Rapport de Sciences des Données et Apprentissage

Adrien JAYAT Etienne DE POIX Etienne LAMOLE

12 janvier 2025

Résumé

Ce rapport présente une analyse de données sur la classification des cultures agricoles à l'aide d'images satellites, dans le cadre du projet ClassiField. Trois approches principales, les forêts d'arbres décisionnels, les réseaux de neurones entièrement connectés et des réseaux de neurones convolutifs (CNN), sont explorées. Les résultats sont évalués en termes de précision, à l'aide de métriques telles que la précision et la matrice de confusion. Nous concluons avec une discussion sur les performances et les perspectives d'amélioration.

Table des matières

Introduction

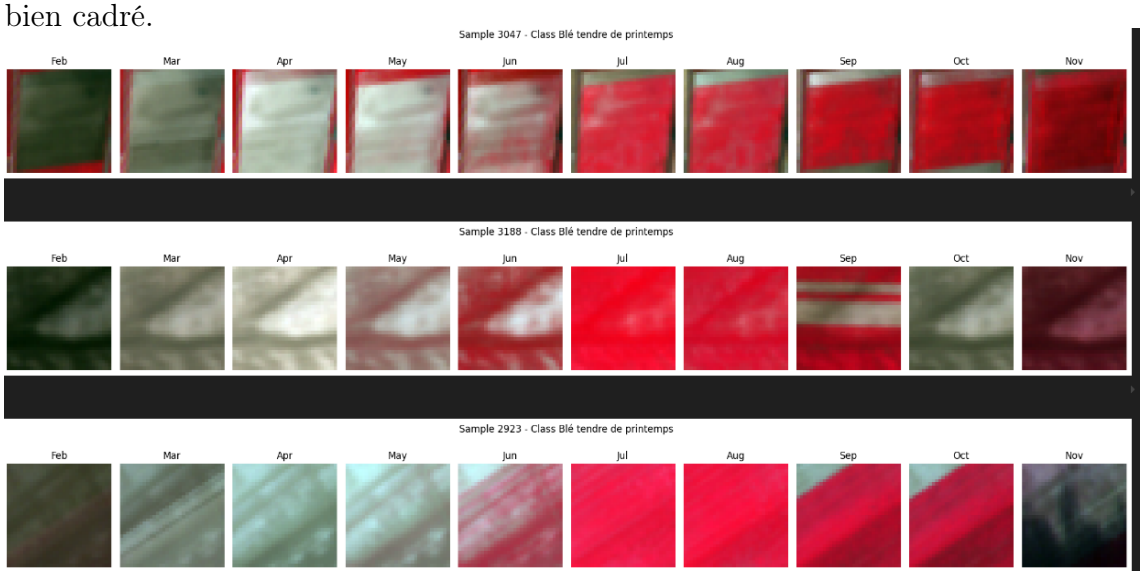
ClassiField est un projet visant à classifier les champs agricoles à partir d'images satellites. Chaque champ est représenté par une série d'images prises mensuellement entre février et novembre. L'objectif principal est de développer des modèles robustes capables de gérer des ensembles de données déséquilibrés pour une classification précise. Cette section introduit le contexte et les motivations de l'étude, tout en précisant les défis liés à l'analyse des données.

Méthodologie

2.1 Dataset

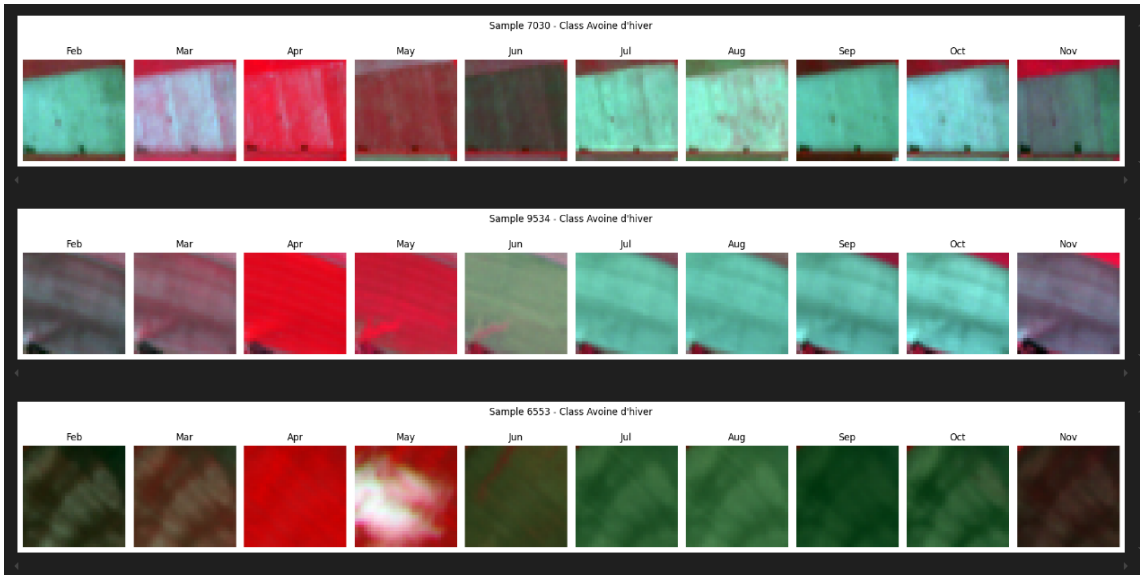
2.1.1 Description des données

Les données sont un ensemble d'images de champs, tous classés selon différentes variétés (avoine, blé, colza, maïs). Ces images sont de petite taille, 32 x 32 pixels, et paraissent donc floues. Elles sont centrées sur un champ, qui est en général plutôt bien cadré.

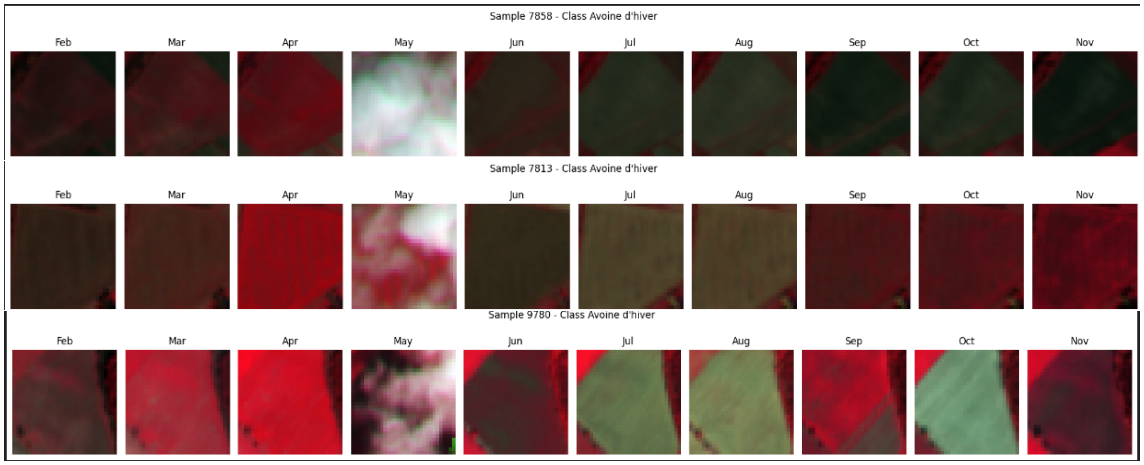


2.1.2 Analyse des données

La majorité des données est de qualité correcte pour la prédiction. Prenons l'exemple de l'avoine d'hiver. On voit que le champ est globalement rouge en avril et en mai, puis noir en juin et enfin, bleu-vert sur le reste de l'année. Ces informations sont donc exploitables pour classifier les champs. Ce motif est différent pour le champ de blé vu précédemment, où c'était en juillet, août et septembre que la couleur était rouge.

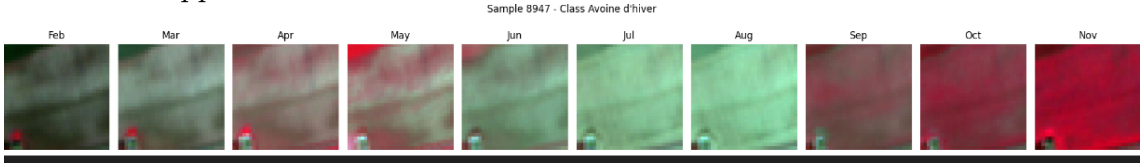


Cependant, tous les échantillons ne vont pas dans ce sens. En effet, pour une partie non négligeable, on peut observer la présence de nuages. Et ceux-ci semblent affecter le dataset d'une manière assez particulière.



On remarque premièrement que le nuage cache un mois du dataset. C'est assez rare qu'il y en ait plus. Cela peut affecter l'apprentissage, mais ce n'est pas si dérangeant, considérant la quantité de données. Ce manque peut être appris. Le plus impactant, en réalité, est le fait que les luminosités de toutes les autres photos semblent très réduites. Cela constitue une grande perte d'informations, qui est difficile à retrouver. Ces photos-là divergent grandement du dataset de base, et compliqueront nécessairement l'apprentissage, surtout qu'il y en a beaucoup (5 à 10%).

Enfin, il y a aussi des données moins courantes, mais qui divergent de la moyenne sans raison apparente.



Par exemple, cette donnée est classée en avoine d'hiver, mais la majorité des images rouges arrivent en septembre, octobre et novembre. Ces données assez rares (autour de 5%) pourraient originer soit d'une mauvaise classification, soit d'un cas particulier (par exemple, il y a une maladie sur la récolte, et le champ est labouré avant la fin de l'année et transformé soit en friche, soit en un autre champ).

2.1.3 Préparation des données

Les données sont extraites de fichiers numpy contenant les images et les labels associés. Ces données comprennent une variété de classes représentant différentes cultures, telles que l'avoine (hiver et printemps), le blé, le colza et le maïs.

Cette phase inclut des étapes de prétraitement des données, comme la normalisation des valeurs des pixels pour optimiser l'apprentissage des modèles.

2.2 Modèles de classification

2.2.1 Réseau de neurone dense

Une première approche plus simple d'un point de vue algorithmique consiste à simplifier le jeu de données en ne considérant que la couleur prépondérante du champ. Celle-ci peut être calculée via une moyenne pondérée par filtre gaussien appliqué au centre de l'image. La série temporelle est ainsi réduite à 10 valeurs de dimension 3, que nous considérerons comme 30 valeurs d'entrée. Après plusieurs essais, nous avons fait le choix d'utiliser une couche intermédiaire de 25 neurones, pour terminer avec une sortie de 20 valeurs (correspondant aux 20 types de champs). Après expérimentation de différentes méthodes, la fonction d'activation "leaky ReLU" et la méthode d'optimisation "Adam" ont été retenues. Bien que cette méthode reste très naïve, des résultats corrects pourront être observés en un temps d'apprentissage de l'ordre de la seconde.

2.2.2 Random Forest Classifier

Les forêts d'arbres décisionnels (Random Forest Classifier) constituent un type d'algorithmes d'ensembles qui crée plusieurs arbres de décision indépendants et combine leurs prédictions pour améliorer la précision. Cet algorithme est robuste face aux données bruitées et résistant au surapprentissage. Dans ce projet, nous avons utilisé une recherche d'hyperparamètres à l'aide de GridSearchCV et une validation croisée avec Stratified K-Fold.

GridSearchCV : Cette méthode permet de tester systématiquement plusieurs combinaisons d'hyperparamètres définis dans une grille. Cela garantit que nous trouvons les paramètres optimaux pour maximiser les performances du modèle.

Stratified K-Fold : Dans cette validation croisée, le jeu de données est divisé en k sous-ensembles tout en conservant la distribution des classes dans chaque sous-ensemble. Cela est particulièrement utile pour les jeux de données déséquilibrés.

Listing 2.1 – GridSearchCV et K-Fold

```python
from sklearn.model_selection import GridSearchCV, StratifiedKFold

param_grid = {
    "n_estimators": [80],
    "min_samples_leaf": [20],
}

skfold = StratifiedKFold(n_splits=5, shuffle=True, random_state=0)

# Recherche par grille avec validation croisee
grid_search = GridSearchCV(
    random_forest,
    param_grid,
    cv=skfold,
    scoring="balanced_accuracy",
    verbose=3,
)

grid_search.fit(X_train_flattened, Y_train)

best_rf = grid_search.best_estimator_
print(f"Best Parameters: {grid_search.best_params_}")
```

2.2.3 Convolutional Neural Network

Les réseaux convolutifs (CNN) sont conçus spécifiquement pour traiter les données d'images. Ce projet utilise des couches convolutionnelles 3D pour analyser les séries temporelles d'images. Voici l'architecture utilisée :

Listing 2.2 – Architecture du CNN 3D

```python
# Construction du modele CNN 3D
model = Sequential([
    Conv3D(32, kernel_size=(3, 3, 3), activation='relu',
       input_shape=input_shape),
    MaxPooling3D(pool_size=(2, 2, 2)),
    Dropout(0.25),

    Conv3D(64, kernel_size=(3, 3, 3), activation='relu'),
    MaxPooling3D(pool_size=(2, 2, 2)),
    Dropout(0.25),

    Flatten(),
    Dense(256, activation='relu'),
    Dropout(0.5),
    Dense(num_classes, activation='softmax')
])

# Compiler le modele
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

2.3 Validation et métriques

Pour évaluer les performances des modèles, plusieurs métriques ont été utilisées :

— **Matrice de confusion** : Elle permet d'analyser les prédictions par classe et de détecter les classes mal prédites.

— **Balanced Accuracy** : Celle-ci prend en compte les classes minoritaires pour fournir une mesure équilibrée des performances.

— **Accuracy globale** : Celle-là indique la proportion totale des prédictions correctes.

Résultats

3.1 Performances du réseau dense

La première méthode par simplification du jeu de données et analyse via un réseau de type "fully connected" est de loin la plus rapide et converge en moins de 3 secondes. Cette dernière octroie des résultats en balanced accuracy de 33%. Notons que cette métrique de "précision pondérée" compense le sous-effectif des classes sous-représentées dans le calcul de la moyenne de classifications correctes. Cependant, à cause de l'application du filtre de moyenne, le réseau reste sensible aux images endommagées par le passage éventuel de nuages, par exemple.

3.2 Performances du Random Forest Classifier

Après une recherche d'environ une minute, la méthode via des forêts d'arbres décisionnels a atteint une balanced accuracy de 50%, avec une performance régulière pour les classes majoritaires, mais montre des difficultés pour les classes sous-représentées. Les résultats indiquent que cet algorithme est rapide et simple à implémenter, mais qu'il atteint ses limites pour capturer les nuances des données complexes.

3.3 Performances du CNN

Le modèle CNN a obtenu une balanced accuracy de 65%. Ses points forts incluent une meilleure précision pour les classes minoritaires et une capacité à détecter les motifs complexes dans les images. Toutefois, cette approche exige davantage de ressources computationnelles et de temps d'entraînement.

3.4 Analyse comparative

La comparaison entre ces approches met en lumière que le Random Forest est mieux adapté pour une implémentation rapide avec des ressources limitées, tandis que le CNN excelle dans la classification précise des données d'images, au prix d'un coût en temps et en calcul.

Conclusion

Ce rapport met en évidence les avantages et limitations des deux approches explorées. Alors que les forêts d'arbres décisionnels offrent une solution efficace pour une première exploration des données, le CNN s'impose pour des besoins de précision élevée et une analyse approfondie des classes complexes. Les perspectives incluent l'utilisation d'augmentation de données pour enrichir l'ensemble d'apprentissage et la mise en place de modèles hybrides combinant les points forts qu'offrent ces deux approches.