ChampiPy

Cahier des charges

Contexte et objectifs

ChampiPy est projet de Deep Learning permettant d'obtenir le nom d'une espèce de champignon grâce à une photo.

Le modèle a été réalisée dans le cadre de la formation Data Scientist de Janvier 2022 par Romain COUSSY, Emeline SILVESTRE, Paul VENTURA et moi-même.

L'objectifs maintenant est de mettre ce modèle à disposition d'un groupe d'utilisateur confirmés sur https://mushroomobserver.org/ afin de vérifier les propositions des utilisateurs moins expérimentés.

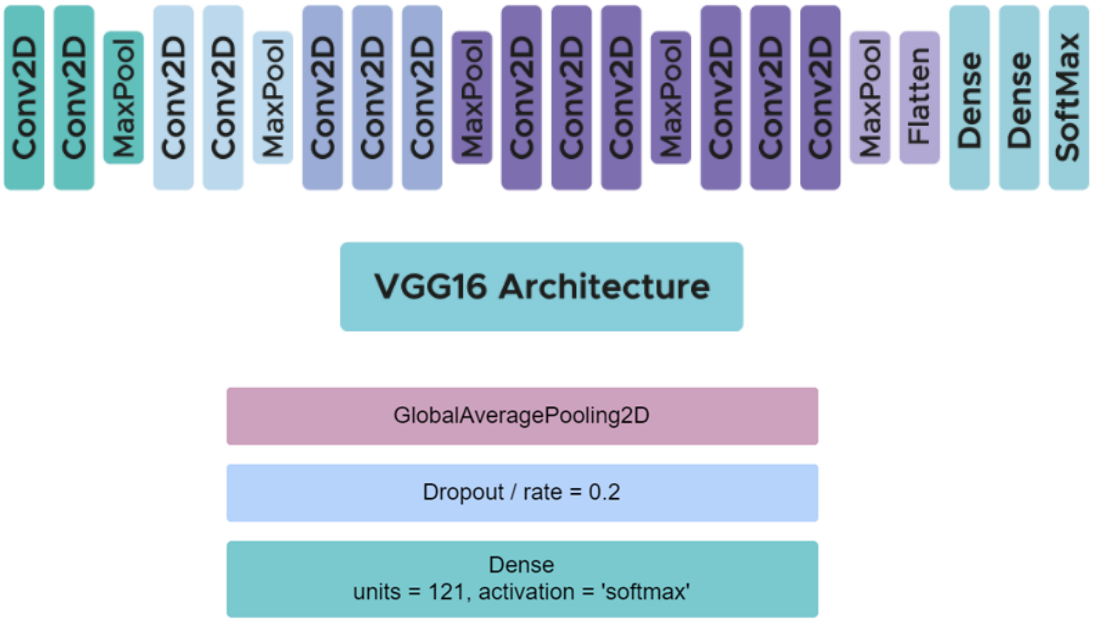
Plus particulièrement, à partir de l'URL d'une image, le modèle prédira l'espèce et indiquera le pourcentage de précision de cette prédiction.

L'utilisateur à l'origine de cette demande, Bob, sera également l'administrateur de l'application.

L'exploitation du ce modèle se fera, dans un premier temps, par l'intermédiaire d'une API puis son intégration au site sera étudiée.

Modèle

Le modèle est basé sur VGG16 avec 3 couches supplémentaires. Il est un bon compromis entre rapidité d'entrainement et précision de prédiction.

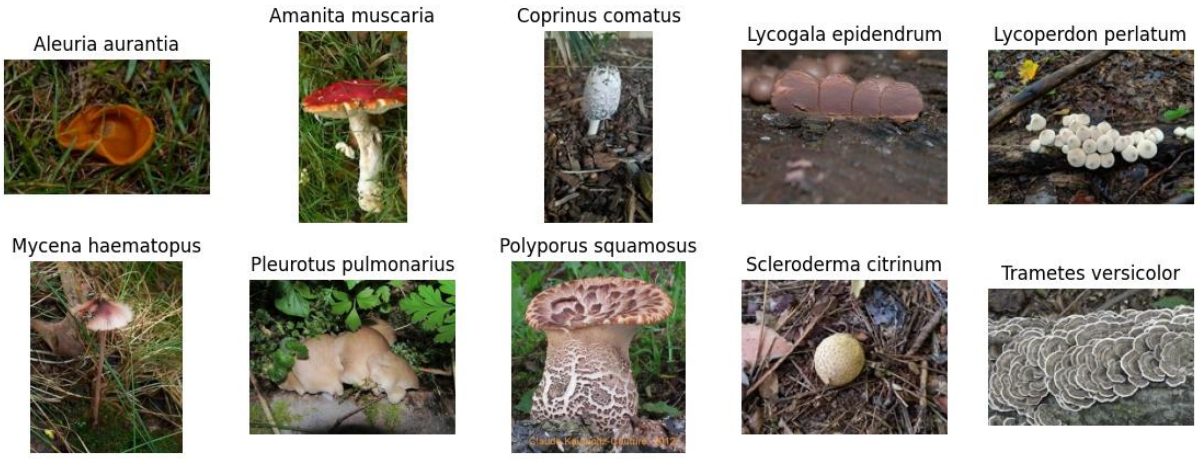


Néanmoins, pour ce projet, 2 contraintes ont été introduites :

- Le temps d'entrainement ne devrait pas dépasser 30 minutes
- Le jeu de données pour l'entrainement sera de 1.000 images

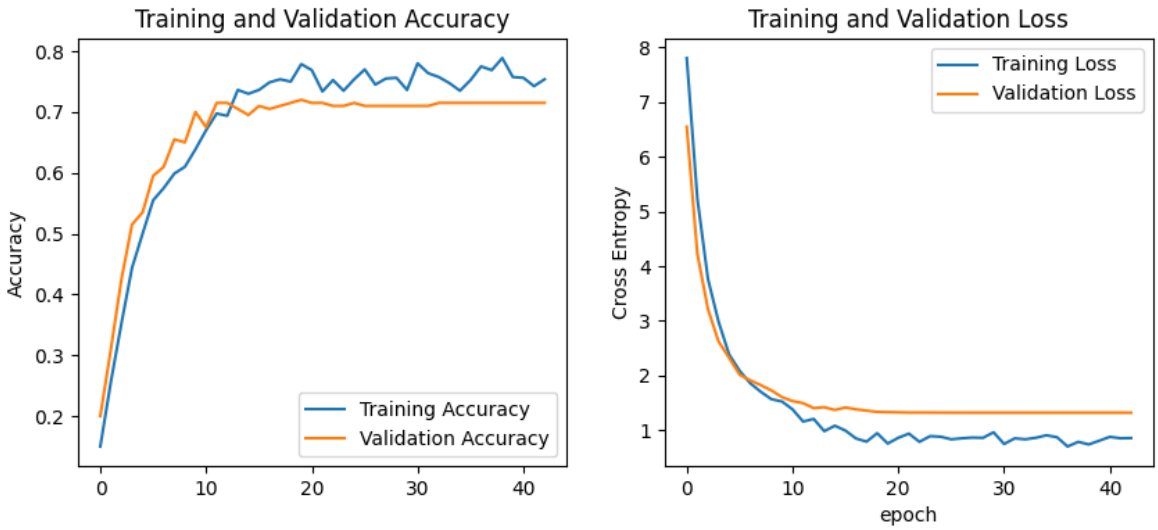
Afin de réduire les temps d'entrainements et le volume de données nécessaire à cet entrainement, j'ai réduit le nombre d'espèce utilisés par le modèle. De 121 espèces nous passons à 10, avec 100 images par espèce. Le jeu de données sera à 80% pour l'entrainement et 20% pour la validation. Ces 10 espèces sont celles qui ont obtenu le meilleur score de prédiction.

Les voici :



Les performances attendues sont une **précision supérieure 70%**.

Voici les performances du dernier modèle entrainé :



Le modèle devra être ré-entrainé si 500 nouvelles images sont disponibles.

L'optimisation des hyperparamètres doit être possible.

La base de données

Les données disponibles proviennent du site https://mushroomobserver.org/. Nous avons à disposition plus de 500 photos par espèce, stockées sur un disque dur au format jpeg.

Pour mesurer la performance du modèle dans le temps, il sera nécessaire de sauvegarder les données et résultats de prédictions.

Les photos utilisées par les utilisateurs dans la vie de l'application seront utilisées pour améliorer le modèle si leur score de prédiction est supérieur à 70%.

Les couples login/mot de passe des utilisateurs doivent également être stockés.

L'hébergement de la base de données dans le cloud pourrait être un argument en faveur de l'intégration de la solution au site et d'autres fonctionnalités pourraient être ajoutées.

Base de données des utilisateurs :

Login	Mot de passe
alice	wonderland
bob	builder
clementine	mandarine
admin	admin

L'API

L'api sera composée de 3 parties :

Fonctions générales

Nom	Description	Accès
status	Etat de l'API `Paramètres : Non`	Tout le monde
dbconnex	Etat de la connexion à la base de données `Paramètres : Non`	Tout le monde
user	Indique le nom de l'utilisateur s'il est authentifié `Paramètres : Non`	Utilisateurs authentifiés

Prédictions

Nom	Description	Accès

predictions	Effectuer une prédiction et obtenir un ou plusieurs résultats
	`Paramètres :`
	`- File : adresse d'une image`
	`- Nb_preds : nombre de résultats attendu`
		Utilisateurs authentifiés

Les prédictions doivent être stockées dans la base de données pour une réutilisation du résultat si la même image est utilisée et les images ayant une précision > 70% doivent être enregistrées dans la base pour servir au réentrainement du modèle

Fonctions de supervision

Nom	Description	Accès
accuracy	Evalue la précision du modèle `Paramètres : Non`	Administrateur

past_pred_acc	Affiche la moyenne de précision des X dernières prédictions
	`Paramètres :`
	`- Nb_last_preds : nombre de prédictions à prendre en compte`
		Administrateur

nb_new_img	Affiche le nombre d'image ajoutées à la DBB depuis la mise en production du modèle
	`Paramètres :`
	`- Model_name : nom du modèle`
	`- Stage : stage du modèle`
		Administrateur

finetune	Effectuer une optimisation des hyperparamètres
	`Paramètres :`
	`- Model_name : nom du modèle`
	`- Stage : stage du modèle`
	`- Variables : nombre de couches du modèle à entrainer`
	`- Epochs : nombre d'époque de l'entrainement`
		Administrateur

Test et monitoring

Test unitaires à mettre en œuvre

Pour la base de données :

- Tester la connexion à la base de données

Pour les prédictions :

- Tester le score de prédiction d'une image physique
- Tester le score de prédiction d'une image via son URL
- Vérifier que la tentative de prédiction d'un format autre qu'une image renvoie une exception
- Vérifier que la tentative de prédiction d'une image non existante renvoie une exception
- Vérifier qu'une exception est levée si le nombre de résultat à obtenir n'est pas un entier

Pour l'API :

- Vérifier que les différents points de terminaison renvoi un code HTTP 200 et fonctionnent comme attendu en fonction du niveau d'accès. **Sauf finetune qui ne bénéficie pas d'un mode de test**.

Monitoring

L'application étant encore au stade "semi-expérimentale", il serait pratique de pouvoir choisir assez simplement (pour un Data Scientist) quel modèle utiliser entre les différentes expérimentations et le mettre à disposition des utilisateurs de manière transparente.

Nous devons nous assurer périodiquement ou manuellement que, sur un jeu d'évaluation, la précision du modèle ne descend pas sous 70%.

Sur la même fréquence, nous devons également être alertés si 500 nouvelles images ont été ajoutées à la base depuis la mise en production du modèle. Le nouveau jeu de données sera composé des 1.000 images les plus récentes.

Schéma d'implémentation