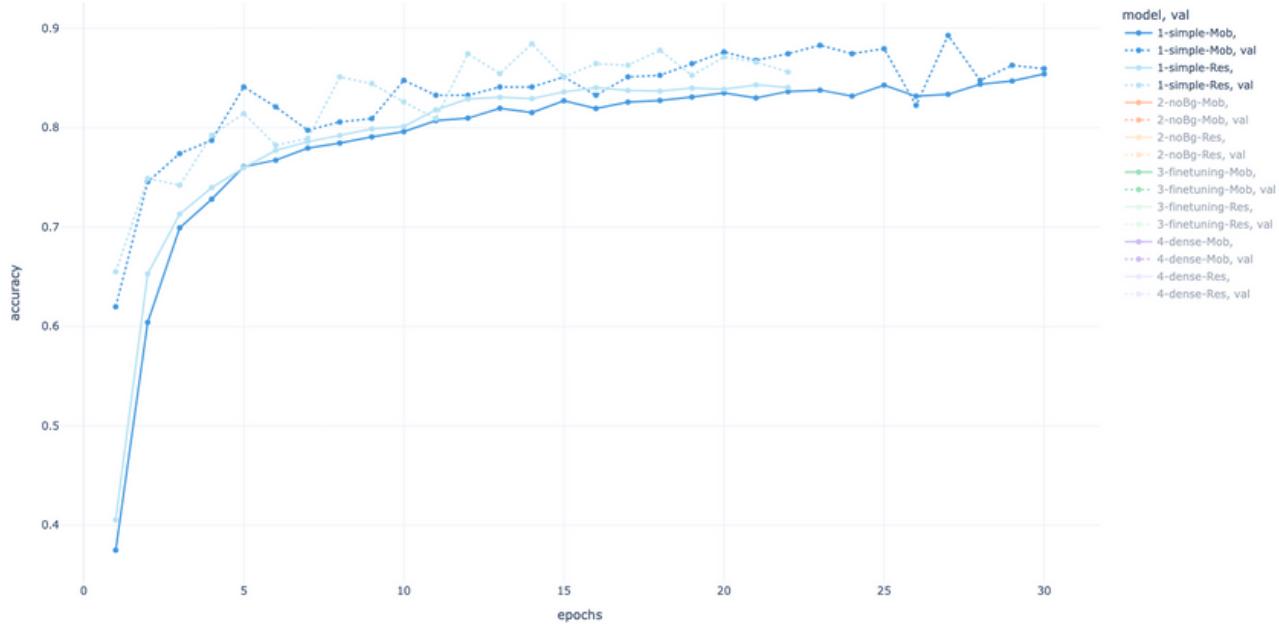
```
x = self.base_model.model.output
x = Dropout(0.2)(x)
output = Dense(12, activation='softmax', name='main')(x)
self.model = Model(inputs=self.base_model.model.input, outputs=output)
```

où base_model.model fait référence soit à une instance de MobilNetV3Large soit à une instance de ResNet50V2

Le code de ce test étalon est disponible sur le GitHub : [stage1.py](#)

Les résultats montrent ResNet un peu plus performant que MobileNet. Dans les deux cas, il y a peu d'overfitting.

Evolution de la précision des modèles sur les ensembles d'entraînement et de validation



Matrice de confusion MobileNetV3Large

Black-grass -	13	0	0	0	4	0	12	0	0	0	0	0
Charlock -	0	33	0	0	0	1	0	0	0	0	0	1
Cleavers -	0	2	34	0	0	0	0	0	2	0	0	0
Common Chickweed -	0	0	0	74	0	4	0	0	0	2	0	0
Common wheat -	0	0	0	0	14	2	2	0	0	1	0	2
Fat Hen -	0	1	0	0	1	45	1	0	0	0	0	6
Loose Silky-bent -	8	0	0	0	0	0	69	0	0	0	0	0
Maize -	0	0	0	0	0	0	0	20	0	0	0	0
Scentless Mayweed -	0	0	0	2	2	1	2	0	35	8	0	2
Shepherd's Purse -	0	0	3	2	0	0	0	0	16	1	0	0
Small-flowered Cranesbill -	0	0	2	0	0	0	0	1	67	0	0	0
Sugar beet -	0	0	0	1	0	1	0	0	1	53	0	0

Matrice de confusion ResNetv2

Black-grass -	13	0	0	0	0	4	0	12	0	0	0	0
Charlock -	0	33	0	0	0	1	0	0	0	0	0	1
Cleavers -	0	2	34	0	0	0	0	0	2	0	0	0
Common Chickweed -	0	0	0	74	0	4	0	0	0	0	0	2
Common wheat -	0	0	0	0	14	2	2	0	0	1	0	2
Fat Hen -	0	1	0	0	1	45	1	0	0	0	0	6
Loose Silky-bent -	8	0	0	0	0	0	69	0	0	0	0	0
Maize -	0	0	0	0	0	0	0	20	0	0	0	0
Scentless Mayweed -	0	0	0	2	2	1	2	0	35	8	0	2
Shepherd's Purse -	0	0	3	2	0	0	0	0	16	1	0	0
Small-flowered Cranesbill -	0	0	2	0	0	0	0	1	67	0	0	0
Sugar beet -	0	0	0	1	0	1	0	0	1	53	0	0

Dans les deux cas (MobileNet à gauche et ResNet à droite), les matrices de confusions montrent une dispersion des erreurs avec une confusion marquée entre les plantes Loose Silky-bent et Black-grass

Ces résultats vont nous servir d'étalon pour la suite.

Suppression du fond des images

La première étape d'optimisation a consisté à supprimer le fond des images afin de dissocier l'image de fond de l'image principale. Ce processus de suppression de fond se déroule en quatre étapes majeures :

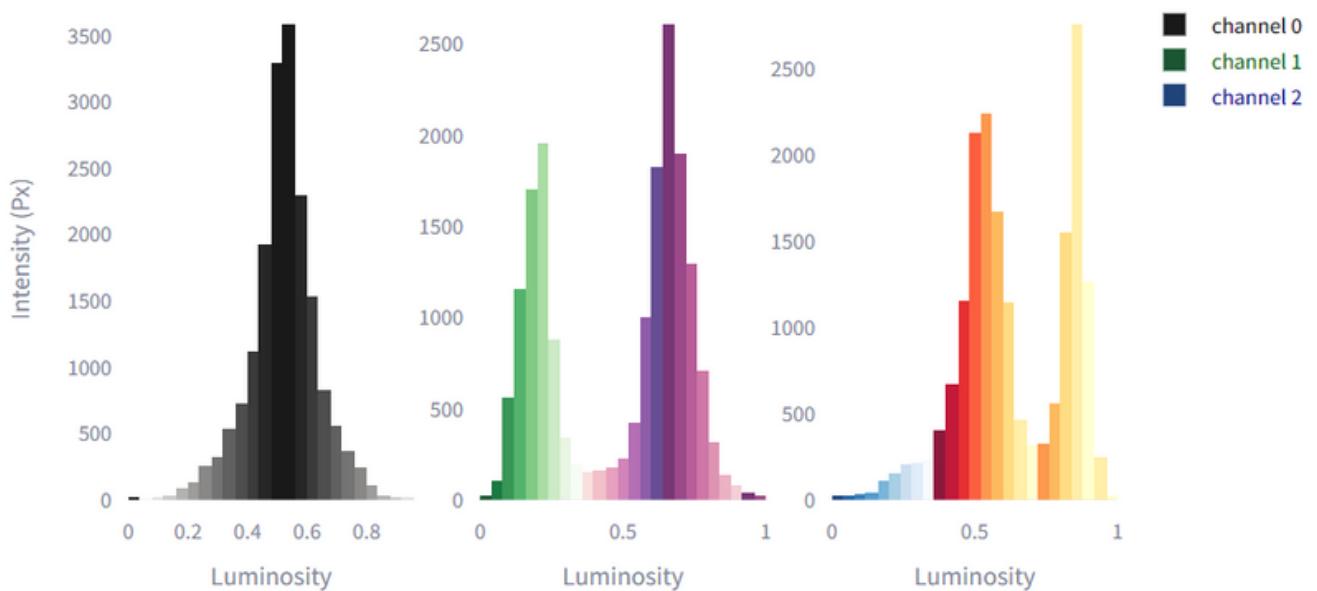
Conversion de l'image d'origine de l'espace colorimétrique RGB au RGB LAB.

L'espace LAB est un espace colorimétrique qui est conçu pour être perceptuellement uniforme, ce qui signifie que les distances entre les couleurs dans cet espace sont plus cohérentes avec la perception humaine de la couleur que dans l'espace RGB. Cet espace est souvent utilisé pour des tâches où la perception de la couleur par l'œil humain est importante et pour les projets de vision par ordinateur ou de l'imagerie médicale.

Visualisation des histogrammes de couleurs dans le RGB LAB.

Les histogrammes de couleurs sont des outils puissants et très utiles pour créer une image sémantique car ils permettent de comprendre la répartition des pixels et la luminance sur une image. Ils fournissent également un domaine de pixellisation pour l'objectif souhaité.

Représentation de l'espace RGB LAB



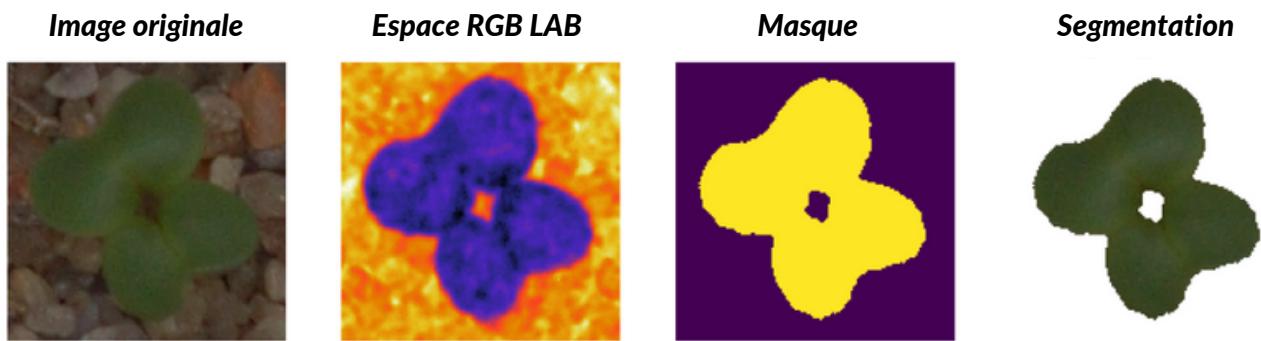
Création d'une image sémantique

La création de l'image sémantique consiste à créer des masques différents pour chaque objet contenu dans l'image par attribution des pixels grâce à l'érosion et la dilatation. L'érosion permet de supprimer tous les artefacts qui ne sont pas dans le domaine de pixellisation, par contre la dilatation permet de reconstruire ces pixels en se basant sur les proches voisins.

Superposition de l'image sémantique et l'image d'origine

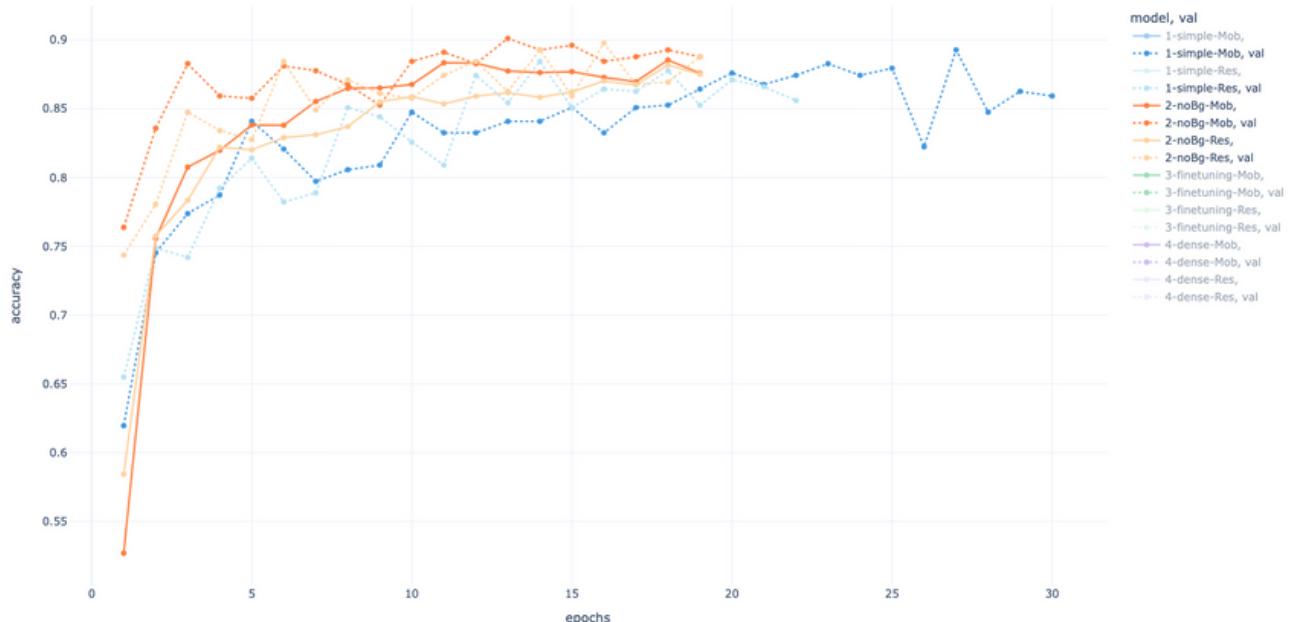
En superposant ces deux images, les masques contenus dans l'image sémantique sont convertis en RGB et représentent l'image finale segmentée.

Séparation de la plante du fond - Exemple sur la classe Charlock



Nous obtenons les résultats d'accuracy suivant :

Evolution de la précision des modèles suite à segmentation vs sans segmentation

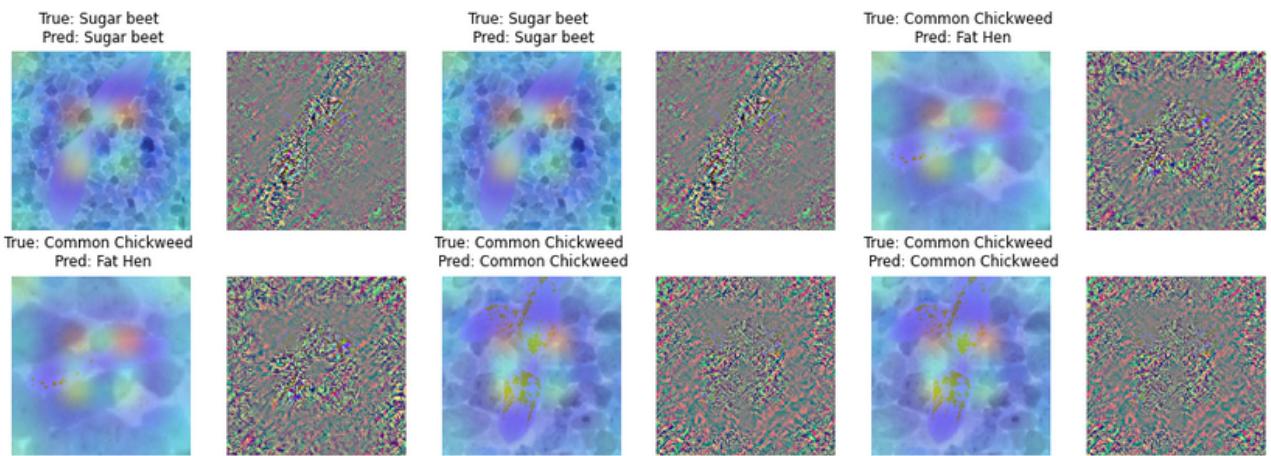


Le code de ce test incluant la suppression du fond est disponible sur le GitHub : [stage2.py](#). Pour les deux modèles, la suppression du fond des images permet d'augmenter l'accuracy. Mais on voit que c'est maintenant MobileNet qui est plus performant que ResNet.

Les matrices de confusions sont comme dans l'étalon assez dispersées avec une confusion forte entre Loose Silky-bent et Black-grass.

Ce choix est validé quand on fait l'analyse Grad-CAM. Le Grad-CAM est une méthode où l'on recherche à analyser quelles parties ont été utilisées par le réseau neuronal convolutif pour prendre sa décision finale. Cette méthode produit en sortie des cartes thermiques qui mettent en évidence les régions importantes d'une image.

Grad-CAM et Guided Grad-CAM sur bonnes prédictions du MobileNetV3 sans segmentation

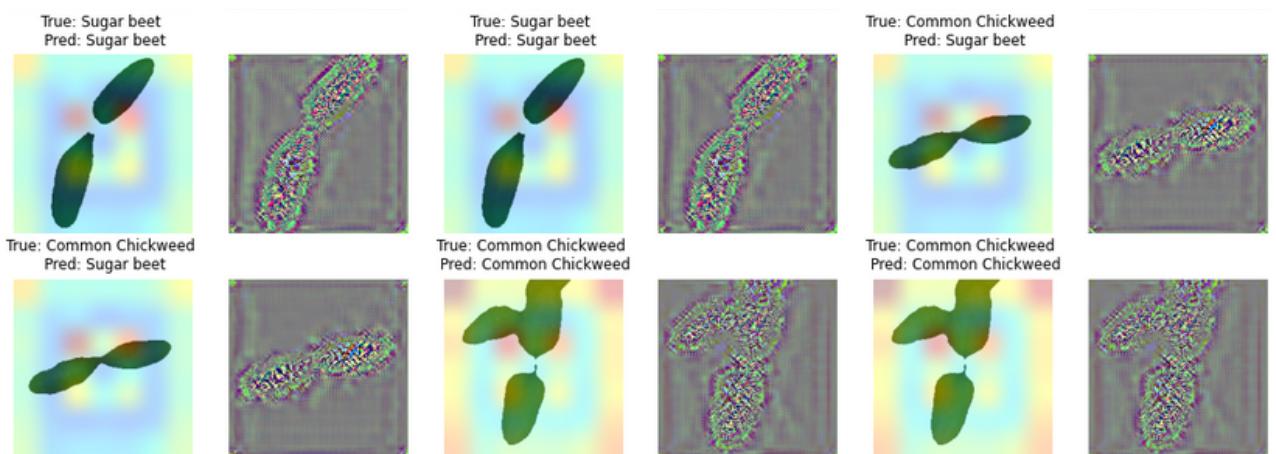


De prime abord on note que le modèle n'utilise pas vraiment la plante pour faire sa classification, et se concentre beaucoup sur les bords de l'image, c'est-à-dire le fond, c'est d'autant plus parlant quand on regarde le Guided Grad-CAM l'image en gris sur la droite.

C'est-à-dire que les modèles accordent une grande importance au fond, ce qui a un réel impact sur les performances des modèles, et qui justifie également le choix de faire la segmentation des images.

Sur l'image suivante on peut voir l'impact de la segmentation sur la prise de décision du CNN:

Grad-CAM et Guided Grad-CAM sur bonnes prédictions du MobileNetV3Large avec segmentation



En analysant le Guided Grad-CAM on remarque immédiatement que l'attention des modèles s'est beaucoup plus concentrée sur la plante en elle-même.

Ce qui nous offre deux avantages :

- une performance plus élevée des modèles.
- une portabilité de ceux-ci, puisqu'il suffit de faire une segmentation pour pouvoir analyser des images provenant de datasets différents. Tout en gardant à l'esprit que la segmentation appliquée ici, aura peut-être besoin d'être réadaptée à une typologie d'images différentes.

Fine-Tuning des CNN pré-entraînés.

Nous avons ensuite travaillé sur le fine-tuning de MobileNet et ResNet.

La méthode préconisée consiste à dégeler un certain nombre de couches et de les ré-entraîner. Une étude montre que le taux optimal est de 50% [Fine-Tuning A Lightweight Convolutional Neural Networks for COVID-19 Diagnosis – Conference: CSBio2020: The 11th International Conference on Computational Systems-Biology and Bioinformatics]. Nous avons aussi expérimenté cela comme le montre le graphique en Annexe 1

La documentation de Keras indique que ce processus doit se faire en 2 étapes :

1. entraînement de la classification n'incluant pas le CNN (par ex 12 epochs)
2. entraînement sur les derniers 50% des couches du CNN (par ex sur 20 epochs)

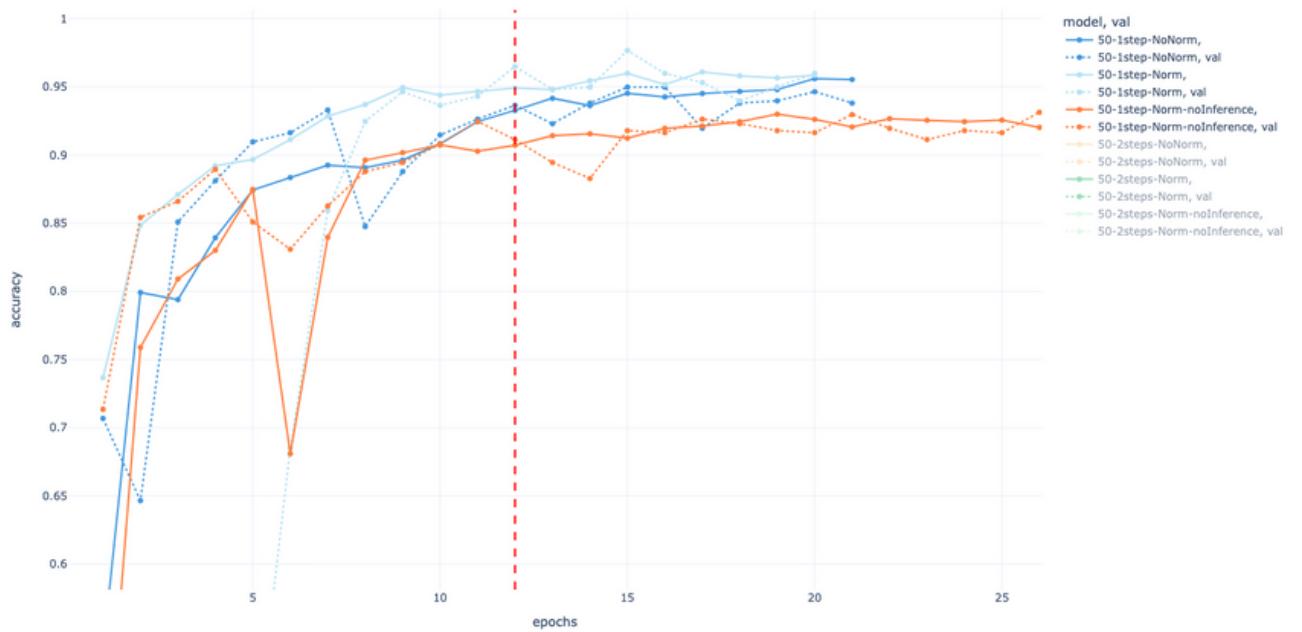
Elle indique aussi que dans la seconde phase, il faut positionner le paramètre `training=False` afin que les couches de dropout s'annulent et que les couches de Normalisation passent en mode inférence.

Néanmoins plusieurs retours d'expérience sur le web indiquent d'annuler directement l'entraînement des couches de normalisation sans changer le mode d'inférence. Quand d'autres ré-entraînent même ces couches de normalisation et ce directement dans la première phase d'entraînement.

Afin de se positionner dans ces diverses méthodes, nous avons entraîné et comparé avec le CNN MobileNet les cas suivants :

- une seule phase : entraînement des derniers 50% des couches :
 - a. entraînement agressif
 - b. sans entraînement des couches de normalisation
 - c. avec inférence sur la normalisation (`training=False`)

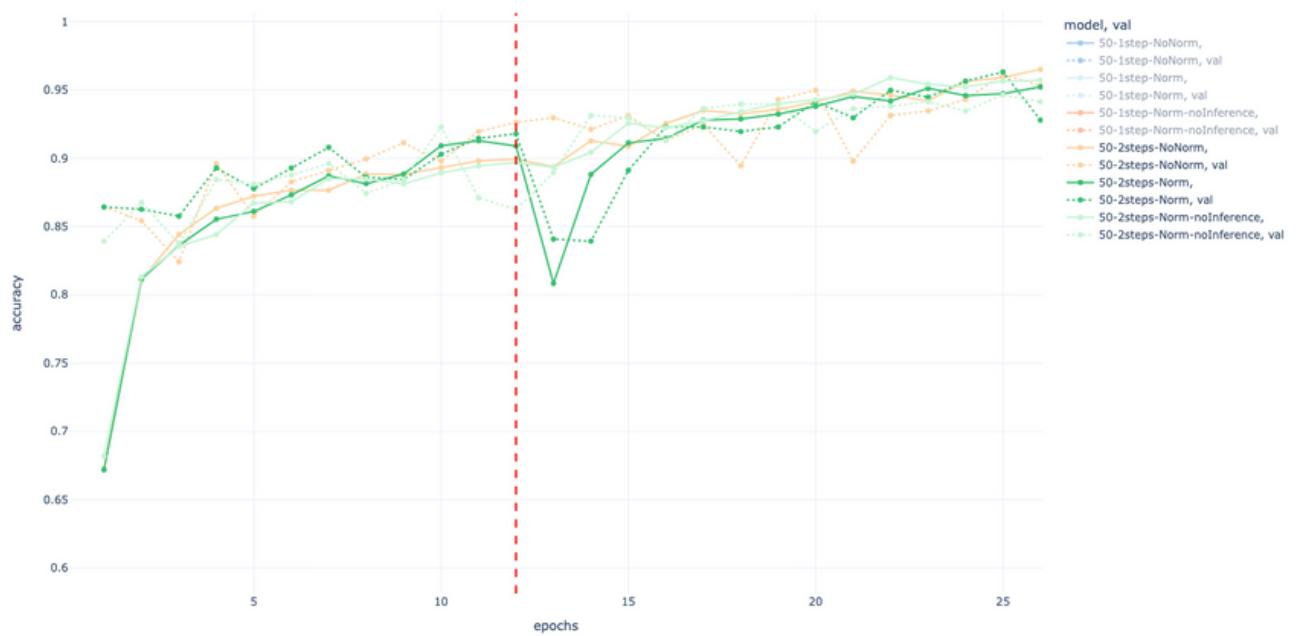
Evolution de la précision des modèles suite à Fine-Tuning dès la 1ère époque



Les résultats montrent clairement de meilleures performances avec l'entraînement direct.

- deux phases : entraînement dans une seconde phase des 50% des couches dégelées du modèle :
 - entraînement agressif
 - sans entraînement des couches de normalization
 - avec inférence sur la normalisation (training=False)

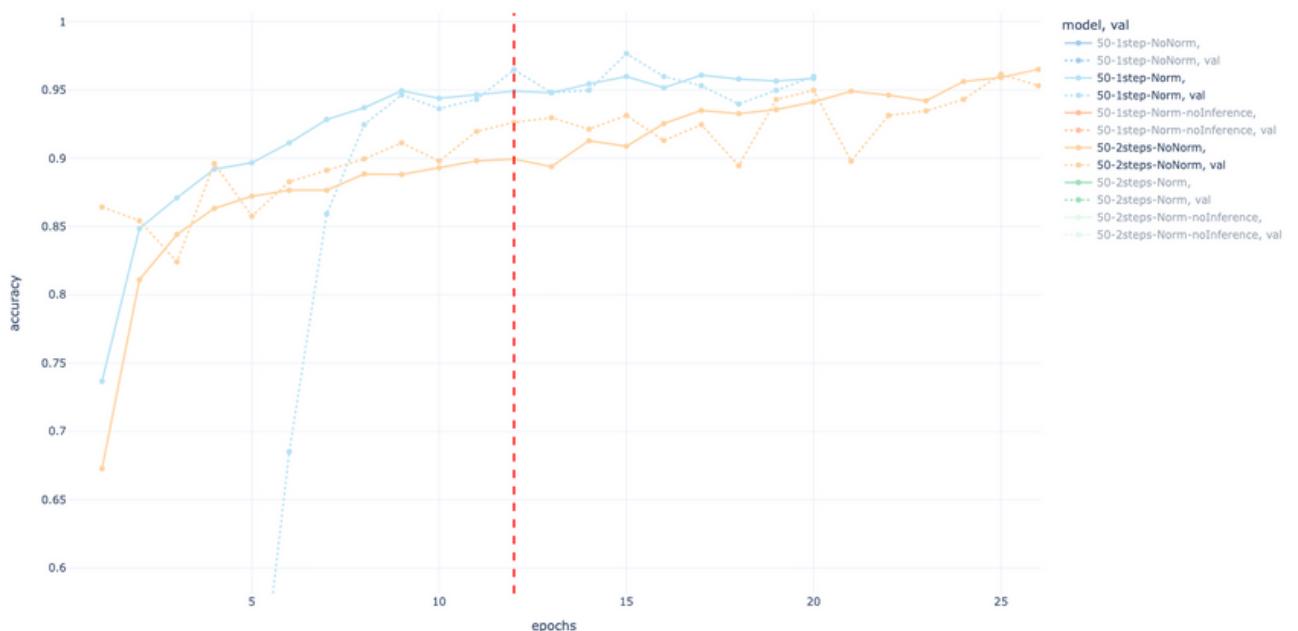
Evolution de la précision des modèles suite à Fine-Tuning en deux étapes



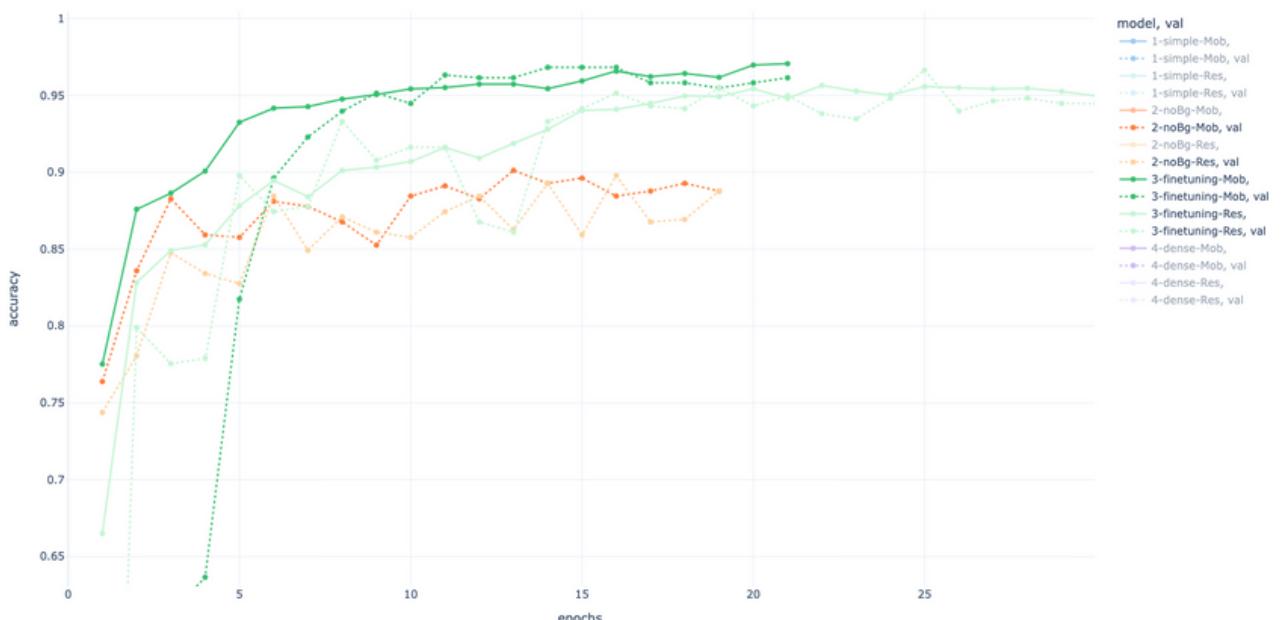
Les résultats sont plus rapprochés : mais les méthodes supprimant la normalisation ou les positionnant en mode inférence sont sensiblement meilleures.

Néanmoins si on compare les meilleures méthodes en 1 phase et en 2 phases, contre toute attente, il apparaît clairement que la méthode agressive en une phase unique incluant l'entraînement des couches de normalisation est plus performante sur nos données. Le code de cette expérimentation est disponible sur : [testClassif2.py](#)

Comparaison des deux méthodes sur les deux modèles les plus performants



Nous appliquons le principe de fine-tuning agressif sur nos CNN en repartant du test précédent : c'est à dire étalon + suppression du fond des images. (Code sur GitHub : [stage3.py](#))



Pour les 2 CNN, l'accuracy est bien meilleure (0.96). Il y a un léger overfitting. Mais les matrices de confusion présentent beaucoup moins de dispersion que précédemment. Les confusions se situent principalement entre Loose Silky-bent et Black Grass.

Le fine-tuning améliore grandement les performances par rapport à nos précédents tests avec des erreurs de plus en plus ciblées.

Matrice de confusion MobileNetV3Large

	Black-grass	Charlock	Cleavers	Common Chickweed	Common wheat	Fat Hen	Loose Silky-bent	Maize	Scentless Mayweed	Shepherd's Purse	Small-flowered Cranesbill	Sugar beet
Black-grass	21	0	0	0	1	0	7	0	0	0	0	0
Charlock	0	34	1	0	0	0	0	0	0	0	0	0
Cleavers	0	0	38	0	0	0	0	0	0	0	0	0
Common Chickweed	0	0	0	79	0	0	0	0	1	0	0	0
Common wheat	0	0	0	0	21	0	0	0	0	0	0	0
Fat Hen	0	0	0	0	0	54	0	0	0	0	0	0
Loose Silky-bent	11	0	0	0	0	0	66	0	0	0	0	0
Maize	0	0	0	0	0	0	0	20	0	0	0	0
Scentless Mayweed	0	0	0	0	0	0	0	52	0	0	0	0
Shepherd's Purse	0	0	0	0	0	0	0	0	22	0	0	0
Small-flowered Cranesbill	0	0	0	0	0	0	0	0	0	70	0	0
Sugar beet	0	0	1	0	0	0	0	0	0	0	55	0

Matrice de confusion ResNetv2

	Black-grass	Charlock	Cleavers	Common Chickweed	Common wheat	Fat Hen	Loose Silky-bent	Maize	Scentless Mayweed	Shepherd's Purse	Small-flowered Cranesbill	Sugar beet
Black-grass	23	0	0	0	2	0	4	0	0	0	0	0
Charlock	0	34	0	0	0	0	0	0	0	0	0	1
Cleavers	0	0	38	0	0	0	0	0	0	0	0	0
Common Chickweed	0	0	0	79	0	0	0	0	1	0	0	0
Common wheat	0	0	0	0	20	1	0	0	0	0	0	0
Fat Hen	1	0	0	0	0	53	0	0	0	0	0	0
Loose Silky-bent	17	0	0	0	0	0	60	0	0	0	0	0
Maize	0	0	0	0	0	0	20	0	0	0	0	0
Scentless Mayweed	0	0	0	0	0	0	0	52	0	0	0	0
Shepherd's Purse	0	0	0	0	0	0	0	0	22	0	0	0
Small-flowered Cranesbill	0	0	0	0	0	0	0	0	0	70	0	0
Sugar beet	0	0	1	0	0	0	0	0	0	0	54	0

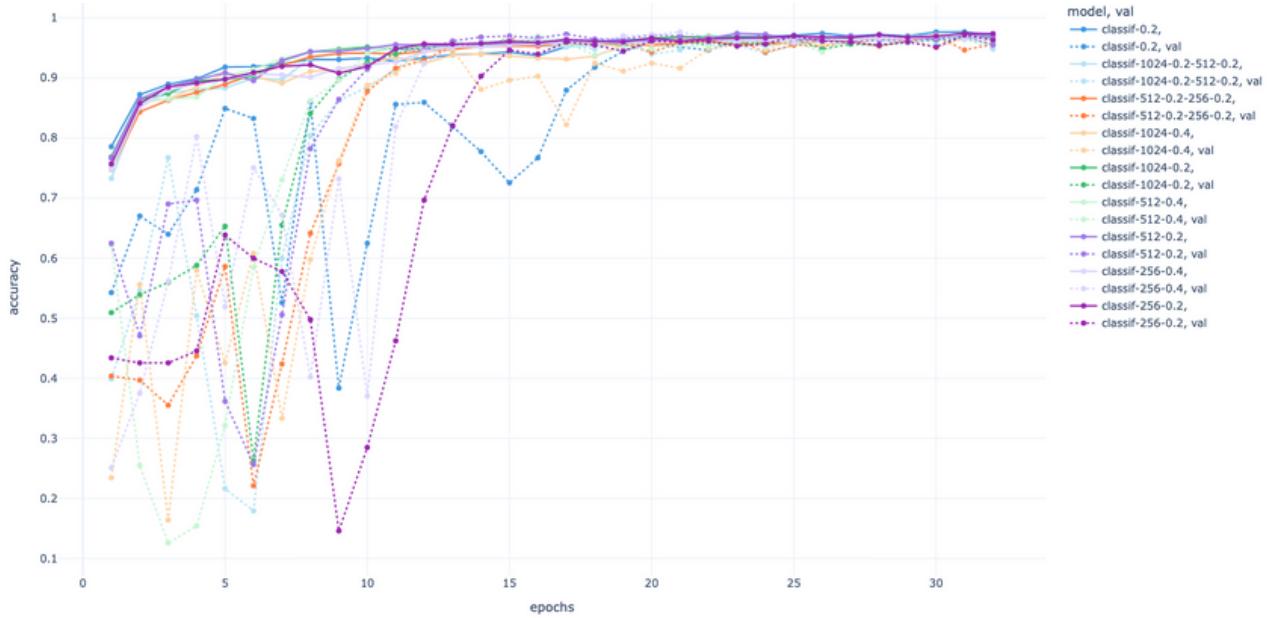
Amélioration des couches de classification

Jusqu'à présent nous nous sommes contentés d'un Dropout(0.2) et de la Dense(12) en couches de classification. Sur la base de nos derniers résultats, nous avons exploré plusieurs options supplémentaires en nous concentrant sur MobileNet : nous avons utilisé

- une régularisation L2(1e-4) et une activation ReLU sur les couches Dense pour limiter le sur-apprentissage (sauf la dernière qui est en Softmax)
- avec les combinaisons suivantes :
- Drop(0.2) - Dense(12)
- Dense(256) - Drop(0.2) - Dense(12)
- Dense(256) - Drop(0.4) - Dense(12)
- Dense(512) - Drop(0.2) - Dense(12)
- Dense(512) - Drop(0.4) - Dense(12)
- Dense(1024) - Drop(0.2) - Dense(12)
- Dense(1024) - Drop(0.4) - Dense(12)
- Dense(512) - Drop(0.2) - Dense(256) - Drop(0.2) - Dense(12)
- Dense(1024) - Drop(0.2) - Dense(512) - Drop(0.2) - Dense(12)

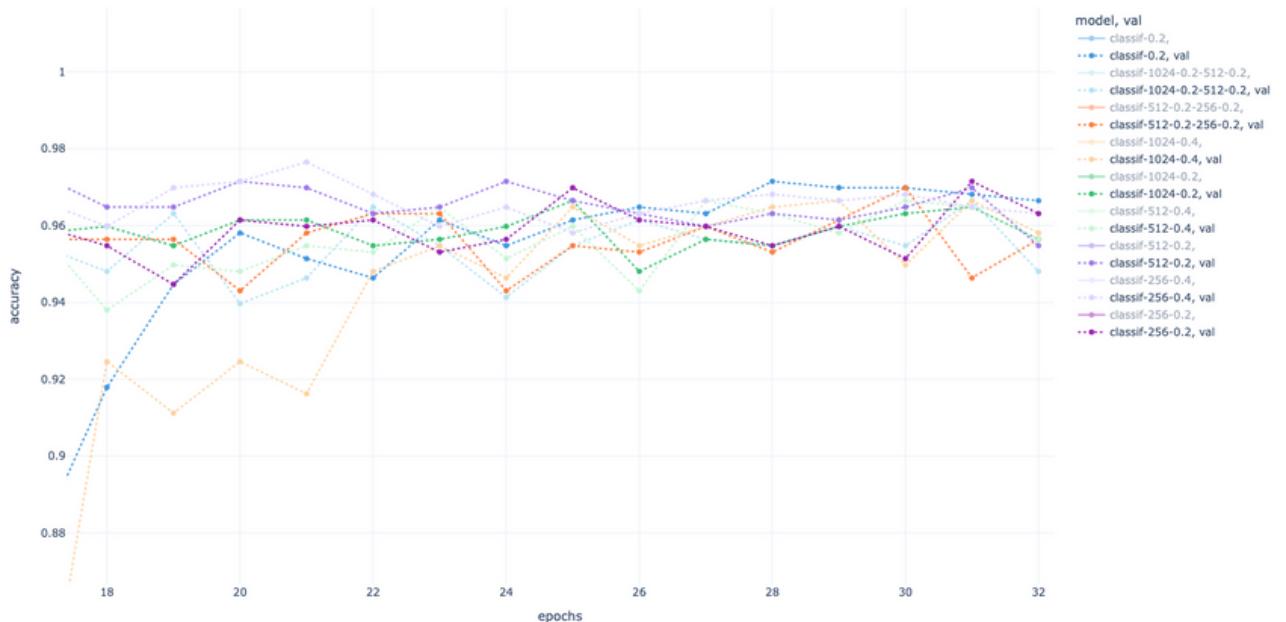
Le code de cette expérimentation est disponible sur le GitHub : [testClassif2.py](#)

Comparaison des modèles selon les différentes structures de couches de classification



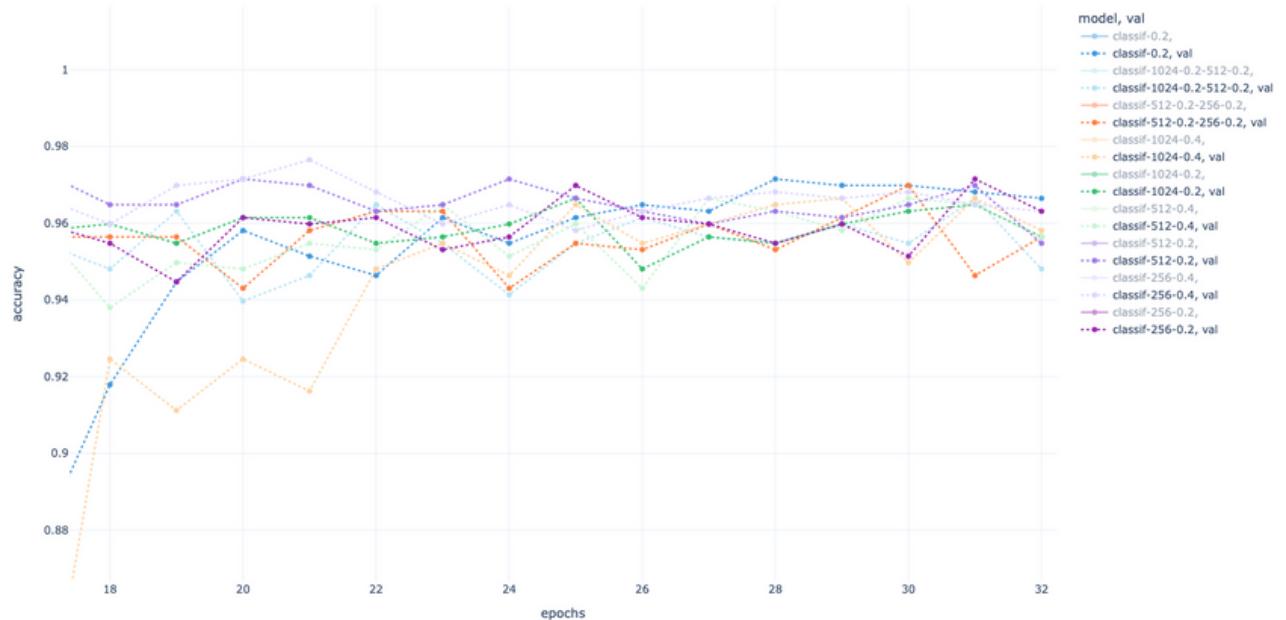
Les résultats sont très similaires. On voit bien que dans tous les cas, l'ajout du L2 combiné au fine-tuning agressif apporte beaucoup de bruit en début d'entraînement : il faudra presque 20 epochs pour que la val_accuracy se stabilise. Nous avons été du coup obligés de rallonger les patiences de ReduceLROnPlateau et EarlyStopping à 5 et 10 respectivement au lieu de 3 et 5.

Zoom et comparaison des modèles les plus performants



En regardant de plus près, c'est finalement la classification la plus simple (Drop(0.2), Dense(12, regul=L2)) qui obtient les meilleurs résultats. C'est donc cette classification que nous appliquerons pour continuer notre comparatif (Code GitHub : [stage4.py](#))

Les résultats sont légèrement meilleurs pour MobileNet que dans les tests précédents mais moins bons pour ResNet. On remarque aussi que le temps de stabilisation est plus long avec la régularisation L2.



Quant aux matrices de confusion, elles montrent des résultats encore plus compacts pour MobileNet avec une confusion presque entièrement concentrée sur Loose Silky-bent et Black-grass.

Matrice de confusion MobileNetV3Large

Matrice de confusion ResNetv2

Notre modèle final prendra donc les couches de classification (Dropout(0.2), Dense(12, regul=L2)) appliquées au CNN MobileNet avec les derniers 50% des couches entraînables et la segmentation des images par suppressions du fond.

Résultats finaux et pistes d'amélioration

Performance et comportement du modèle final

En résumé, notre meilleur modèle est basé sur

- une segmentation des images par suppression du fond
- du transfer-learning avec le modèle MobileNetV3Large entraîné sur imagenet,
- un finetuning agressif des dernières couches (50%) de ce modèle
- des couches de classification simples : Drop(0.2) puis Dense(12, softmax, regul=L2)

Code sur GitHub : [MobileNetV3 dans stage4.py](#)

Modèle sérialisé sur GitHub : [4-dense-Mob_model.h5](#)

Ce modèle fait 33.9Mo ce qui est plutôt raisonnable et facilement portable sur un appareil léger comme un robot désherbeur.

Le rapport de classification de notre meilleur modèle est le suivant :

	precision	recall	f1-score	support
Black-grass	0.72	0.79	0.75	29
Charlock	1.00	0.97	0.99	35
Cleavers	0.97	1.00	0.99	38
Common Chickweed	1.00	0.99	0.99	80
Common wheat	0.91	1.00	0.95	21
Fat Hen	1.00	1.00	1.00	54
Loose Silky-bent	0.93	0.87	0.90	77
Maize	1.00	1.00	1.00	20
Scentless Mayweed	0.98	1.00	0.99	52
Shepherd's Purse	1.00	1.00	1.00	22
Small-flowered Cranesbill	1.00	1.00	1.00	70
Sugar beet	1.00	1.00	1.00	56
accuracy			0.97	554
macro avg	0.96	0.97	0.96	554
weighted avg	0.97	0.97	0.97	554

La performance est très correcte. C'est d'autant plus le cas que la confusion entre Loose Silky-bent et Black Grass n'est probablement pas très pénalisante puisque ces deux plantes sont considérées comme des mauvaises herbes et aucune n'a d'usage de culture ou d'engrais. Elles sont aussi toutes deux très nuisibles à la culture de céréales (certains parlent de 40% de perte de rendement [cf. wikipedia]).

Dans ce contexte, nous pourrions considérer ces 2 plantes comme équivalentes et les traiter comme une seule classe dans l'analyse des modèles.

Cette association fait l'objet du notebook [3- Model results.ipynb](#) sur le github et notre modèle a alors un rendement de 0.99% :

	precision	recall	f1-score	support
Charlock	1.00	0.97	0.99	35
Cleavers	0.97	1.00	0.99	38
Common Chickweed	1.00	0.99	0.99	80
Common wheat	0.91	1.00	0.95	21
Fat Hen	1.00	1.00	1.00	54
Maize	1.00	1.00	1.00	20
Scentless Mayweed	0.98	1.00	0.99	52
Shepherd's Purse	1.00	1.00	1.00	22
Silky-Bent - Black Grass	1.00	0.98	0.99	106
Small-flowered Cranesbill	1.00	1.00	1.00	70
Sugar beet	1.00	1.00	1.00	56
accuracy			0.99	554
macro avg	0.99	0.99	0.99	554
weighted avg	0.99	0.99	0.99	554

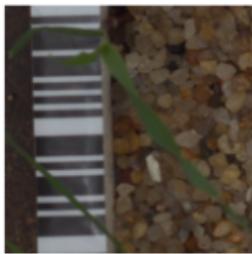
La faiblesse résiderait tout de même dans la précision pour Common Wheat. En effet, la confusion se fait avec des mauvaises herbes. Nous aurions préféré trouver cette confusion sur le rappel pour s'assurer de l'absence de mauvaise herbe.



Voyons tout de même à quoi correspondent ces erreurs :

4-dense-Mob – Result Samples

True: Silky-Bent - Black Grass
Pred: Common wheat



True: Charlock
Pred: Cleavers



True: Common Chickweed
Pred: Scentless Mayweed



True: Silky-Bent - Black Grass
Pred: Common wheat



Rappelons l'apparence de common wheat, Loose silky-bent et Black-grass, les voici dans l'ordre :



A l'œil nu la différence est difficile à voir et lorsque l'on regarde les 2 erreurs faites il est compréhensible que le modèle confonde.

Conclusion et pistes d'amélioration

Cette étude a exploré le jeu de données [V2 Plant Seedlings Dataset](#) consistant à détecter les mauvaises herbes parmi des semis ou jeunes pousses.

Nous avons défini un modèle final CNN sur la base de transfer-learning avec MobileNetV3Large et de la segmentation des images permettant de supprimer l'image de fond. Cette approche a permis au modèle de se focaliser uniquement sur la plante pour faire une prédiction comme l'a montré le GRAD CAM.

À travers cette étude, nous montrons l'efficacité et la robustesse de la segmentation de l'image sémantique dans les problématiques de deep learning.

Les résultats sont satisfaisants ($F1 : 97\%$) voire plus que satisfaisants ($F1 : 99\%$) si l'on remet l'usage du modèle dans le contexte d'un robot désherbeur qui doit supprimer les mauvaises herbes des plantes cultivées.

Le modèle est léger (34 Mo) et facilement utilisable dans ce contexte.

Notre méthodologie de design découpée en étapes ne nous assure pas d'avoir trouvé le meilleur modèle. En effet, chaque changement amène des effets de bord qui pourraient remettre en question les décisions prises précédemment. Néanmoins, l'étude séparée des différents hyperparamètres nous a permis de mieux appréhender leurs potentiels effets et surtout de voir que les certitudes ne sont pas compatibles avec le design de modèle. Dans cette activité, la démarche empirique reste de mise.

Nous avons plusieurs pistes d'amélioration pour continuer cette étude :

- Être capable de détecter les jeunes pousses au sein d'un sol qui en est rempli (détection d'objet et reconnaissance) et permettre ainsi à un robot désherbeur d'avoir une vision large lors de son travail. Cependant, pour y arriver, un travail énorme est effectué en amont pour créer un nouveau dataset avec les boîtes englobantes associées à chaque plante pour un projet de vision par ordinateur robuste.

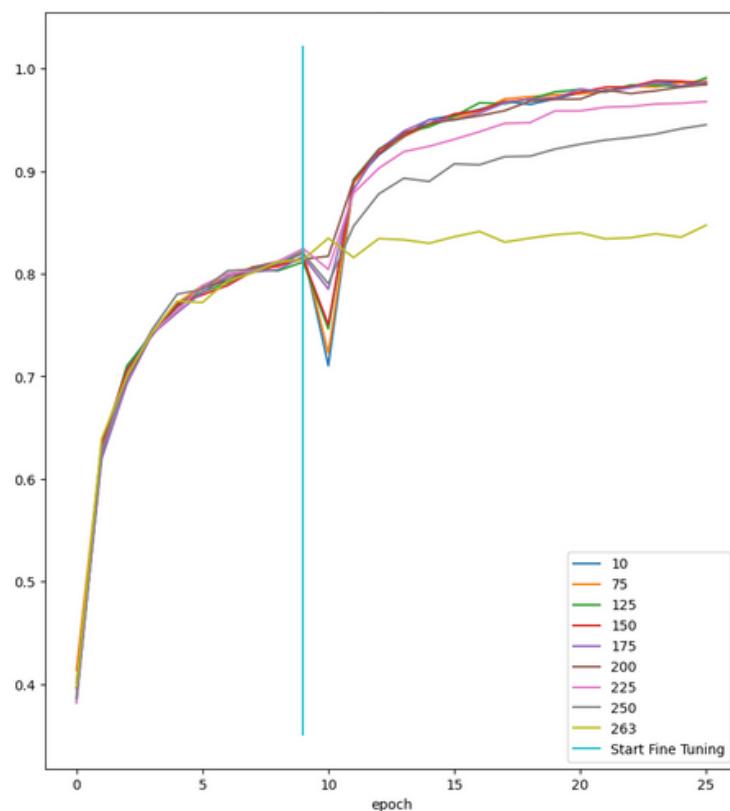


- Il faudra explorer les méthodes YOLO, SSD ou Faster R-CNN et voir le ratio légèreté/performance de ces modèles afin de choisir celui qui serait le plus approprié et adapté à la problématique posée.
- Déploiement du modèle sur un cluster pour un suivi et une maintenance continue.
- Définir une infrastructure logiciel légère qui pourrait être embarquée sur un robot capable de se déplacer et de couper ces mauvaises herbes.
- L'intégration d'un mécanisme de feedback humain (cf. annexes) offre le potentiel d'accumuler un jeu de données complémentaire qui, en atteignant un volume conséquent, pourrait servir de base pour un apprentissage par renforcement. Ce processus permettrait d'affiner les performances du modèle en intégrant des nouvelles données annotées et plus variées que le jeu de données initial.

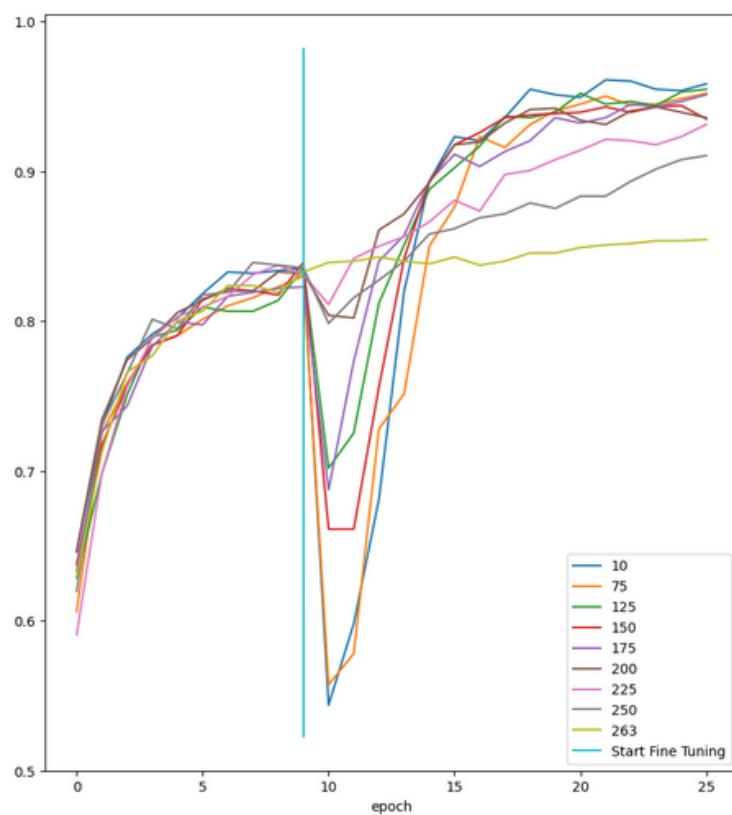
Annexes

1. Graphique comparatif de la précision en fonction du nombre de layers dé-gelés

Comparaison sur l'ensemble d'entraînement



Comparaison sur l'ensemble de validation



Annexes

2. Mise en production du modèle - Démonstration disponible sur le Streamlit

Prédiction sur une image présente dans le lot de Test - Modèle non entraîné sur ces images

Lot d'images de test

Informations

Sélectionnez une image

small-flowered_cranesbill_17.png

Aperçu de l'image à prédire



Selon le modèle, il s'agit de l'espèce Small-flowered Cranesbill

Correcte

Incorrecte

La prédiction est correcte - Enregistrement du Feedback

Sélectionnez une image

small-flowered_cranesbill_17.png

Aperçu de l'image à prédire



Selon le modèle, il s'agit de l'espèce Small-flowered Cranesbill

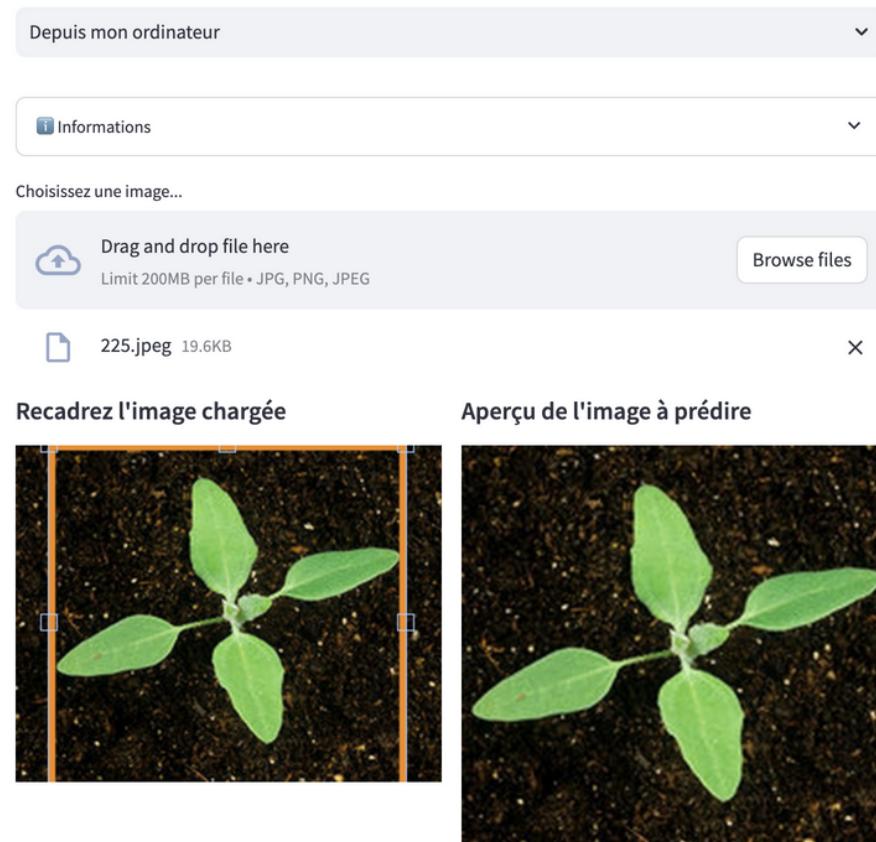
Correcte

Incorrecte

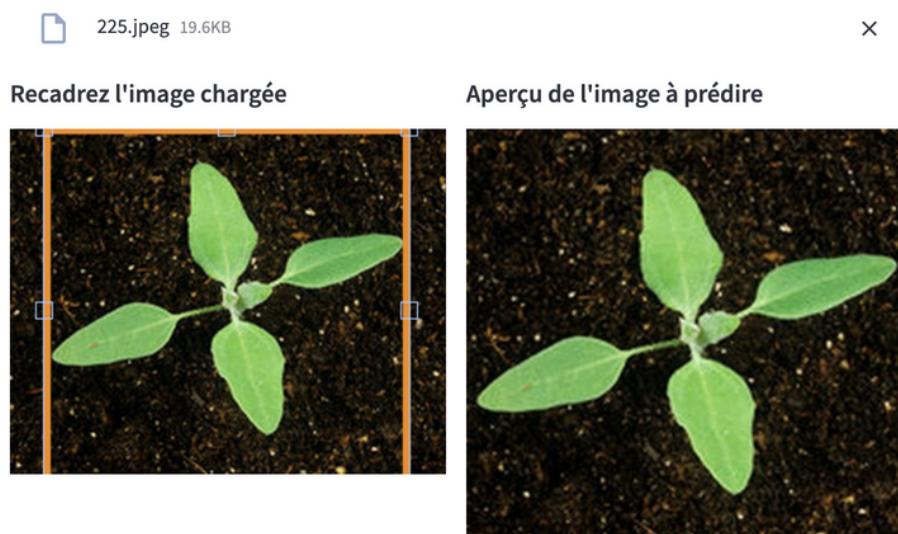
Feedback enregistré avec succès !

Annexes

Téléchargement d'une image depuis l'ordinateur - Fonction de recadrage



La prédiction est incorrecte - Enregistrement du Feedback avec mention de la bonne classe



Selon le modèle, il s'agit de l'espèce **Sugar beet**

Précisez la bonne classe :

X

X ▼

Annexes

Prédiction à partir de l'URL d'une image - Bonne prédiction

A partir d'une URL

Informations

Entrez l'URL de l'image

https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcT6iDEqnD3Mlg4rZWA_NRp-hH6wSLGBkyWF

Recadrez l'image chargée



Aperçu de l'image à prédir



Selon le modèle, il s'agit de l'espèce Shepherd's Purse

Correcte

Incorrecte

Image de référence - Cliquez sur l'image pour être redirigé

