Projet : Classification de Scènes Naturelles

Jeu de données proposé : Intel Image Classification sur Kaggle

Ce jeu de données contient environ 25 000 images réparties en 6 catégories de scènes naturelles : 'buildings' (bâtiments), 'forest' (forêt), 'glacier', 'mountain' (montagne), 'sea' (mer), et 'street' (rue).

Plan de Projet Détaillé

1. Introduction au Projet

Comprendre le problème de la classification d'images de scènes naturelles et son application (par exemple, dans l'organisation de photos, la robotique environnementale ou l'analyse géographique).

Étape : Introduction au dataset "Intel Image Classification" et exploration des 6 classes de scènes présentes.

2. Préparation des Données

Objectif : Préparer les images pour l'entraînement du modèle CNN.

Étapes :

- Charger et explorer le dataset.
- Prétraiter les images : les images sont déjà de taille 150x150, mais la normalisation reste cruciale.
- Diviser le dataset en ensembles d'entraînement, de validation et de test (les dossiers sont déjà structurés ainsi sur Kaggle).
- Augmentation des données : Appliquer des techniques comme la rotation, le zoom, le retournement horizontal (très pertinent ici), et les variations de luminosité pour améliorer la robustesse du modèle.

3. Conception et Implémentation du Modèle CNN

Objectif : Construire un modèle CNN pour la classification des scènes naturelles.

Étapes :

- Commencer avec une architecture CNN simple pour valider le pipeline.
- Implémenter les couches de convolution, de pooling, et `fully connected`.
- Utiliser des techniques de régularisation comme le `Dropout` pour éviter le surapprentissage.

4. Entraînement du Modèle

Objectif : Entraîner le modèle CNN avec les données préparées.

Étapes :

- Choisir une fonction de coût adaptée à la classification multi-classes, comme `CategoricalCrossentropy`.
- Utiliser un optimiseur comme `Adam`.
- Monitorer les courbes de performance (accuracy, loss) sur l'ensemble de validation pour détecter le surapprentissage.

5. Évaluation du Modèle

Objectif : Évaluer la performance finale du modèle sur le jeu de test.

Étapes :

- Calculer la précision, le rappel et le score F1.
- Générer une **matrice de confusion** pour visualiser les erreurs de classification. Par exemple, le modèle confond-il souvent 'montagne' et 'glacier' ?

6. Améliorations et Expérimentations

Objectif : Explorer des moyens d'améliorer les performances.

Étapes :

- Tester l'impact de différentes techniques d'augmentation de données.
- Ajuster les hyperparamètres (taux d'apprentissage, taille des batchs).
- **Intégrer le transfer learning :** Utiliser un modèle pré-entraîné sur ImageNet (ex: VGG16, ResNet50, MobileNet) et l'adapter à ce dataset.

7. Interprétation et Visualisation des Résultats

Objectif : Comprendre comment le modèle prend ses décisions.

Étapes :

- Utiliser des techniques comme **Grad-CAM** pour visualiser quelles parties des images sont les plus importantes pour une prédiction donnée (par exemple, le ciel pour 'mer' ou la texture des arbres pour 'forêt').
- Analyser les erreurs courantes (ex: confusion entre 'forêt' et 'montagne' si les deux contiennent beaucoup d'arbres) et proposer des solutions.

8. Code et Présentation

Objectif : Synthétiser et présenter votre travail.

Étapes :

- Fournir un code Python propre et commenté.
- Préparer une présentation de vos résultats (8 solides).