Innovation en Classification des Déchets Ménagers

Groupe : 4IABD1 : 2023/2024

Préparé par :

Dounia HARAG

Chaimae MOUH

Alain Christian NZEGAING BEHALAL

Ayoub ERRADI

Table des matières

Introduction

Dans le cadre de notre projet annuel 2024, nous avons entrepris de concevoir un site web innovant et utile pour la classification des déchets ménagers. L'objectif principal de ce projet est de faciliter le tri et le recyclage des déchets quotidiens tels que l'aluminium, le carton, le verre, le plastique, les déchets organiques et les déchets électroniques. En combinant les avancées technologiques de l'apprentissage automatique et de l'intelligence artificielle, nous avons développé un modèle capable de reconnaître et de catégoriser automatiquement ces types de déchets à partir d'images.

Présentation du Projet

Notre projet se compose de plusieurs phases clés, chacune jouant un rôle essentiel dans l'accomplissement de notre objectif. Voici un aperçu des différentes étapes que nous avons suivies :

1. **Collecte et Préparation des Données** : La première étape de notre projet a consisté à collecter un ensemble diversifié et représentatif d'images de déchets. Nous avons utilisé diverses sources pour compiler un dataset complet comprenant des images d'aluminium, de carton, de verre, de plastique, de déchets organiques et de déchets électroniques. Les images ont ensuite été prétraitées et annotées pour créer un dataset prêt à être utilisé pour l'apprentissage supervisé.

2. **Développement du Modèle MobileNetV2** : Nous avons implémenté le modèle MobileNetV2, réputé pour son efficacité en classification d'images, afin de reconnaître les différents types de déchets. Cette phase inclut l'entraînement du modèle avec notre dataset, l'optimisation des hyperparamètres et l'évaluation des performances du modèle. Nous avons également utilisé des techniques de validation croisée pour assurer la robustesse et la précision du modèle.

3. **Création du Site Web** : Pour rendre notre solution accessible et pratique, nous avons développé un site web utilisant Django, un framework web puissant et flexible. Ce site permet aux utilisateurs de télécharger des photos de leurs déchets et de recevoir instantanément une classification ainsi que des conseils sur le recyclage approprié. L'interface utilisateur a été conçue pour être intuitive et conviviale, facilitant ainsi l'utilisation du site par un large public.

4. **Déploiement sur AWS** : La phase finale de notre projet a consisté à déployer le site web sur une infrastructure cloud robuste et évolutive, en utilisant les services d'Amazon Web Services (AWS). Nous avons configuré une instance EC2 pour héberger notre site et assuré une gestion efficace des ressources. L'utilisation d'AWS nous a permis de garantir une haute disponibilité, une sécurité renforcée et une scalabilité adaptée aux besoins du site.

De ce fait, nous répondrons à des problématiques bien définies :

- Quelle a été notre approche pour la réalisation de ce projet annuel ?
- Quels ont été les problèmes majeurs pour la réalisation du projet ?
- Quel est notre avis personnel sur les résultats obtenus et qu'aurait-on pu faire mieux durant ce travail ?

DataSet

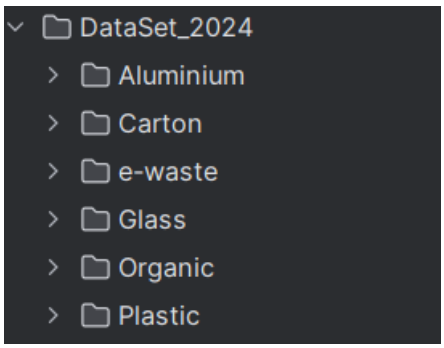
La première étape pour la réalisation de ce projet ambitieux était l'acquisition d'un dataset, sachant que nous avions besoin d'un dataset très conséquent pour la réalisation du projet.

Une dataset de grande taille et diversifiée était essentielle pour permettre une généralisation efficace et obtenir des résultats précis dans la classification des déchets ménagers.

Nous avons créé notre dataset à l'aide de plusieurs datasets disponibles sur **Kaggle**. En combinant ces différentes sources, nous avons pu assembler une collection complète et diversifiée d'images représentant du **carton**, du **verre**, de **l'aluminium**, des **déchets organiques**, du **plastique** et des **déchets d'équipements électriques et électroniques** (DEEE).

Une fois les images collectées, nous avons classé les photos sous six dossiers distincts, chacun représentant une catégorie de déchets. Chaque dossier contient environ 4500 images. Cette organisation a permis de structurer efficacement les données et de faciliter les étapes de traitement ultérieures.

Cette acquisition et organisation des données a été une étape cruciale dans notre projet. L'expérience nous a montré l'importance de disposer d'un dataset bien dimensionné et diversifié pour atteindre nos objectifs de manière plus précise et fiable. Grâce à cette méthode, nous avons pu maximiser la qualité de notre projet tout au long de son développement



Modèles et Évaluation

Pour accomplir notre objectif de classification des déchets ménagers, nous avons exploré plusieurs architectures de modèles de réseaux de neurones convolutifs (CNN). Nous avons comparé les performances de ces modèles pour sélectionner celui offrant la meilleure précision. Voici un aperçu des modèles testés et des résultats obtenus :

Modèle CNN de Base

Le premier modèle CNN que nous avons développé était une architecture simple comprenant plusieurs couches de convolution et de pooling. Bien que ce modèle ait fourni des résultats prometteurs, avec une précision de test de 62 %, il n'était pas suffisant pour répondre à nos exigences de performance.

```
# Définir le modèle CNN adapté
model = Sequential([
    Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(64, 64, 3)),
    MaxPooling2D(pool_size=(2, 2)),

    Conv2D(64, kernel_size=(3, 3), activation='relu'),
    MaxPooling2D(pool_size=(2, 2)),

    Conv2D(128, kernel_size=(3, 3), activation='relu'),
    MaxPooling2D(pool_size=(2, 2)),

    Flatten(),
    Dense(512, activation='relu'),
    Dropout(0.5),
    Dense(6, activation='softmax')
])

# Compiler le modèle
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Entraîner le modèle
history = model.fit(datagen.flow(train_data, train_labels, batch_size=32),
                    epochs=20,
                    validation_data=(test_data, test_labels))

# Évaluer le modèle sur l'ensemble de test
test_loss, test_acc = model.evaluate(test_data, test_labels)
```

Modèle CNN Amélioré

Pour améliorer les performances, nous avons enrichi notre modèle avec des couches de normalisation par batch et des techniques de régularisation supplémentaires. Ce modèle a montré une précision améliorée de 65 % sur l'ensemble de test.

```
# Définir le modèle CNN amélioré
model = Sequential([
    Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(64, 64, 3)),
    BatchNormalization(),
    MaxPooling2D(pool_size=(2, 2)),
    Dropout(0.25),

    Conv2D(64, kernel_size=(3, 3), activation='relu'),
    BatchNormalization(),
    MaxPooling2D(pool_size=(2, 2)),
    Dropout(0.25),

    Conv2D(128, kernel_size=(3, 3), activation='relu'),
    BatchNormalization(),
    MaxPooling2D(pool_size=(2, 2)),
    Dropout(0.25),

    Flatten(),
    Dense(512, activation='relu'),
    BatchNormalization(),
    Dropout(0.5),
    Dense(6, activation='softmax')
])

# Compiler le modèle
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Définir un callback pour réduire le taux d'apprentissage
```

Modèle MobileNetV2

Enfin, nous avons implémenté MobileNetV2, un modèle pré-entraîné réputé pour son efficacité en classification d'images. En utilisant l'apprentissage par transfert, nous avons adapté MobileNetV2 pour notre tâche spécifique de classification des déchets. Ce modèle a nettement surpassé les précédents, atteignant une précision de 82 %.

```
# Charger le modèle MobileNetV2 pré-entraîné sans les couches de classification
base_model = MobileNetV2(weights='imagenet', include_top=False, input_shape=(128, 128, 3))

# Geler les couches de MobileNetV2
for layer in base_model.layers:
    layer.trainable = False

# Ajouter les couches de classification personnalisées
model = Sequential([
    base_model,
    Flatten(),
    Dense(512, activation='relu'),
    Dropout(0.5),
    Dense(6, activation='softmax')
])

# Compiler le modèle
model.compile(optimizer=Adam(learning_rate=0.001), loss='categorical_crossentropy', metrics=['accuracy'])

# Entraîner le modèle
history = model.fit(datagen.flow(train_data, train_labels, batch_size=32),
                    epochs=20,
                    validation_data=(test_data, test_labels))
```

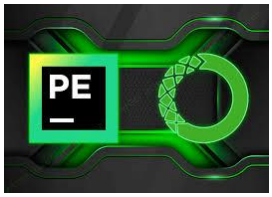
Justification du Choix du Modèle

Après avoir comparé les performances des différents modèles, nous avons constaté que MobileNetV2 offrait une précision de 82 %, bien supérieure à celle des autres modèles testés. Cette performance accrue, combinée à l'efficacité de MobileNetV2 dans la classification d'images, en a fait le choix idéal pour notre projet. L'apprentissage par transfert nous a permis de tirer parti des connaissances acquises par MobileNetV2 sur des millions d'images, améliorant ainsi la précision et la robustesse de notre modèle pour la tâche spécifique de classification des déchets ménagers.

Développement de l'Application Web

Nous avons réalisé une application web en utilisant **Python** et le framework **Django**. Cette application permet d'importer des images de déchets ménagers (carton, verre, aluminium, déchets organiques, plastique, et déchets d'équipements électriques et électroniques) puis de les soumettre à notre système de machine learning avancé.

L'application est capable de reconnaître et de classifier les différents types de déchets.



Fonctionnalités de l'Application

L’application web permet aux utilisateurs de télécharger des images de déchets ménagers. Une fois les images soumises, elles sont traitées par notre algorithme de machine learning, qui détermine instantanément la catégorie à laquelle chaque déchet appartient. Les catégories incluent :

- Carton
- Verre
- Aluminium
- Déchets organiques
- Plastique
- Déchets d'équipements électriques et électroniques (DEEE)

Configuration Technique

Pour le développement de notre application, nous avons utilisé les outils et configurations suivants :

- **Environnement de Développement** :
  - **PyCharm** : Nous avons utilisé l'IDE PyCharm pour le développement de notre application. PyCharm offre des outils robustes pour le développement Python, facilitant l'écriture, le test et le débogage de notre code.
  - **Miniconda** : Pour gérer les dépendances et les environnements virtuels, nous avons utilisé Miniconda. Cela nous a permis d'isoler les bibliothèques nécessaires et d'assurer la compatibilité des versions.
- **Langage de Programmation** :
  - **Python** : Python a été notre langage de programmation principal. Sa syntaxe simple et lisible a facilité le développement rapide d'applications robustes.
- **Framework Web** :
  - **Django** : Nous avons choisi Django pour créer notre application web. Django, basé sur le modèle MVC (Modèle-Vue-Contrôleur), simplifie la gestion de la logique métier, des modèles de données et des vues. Il offre également des fonctionnalités de sécurité intégrées pour protéger les données sensibles.

Avantages de l'Application

- *Facilité d'utilisation* : L'interface utilisateur est conçue pour être intuitive, permettant à toute personne de télécharger des images de déchets et d'obtenir une classification instantanée.

- *Précision* : Grâce à notre système de machine learning sophistiqué, l'application offre une reconnaissance précise des types de déchets, améliorant ainsi l'efficacité du tri et du recyclage.

- *Sécurité* : Django fournit des fonctionnalités de sécurité intégrées, assurant la protection des données téléchargées par les utilisateurs

Utilisation de Git pour la Gestion du Code

Initialisation du Dépôt Git

- Nous avons initialisé un dépôt Git dans notre répertoire de projet local pour suivre les versions du code source.

Ajout des Fichiers au Dépôt

- Nous avons ajouté les fichiers du projet au dépôt et effectué un premier commit pour enregistrer l'état initial du projet.

Configuration du Dépôt Distant

- Nous avons ajouté l'URL du dépôt distant https://github.com/chaimaemouh/Projet_Annuel_M1_2024.git pour permettre la synchronisation de notre code source.

Gestion des Fichiers Volumineux avec Git LFS

Pour gérer les fichiers volumineux, tels que le modèle `mobilenetv2_model.h5`, nous avons utilisé Git LFS (Large File Storage). Cela permet de suivre et de stocker efficacement les fichiers de grande taille sans affecter les performances du dépôt Git.

Voici quelques commandes qu'on a utilisé :

git lfs install

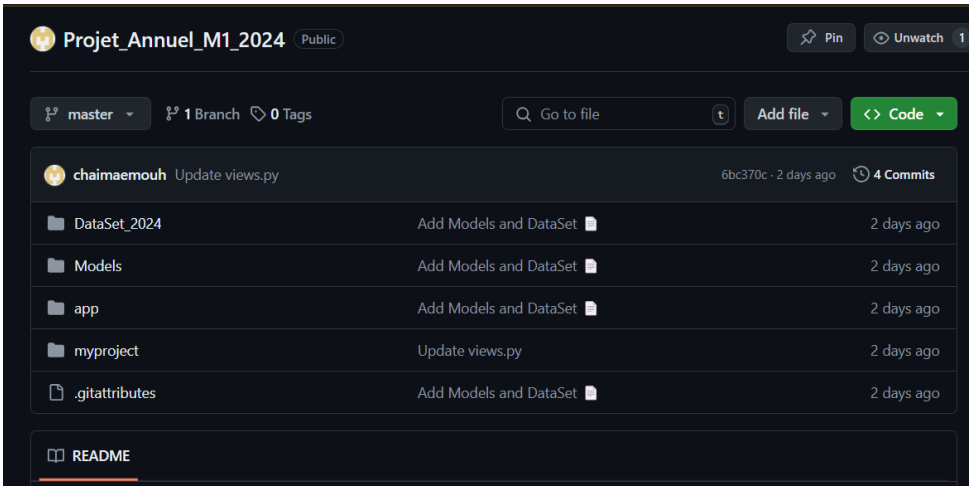
git lfs track "*.h5"

git add .gitattributes

git add mobilenetv2_model.h5

git commit -m "Ajout du modèle MobileNet avec Git LFS"

git push origin master



Déploiement sur AWS EC2

Pour héberger notre application web et gérer les ressources nécessaires, nous avon utilisé les services d'Amazon Web Services (AWS). En particulier, nous avons créé une instance EC2 pour assurer le déploiement et le fonctionnement optimal de notre application.

Configuration Technique d'AWS

Nous avons créé une instance EC2 pour héberger notre application web. EC2 (Elastic Compute Cloud) nous a permis de lancer des serveurs virtuels selon nos besoins spécifiques.

- Scalabilité : Ajustement de la capacité du serveur selon la demande.

- Flexibilité : Sélection de l'instance adaptée à nos besoins en CPU, mémoire et stockage.

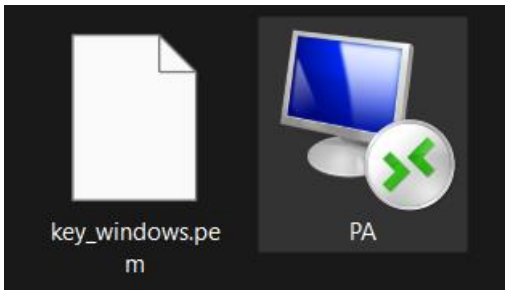
- Sécurité : Utilisation de groupes de sécurité pour contrôler l'accès et protéger les données.

Lancement de l'Instance EC2 :

- Utilisation de la console AWS pour lancer une instance EC2 en choisissant un AMI Windows Server.

Connexion à l'Instance :

- Utilisation d'un client RDP pour se connecter à l'instance EC2 en utilisant la paire de clés téléchargée.

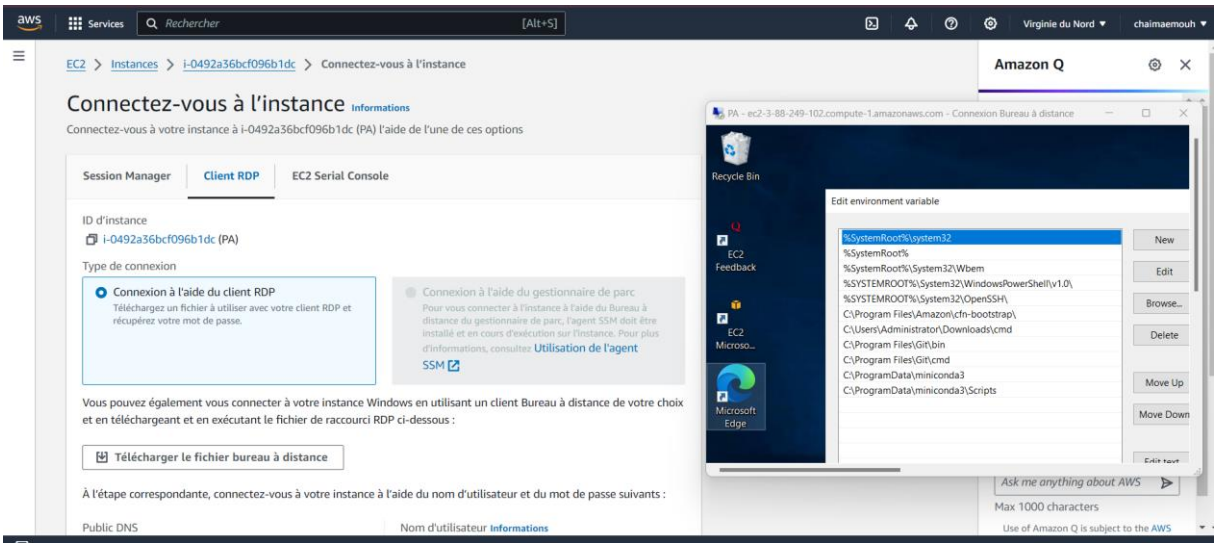


Installation des Logiciels Nécessaires :

- Installation de Python: Téléchargement et installation de Python, en veillant à l'ajouter au PATH du système.

- Installation de Git : Téléchargement et installation de Git pour gérer notre dépôt de code.

- Installation de Miniconda: Téléchargement et installation de Miniconda pour gérer les environnements de développement et les dépendances Python.



Configuration de l'Environnement de Développement

1. Création d'un Environnement Conda :

- Création d'un nouvel environnement Python avec les versions spécifiques nécessaires pour notre projet :

conda create -n myenv python=3.8

conda activate myenv

2. Installation des Packages Nécessaires :

- Installation de TensorFlow et Keras en utilisant conda .

pip install tensorflow=2.13.0 keras=2.13.1

- Installation de Django et d'autres dépendances nécessaires

Déploiement de l'Application

1. Clonage du Dépôt Git:

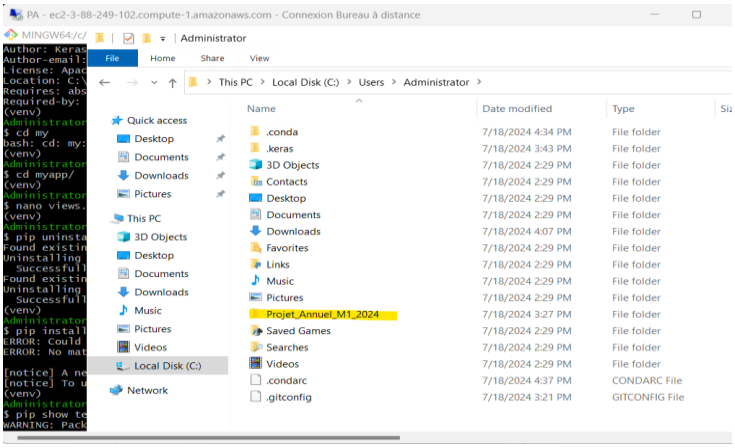
- Clonage du dépôt GitHub contenant le projet Django sur l'instance EC2.

2. Configuration du Projet Django :

- Navigation jusqu'au répertoire de notre projet Django et installation des packages nécessaires.

3. Modification des Paramètres Django :

- Mise à jour du fichier `settings.py` pour inclure l'adresse IP publique de notre instance EC2.

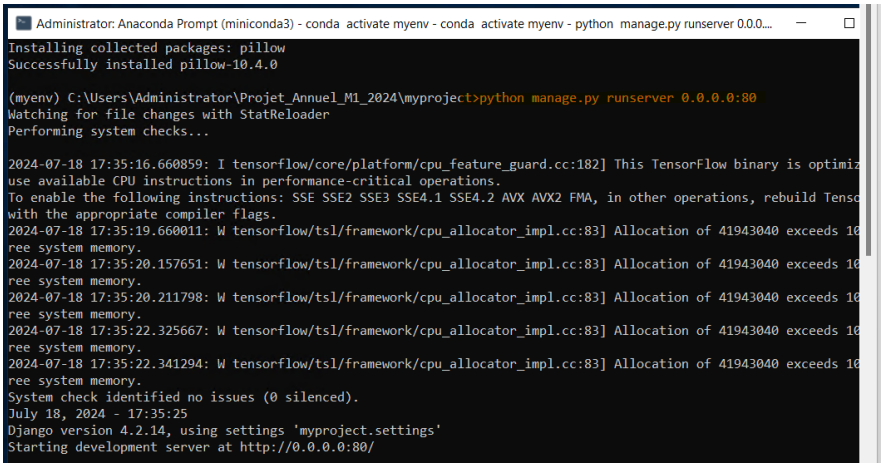


Exécution et Accessibilité de l'Application Django

1. Lancement du Serveur de Développement :

- Démarrage du serveur de développement Django :

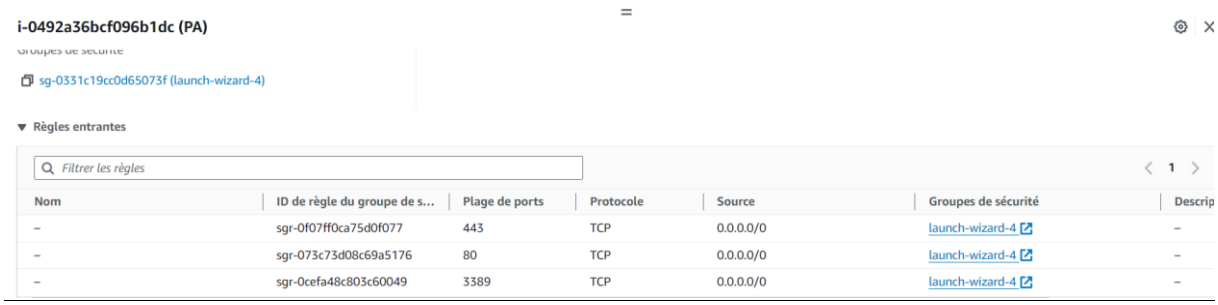
python manage.py runserver 0.0.0.0:80



- Vérification de l'accès à l'application en naviguant vers l'adresse IP publique de notre instance EC2.

Configurations de Sécurité :

- Configuration des groupes de sécurité pour autoriser le trafic HTTP et HTTPS.
- Configuration du pare-feu Windows pour permettre le trafic sur le port 80.



Nous avons réussi à déployer notre application web de manière efficace et sécurisée sur AWS, permettant aux utilisateurs d'accéder facilement à notre système de classification des déchets.

Raisons pour l'Utilisation de Windows sur AWS

<u>Pourquoi nous n'avons pas utilisé Ubuntu :</u>

- Environnement de développement initialement configuré sous Windows.

- Interface graphique conviviale (RDP) offerte par Windows.

-Évitement de problèmes potentiels liés aux différences de compatibilité et aux commandes spécifiques à Linux.

Conclusion

Le projet de classification des déchets ménagers que nous avons entrepris dans le cadre de notre projet annuel 2024 représente une convergence de diverses technologies avancées et de méthodes rigoureuses de développement. À travers ce projet, nous avons non seulement acquis une compréhension approfondie des algorithmes de machine learning, mais nous avons également mis en œuvre une solution pratique et applicable à un problème environnemental pertinent. Voici un résumé détaillé des principaux aspects de notre projet et des leçons apprises.

1. Importance de la Collecte et Préparation des Données

La première étape cruciale de notre projet a été l'acquisition et la préparation d'un dataset de qualité. Nous avons compris que la diversité et la taille du dataset sont fondamentales pour entraîner des modèles d'apprentissage automatique capables de généraliser correctement. En utilisant plusieurs sources de datasets disponibles sur Kaggle, nous avons assemblé une collection complète d'images de déchets divers, couvrant des catégories telles que le carton, le verre, l'aluminium, les déchets organiques, le plastique et les déchets d'équipements électriques et électroniques (DEEE).

La préparation des données a impliqué des étapes de prétraitement minutieuses, y compris la normalisation, le redimensionnement des images et l'annotation des données. Cette rigueur dans la préparation des données a permis d'optimiser l'apprentissage supervisé et de garantir des résultats plus fiables et précis.

2. Exploration et Sélection des Modèles

Nous avons testé plusieurs architectures de réseaux de neurones convolutifs (CNN) pour la classification des déchets. Les modèles initialement développés, bien qu'encourageants, n'ont pas atteint les niveaux de précision souhaités. Les premiers modèles CNN de base ont offert une précision de 62 %, ce qui, bien que prometteur, n'était pas suffisant pour une application pratique.

Pour améliorer les performances, nous avons ensuite testé un modèle CNN enrichi avec des couches de normalisation par batch et des techniques de régularisation supplémentaires. Ce modèle a montré une précision améliorée de 65 %, mais nous recherchions encore mieux.

Enfin, nous avons implémenté MobileNetV2, un modèle pré-entraîné réputé pour son efficacité en classification d'images. En utilisant l'apprentissage par transfert, nous avons pu adapter MobileNetV2 à notre tâche spécifique de classification des déchets. Ce modèle a nettement surpassé les précédents, atteignant une précision de 82 %. Cette amélioration significative a validé notre choix de MobileNetV2 comme modèle final pour notre projet.

3. Développement de l'Application Web

Pour rendre notre solution accessible au public, nous avons développé une application web utilisant le framework Django. L'application permet aux utilisateurs de télécharger des images de déchets ménagers et de recevoir instantanément une classification. Cette interface utilisateur intuitive et conviviale facilite le tri des déchets, contribuant ainsi à améliorer les pratiques de recyclage.

Nous avons utilisé PyCharm comme environnement de développement intégré (IDE) pour son support complet des projets Python et ses outils de développement robustes. De plus, nous avons géré les dépendances et les environnements virtuels avec Miniconda, assurant ainsi une compatibilité optimale des versions des bibliothèques utilisées.

4. Déploiement sur AWS

Le déploiement de notre application sur AWS a représenté une étape finale critique. En utilisant une instance EC2, nous avons pu assurer un hébergement fiable et évolutif de notre application. Les avantages d'AWS, tels que la scalabilité, la flexibilité et la sécurité, ont été essentiels pour garantir la performance continue de notre application.

5. Difficultés Rencontrées et Solutions

Au cours du projet, nous avons rencontré plusieurs défis :

- **Gestion des Environnements** : La compatibilité des versions des bibliothèques a été un défi majeur. L'utilisation de Miniconda pour créer des environnements virtuels isolés a résolu ce problème.
- **Performance des Modèles** : Les premiers modèles CNN n'ont pas atteint la précision souhaitée. L'adoption de MobileNetV2 a permis une amélioration significative.
- **Déploiement sur EC2** : La configuration initiale sur Ubuntu a rencontré des obstacles techniques. Le passage à une instance Windows a simplifié l'installation et la configuration des outils nécessaires.

Nous avons également appris que disposer de comptes AWS gratuits par groupe aurait été extrêmement bénéfique pour explorer davantage de services AWS, comme AWS Rekognition. **AWS Rekognition** aurait pu ajouter une dimension supplémentaire à notre projet en permettant une reconnaissance d'images encore plus sophistiquée et intégrée.

Perspectives et Améliorations Futures

Le projet de classification des déchets ménagers que nous avons développé a ouvert plusieurs avenues pour des améliorations futures et des extensions. Parmi les améliorations possibles, on pourrait envisager :

- **Optimisation Continue des Modèles** : Bien que MobileNetV2 ait montré des résultats prometteurs, il existe toujours des possibilités d'optimisation. L'exploration d'autres architectures de modèles ou l'ajustement des hyperparamètres pourrait encore améliorer la précision.
- **Extension des Catégories de Déchets** : Actuellement, notre modèle se concentre sur six catégories de déchets. L'ajout de nouvelles catégories pourrait rendre le système encore plus utile pour une plus grande variété de matériaux recyclables.
- **Intégration de Services Cloud Avancés** : Utiliser des services cloud avancés comme AWS Rekognition pourrait non seulement améliorer la précision de la classification des images, mais aussi ajouter des fonctionnalités telles que la détection d'objets et l'analyse de scènes.

Conclusion générale

En conclusion, ce projet a été une expérience enrichissante qui nous a permis d'appliquer des techniques avancées d'apprentissage automatique à un problème pratique et pertinent. Nous avons non seulement développé une solution technique robuste, mais nous avons également créé une application web accessible et utile pour les utilisateurs. Ce projet a renforcé notre compréhension de la gestion des données, du développement de modèles, de la création d'applications web et du déploiement sur une infrastructure cloud.

Nous espérons que notre travail contribuera à sensibiliser davantage de personnes à l'importance du tri et du recyclage des déchets, et qu'il pourra servir de base pour des développements futurs dans ce domaine essentiel.

Bibliographie

https://www.kaggle.com/datasets/techsash/waste-classification-data/data

https://www.kaggle.com/datasets/mrgetshjtdone/vn-trash-classification/data

https://www.kaggle.com/datasets/alistairking/recyclable-and-household-waste-classification

https://www.kaggle.com/datasets/ayanbanerjee905/recyclable-garbage

https://k21academy.com/amazon-web-services/aws-ec2-instance/

https://github.com

https://youtu.be/uiPSnrE6uWE?si=fTK7gkhRMIAPWd3j

https://youtu.be/9HCsSD5PMSk?si=a6q3LxoggZEEXzFt

https://penseeartificielle.fr/mobilenet-reconnaissance-images-temps-reel-embarque/

https://aws.amazon.com/fr/ec2/instance-types/

https://aws.amazon.com/fr/rekognition/pricing/