Science du Numérique

S9 - Apprentissage Machine

Classification des couvertures de terrains

Rapport

Auteurs :
Nicolas CATONI
Bryan CHEN
Kawtar LYAMOUDI
Jordan RAMASSAMY-MOUTOUSSAMY

29 décembre 2023

Institut National Polytechnique de Toulouse - ENSEEIHT | Science du Numérique
Calcul Haute Performance et Données Massives
2023-2024

Table des matières

Table des figures

1 Introduction

La mission actuelle implique la classification des types de couverture terrestre, en se basant sur des images satellites qui composent dix classes : AnnualCrop, Forest, HerbaceousVegetation, Highway, Industrial, Pasture, PermanentCrop, Residential, River et SeaLake.

Le jeu de données se compose d'images étiquetées avec leurs types de couverture terrestre respectifs. L'objectif principal est de développer un modèle d'apprentissage automatique capable de prédire de manière précise le type de couverture terrestre pour de nouvelles images non encore observées.

Pour ce faire, des modèles de Deep Learning ont été utilisés. 20 000 images sont à notre disposition, dont 80% pour l'entraînement et 20% pour la validation.

Ce projet peut présenter plusieurs défis qui nécessiteront une attention particulière pour atteindre des résultats satisfaisants. Voici quelques-uns des challenges potentiels associés à la classification des types de couverture terrestre basée sur des images satellites :

- **Similitude entre les Images :** Un défi notable réside dans la similitude visuelle entre certaines classes de couverture terrestre. Par exemple, différentes catégories telles que 'River' et 'Highway' peuvent présenter des caractéristiques visuelles similaires dans les images satellites. Distinguer de telles similitudes nécessitera des techniques de modélisation robustes.
- **Déséquilibre de Classes :** Il est fréquent d'observer un déséquilibre dans la répartition des classes, où certaines catégories de couverture terrestre peuvent être sous-représentées par rapport à d'autres. Ceci peut entraîner un biais dans le modèle, nécessitant des stratégies d'équilibrage des classes lors de l'entraînement.

Il est important de noter que ce projet a également quelques limitations : sa performance peut être influencée par la diversité et la qualité des images dans le "training set", ainsi que par les techniques utilisées et les paramètres choisis. Ces facteurs doivent être pris en compte pour garantir des résultats satisfaisants.

2 Sélection du modèle

Pour le premier modèle, nous avons utilisé du transfert learning, en nous inspirant des pratiques courantes utilisées pour le MNIST. En ce qui concerne le modèle, des tentatives ont été faites pour ajuster les hyperparamètres tels que le Weight Decay et le taux d'apprentissage, dans le but d'optimiser les résultats et d'atteindre des performances supérieures. Le premier modèle est composé ainsi de :

```
SimpleCNN(
  (network): Sequential(
    (0): Conv2d(3, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (3): ReLU()
    (4): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (5): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (6): ReLU()
    (7): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (8): ReLU()
    (9): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (10): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (11): ReLU()
    (12): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (13): ReLU()
    (14): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (15): Flatten(start_dim=1, end_dim=-1)
    (16): Linear(in_features=16384, out_features=1024, bias=True)
    (17): ReLU()
    (18): Linear(in_features=1024, out_features=512, bias=True)
    (19): ReLU()
    (20): Linear(in_features=512, out_features=64, bias=True)
  )
  (fc): Linear(in_features=64, out_features=10, bias=True)
)
```

FIGURE 1 – Composition du premier modèle

Nous avons également obtenu la précision et le coût suivants, atteignant ainsi jusqu'à 85% d'accuracy sur l'ensemble d'entraînement.

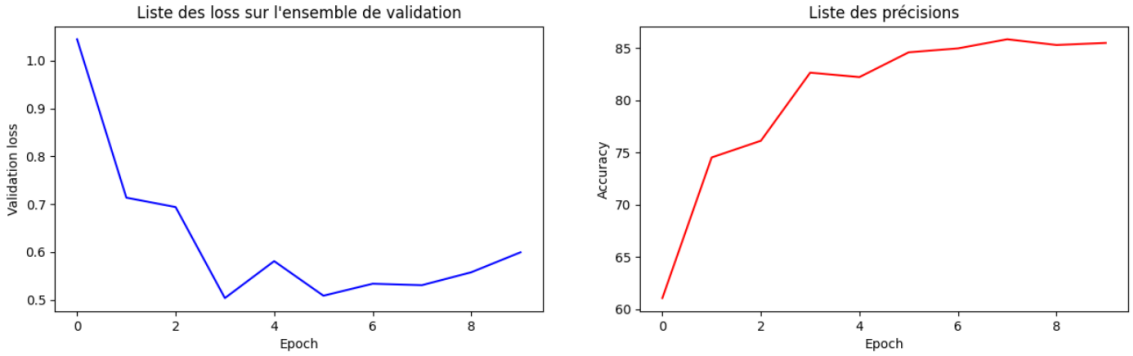

FIGURE 2 – Précision et coût du premier modèle

En passant ce code sur l'ensemble de test, nous avons obtenu une accuracy de 84%.

Nous avons entrepris une série d'essais et d'expérimentations en explorant divers modèles et configurations. Nous avons commencé par l'implémentation d'un réseau de neurones à convolution (CNN) simple, avant de progresser vers des architectures plus complexes, telles que ResNet18, ResNet50 et ResNet100. Cette démarche visait à évaluer la performance de chaque modèle et à comparer leurs capacités respectives dans la classification d'images satellites en 10 classes.

Par exemple, le modèle ResNet18 qui dispose de moins de paramètres entraînables que le modèle ResNet50 apprend sensiblement plus vite que ce dernier et atteint une justesse qui frôle les 92%. Ce qui est malheureusement moins que le score atteint par le modèle ResNet50 (96%)

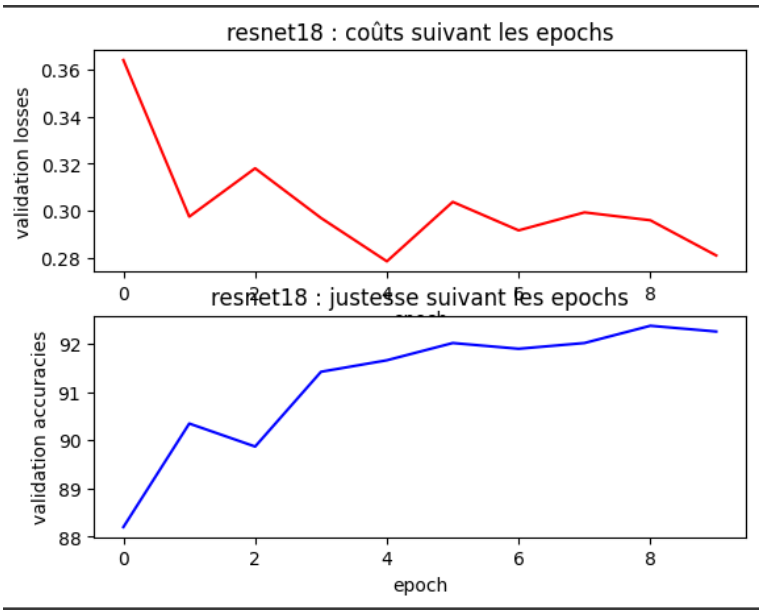

FIGURE 3 – Justesse et coût du modèle ResNet18 (validation

Ainsi le modèle retenu est celui du **ResNet50**, à l'exception près où la couche full-connected est remplacée par une autre donnant le bon nombre de classes en sortie (ici, 10). Ce modèle est pré-entraîné avec les poids de **IMAGENETEK_V2** et on décide de geler toutes les couches sauf la dernière (la couche 4).

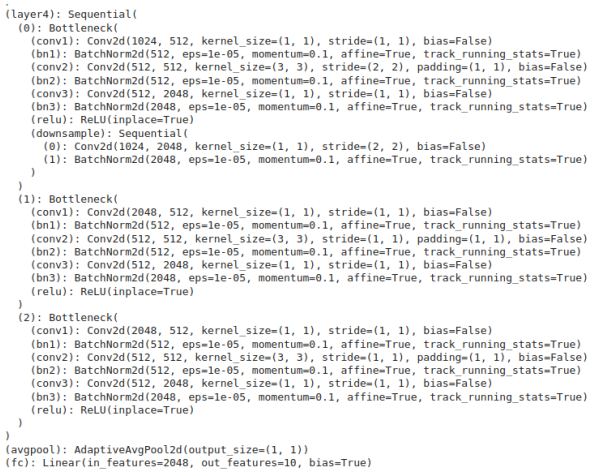

FIGURE 4 – Visualisation de la couche 4 et la couche fully-connected

Nous avons obtenu la précision et le coût suivants, atteignant ainsi jusqu'à 99.6% d'accuracy sur l'ensemble d'entraînement. En passant ce code sur l'ensemble de test, nous avons obtenu une accuracy de 96%.

3 Préparation des données

Les images dans notre ensemble de données sont de dimensions $64 \times 64 \times 3$, indiquant une résolution de 64 pixels en hauteur et en largeur, avec trois canaux de couleur (images RGB : rouge, vert, bleu). Les labels associés à chaque image sont des vecteurs binaires de taille 10, adoptant la forme suivante :

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$$

Ici, ce vecteur particulier sert de label pour la classe 3 parmi les 10 classes disponibles. Ce schéma de représentation est connu sous le nom de "one-hot encoding," où une seule composante du vecteur est activée pour indiquer la classe correspondante.

Afin d'améliorer la diversité de nos données d'entraînement et d'éviter le surajustement du modèle, nous avons mis en œuvre une augmentation de données. Cela inclut des opérations telles que la rotation, la symétrie, et la normalisation des images. La rotation et la symétrie permettent de présenter au modèle des perspectives variées de chaque classe, tandis que la normalisation assure une cohérence dans l'échelle des couleurs.

En plus de ces techniques, nous avons exploré d'autres stratégies d'augmentation telles que la modification de la couleur de l'image, le zoom, le rognage, et la translation. Cependant, après des expérimentations approfondies, nous avons constaté que ces transformations n'apportent pas d'améliorations significatives en termes de précision du modèle. Certaines transformations, comme le découpage de sous image, entraîne même une perte de précision. Ceci s'explique par le fait que ces transformations changent la nature de l'image. En conséquence, nous avons opté pour une sélection judicieuse des techniques d'augmentation qui ont démontré leur efficacité dans notre contexte spécifique.

4 Entraînement et paramètres

Pour le processus d'entraînement, nous avons utilisé un optimiseur stochastique avec la méthode **AMSGrad**. Le taux d'apprentissage (*learning rate*) a été fixé à $lr = 0.0001$, le terme de régularisation (*weight decay*) à $weight\_decay = 0.0001$, et la taille des lots (*batch size*) à 64. Le nombre d'époques

a été défini à 23. La fonction de coût utilisée est la fonction **CrossEntropy** de **Torch**.

Au cours de l'entraînement, nous avons observé une précision d'entraînement d'environ 99% et une perte (*loss*) d'environ 0.01. Ces indicateurs témoignent de la performance élevée de notre modèle sur l'ensemble d'entraînement. Ces paramètres ont été sélectionnés après des expérimentations approfondies visant à trouver un équilibre entre la convergence du modèle et la prévention du surajustement (*overfitting*).

Si la précision ne s'améliorait pas au cours du processus d'entraînement, nous avons mis en œuvre une technique d'arrêt anticipé (*early stopping*). Nous avons défini le nombre d'epoch à 75, bien que ce nombre ne soit pas toujours atteint en raison du déploiement de l'arrêt anticipé. Cette stratégie vise à éviter le surajustement en arrêtant l'entraînement dès que la performance du modèle sur un ensemble de validation cesse de s'améliorer.

5 Evaluation de modèle

Lors de la phase de validation, qui a été réalisée sur 20% de la base de données, nous avons évalué les performances de notre modèle en utilisant plusieurs métriques significatives telles que la justesse (*accuracy*) et la perte (*loss*). La justesse obtenue était d'environ 96%, indiquant un niveau élevé de précision dans les prédictions. La perte associée était d'environ 0.1, soulignant une bonne convergence du modèle pendant la phase de validation.

Pour une analyse plus détaillée, nous avons généré une matrice de confusion. Cette matrice nous permet d'analyser en détail les résultats de classification, identifiant notamment les faux positifs, les faux négatifs, les vrais positifs et les vrais négatifs. En examinant ces résultats, nous avons pu déduire les faiblesses spécifiques du modèle et orienter nos efforts d'amélioration.

Par ailleurs, nous avons utilisé des métriques de performance telles que la précision (*precision*), le rappel (*recall*), le F1-score, et enfin la moyenne (*average*) pour l'ensemble des classes. La précision mesure l'exactitude des prédictions positives, le rappel évalue la capacité du modèle à identifier toutes les instances positives, et le F1-score offre une balance entre ces deux métriques. Ces évaluations détaillées ont contribué à une compréhension approfondie des performances de notre modèle dans la classification des types de couverture terrestre. Le tableau suivant montre les métriques de perfomrances trouvés pour l'architecture basée sur ResNet50 :

Class	Precision	Recall	F1-Score	Support
AnnualCrop	0.97	0.98	0.97	33281
Forest	0.98	0.98	0.98	32660
HerbaceousVegetation	0.96	0.97	0.96	33626
Highway	0.96	0.97	0.97	27255
Industrial	0.98	0.98	0.98	27347
Pasture	0.96	0.96	0.96	21206
PermanentCrop	0.97	0.95	0.96	27784
Residential	0.98	0.98	0.98	32108
River	0.97	0.96	0.96	26979
SeaLake	0.99	0.98	0.98	32154
Macro Avg	0.97	0.97	0.97	294400
Weighted Avg	0.97	0.97	0.97	294400

TABLE 1 – Classification Report pour le ResNet50

6 Analyse des résultats

Bien évidemment, les scores obtenus lors de l'étape de validation sont moins probants du fait que les poids du modèle ne sont pas mis à jour durant cette phase. En conséquence, le score de justesse plafonne à 96%.

En examinant attentivement la matrice de confusion, nous pouvons identifier les classes bien classées ainsi que celles présentant des défis pour le modèle. Globalement, l'erreur de classification est particulièrement faible, soulignant la robustesse globale de notre modèle.

Cependant, une observation intéressante émerge de la matrice de confusion. Nous remarquons une tendance du modèle à confondre quelques classes, telles que "Rivers" et "Highways", ainsi que les deux classes "Herbaceous Vegetation" et "Permanent Crop". Cette confusion est compréhensible du fait des similitudes visuelles entre ces classes d'un point de vue satellitaire. Cela suggère que le modèle peut être sensible à des caractéristiques spécifiques partagées entre les deux classes.

Pour améliorer ces résultats, des stratégies ciblées pourraient inclure une augmentation de données spécifique à ces classes, ou l'exploration de techniques avancées d'extraction de caractéristiques pour mieux distinguer les différences subtiles. Cette analyse approfondie des résultats nous permet de cibler des domaines spécifiques pour des améliorations futures, contribuant ainsi à renforcer la performance et la précision globale du modèle.

7 Visualisation des résultats

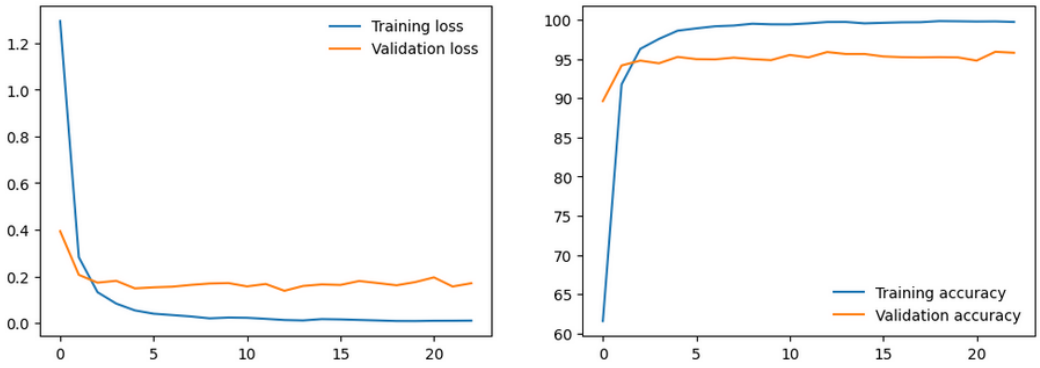

FIGURE 5 – Justesse et le coût des étapes d'entraînement et de validation

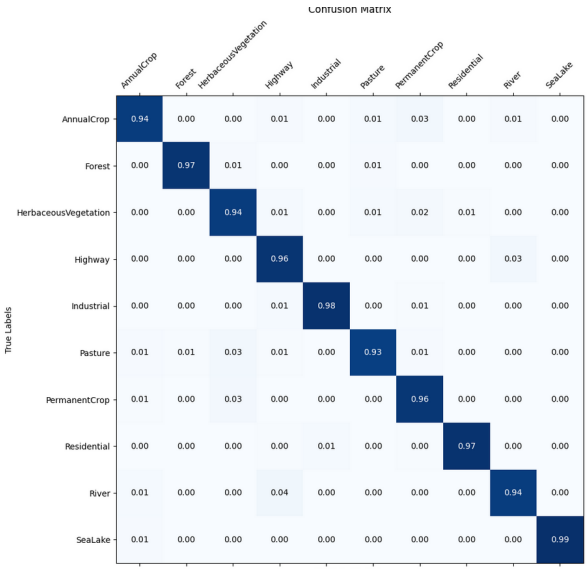

FIGURE 6 – Matrice de confusion

8 Conclusion

En résumé, le modèle proposé a démontré d'excellentes performances globales, avec une justesse d'environ 96%. Cependant, une analyse approfondie de la matrice de confusion a révélé une certaine difficulté du modèle à différencier correctement entre les rivières et les autoroutes. Cette observation suggère une opportunité d'amélioration spécifique.

Un angle d'amélioration potentiel pour remédier à ce problème serait d'envisager un deuxième réseau de neurones spécialisé dans la différenciation entre les rivières et les autoroutes. Ceci permettrait au modèle d'apprendre des caractéristiques spécifiques à ces deux classes, améliorant ainsi la capacité de discrimination.

Bien que plusieurs modèles aient été essayés, tout autant que les différents méthodes de descente de gradient, il a été constaté qu'il était relativement difficile de franchir le plafond de performance (ici 96 %). Malgré les différentes tentatives de modifier le taux d'apprentissage et le weight decay, la justesse a oscillé au mieux autour de 95 %. Or il est judicieux de trouver les bonnes valeurs pour ces hyperparamètres afin de profiter au mieux des performances du modèle sélectionné. Par conséquent, il pourrait être intéressant d'utiliser des méthodes de machine learning telles que l'analyse bayésienne pour trouver ces valeurs optimales. Ainsi on chercherait à maximiser la densité de probabilité liée à la performance du modèle en sélectionnant les bons hyperparamètres. Par ailleurs, une autre approche serait de combiner un modèle EfficientNet, un ResNet50, suivi d'un modèle de deep learning entièrement connecté, cela aurait pu apporter une diversité et une complémentarité significatives à nos résultats. Chaque modèle aurait apporté sa propre perspective et ses capacités distinctes à la tâche de classification de la couverture terrestre basée sur des images satellites.

L'utilisation d'EfficientNet aurait introduit une architecture efficace en termes de paramètres. Son aptitude à gérer efficacement des ressources computationnelles aurait pu être avantageuse, surtout pour la classification de classes présentant des similitudes visuelles. Le choix de ResNet50, pré-entraîné sur ImageNet, aurait tiré parti de la capacité de ce modèle à extraire des caractéristiques complexes. Cette architecture aurait pu capturer des informations essentielles pour la classification des différents types de couverture terrestre, contribuant ainsi à améliorer la précision globale. L'inclusion d'un modèle de deep learning entièrement connecté aurait pu offrir une approche différente en apprenant des combinaisons linéaires de caractéristiques extraites par les couches précédentes. Cela aurait pu être

particulièrement utile pour la différenciation entre des classes présentant des caractéristiques subtiles.

Cette approche séquentielle aurait pu offrir une synergie potentielle pour résoudre les défis de classification identifiés, tels que la distinction entre les rivières et les autoroutes. Explorer cette combinaison séquentielle aurait pu constituer une voie prometteuse pour améliorer encore davantage les performances du modèle, en particulier sur les classes spécifiques présentant des difficultés de distinction. En dépit de ces défis spécifiques, les performances générales du modèle restent prometteuses. En continuant à explorer de nouvelles stratégies d'optimisation et d'augmentation de données, nous pouvons anticiper des améliorations continues et une plus grande précision dans la classification des classes spécifiques.

9 Contributions individuelles

9.1 Sélection de modèles :

Plutôt que de se limiter à un seul modèle, notre équipe a adopté une approche parallèle en testant plusieurs modèles en même temps. Nicolas a initié cette phase, explorant divers modèles pour évaluer leur pertinence par rapport à nos objectifs. Cette diversité d'approches a jeté les bases pour une sélection réfléchie du modèle final.

9.2 Répartition des responsabilités :

Chaque membre de l'équipe a assumé un rôle spécifique en fonction de ses compétences. Bryan a dirigé la décision de choisir ResNet50. Kawtar a pris en charge la préparation des données, tandis que Jordan a pris la responsabilité de détailler le processus d'entraînement. Suite à l'implémentation du ResNet50 après le premier modèle (cf. FIGURE 1), nous avons essayé d'implémenter le ResNet18, le ResNet101 ainsi que l'EfficientNet.

9.3 Expérimentation et feedback :

Nous avons maintenu une communication ouverte et régulière, partageant nos expériences, les résultats de nos expérimentations, et fournissant des feedbacks constructifs. Cela a favorisé un environnement d'apprentissage continu, où les idées étaient échangées librement, contribuant à l'amélioration collective de nos approches.

9.4 Visualisation des résultats :

Kawtar, en charge de la visualisation des résultats, a créé des graphiques clairs illustrant la performance des modèles. Ces visualisations ont été essentielles pour comprendre les tendances d'entraînement, les points forts et les faiblesses, facilitant ainsi la prise de décisions informées. Nicolas et Bryan se sont mis à travailler sur la construction des matrices de confusion, qui pourraient être une mine d'or d'informations afin de distinguer quels sont les points épineux du projet.

9.5 Réunion régulière pour la conclusion :

Notre équipe s'est réunie régulièrement pour discuter des progrès, des défis et des conclusions. Lors de la rédaction de la conclusion, Jordan a consolidé ces discussions, résumant de manière perspicace les principales conclusions et proposant des pistes pour des améliorations futures.

En travaillant de manière collaborative, en capitalisant sur les forces individuelles et en maintenant une approche ouverte à l'expérimentation, notre équipe a réussi à créer un projet de machine learning robuste et informatif.