Projet de TP – Introduction à l'IA (en binôme)

Classification d'images CIFAR

Contexte général

Dans ce projet, vous allez concevoir un **système de classification d'images** en utilisant des **méthodes classiques d'apprentissage automatique**.

⏰ Le projet se déroule sur **4 séances de TP** (**1h20 chacune)**.

📊 Jeu de données

Vous utiliserez le jeu de données **CIFAR-10**, composé d'images couleur de taille 32×32 pixels.

Choix des classes

Chaque groupe devra :

- sélectionner **exactement 5 classes** parmi les 10 classes de CIFAR-10 ;
- justifier brièvement ce choix (similarité visuelle, diversité, difficulté attendue, etc.).

Les 10 classes disponibles sont :

- airplane
- automobile
- bird
- cat
- deer
- dog
- frog
- horse
- ship
- truck

Taille du jeu de données

Afin de limiter le temps de calcul :

- vous utiliserez un sous-ensemble d'environ **8 000 à 10 000 images d'entraînement**
- et environ **2 000 images de test** ;
- seules les images appartenant aux 5 classes sélectionnées doivent être conservées.

Description générale du projet

Le projet consiste à mettre en œuvre un **pipeline complet de classification d'images**, comprenant :

1. l'exploration, l'analyse et la préparation des données (**NumPy** + **Pandas + visualisation**);
2. l'extraction de caractéristiques visuelles (**OpenCV**) ;
3. l'apprentissage de modèles de classification (**Scikit-learn**) ;
4. l'évaluation, la comparaison et l'analyse des erreurs (**Scikit-learn**).

Déroulement par séance (approximatif)

1️⃣ Séance 1 – Chargement des données et exploration

🎯 *Objectif : comprendre le jeu de données et établir un premier modèle de référence.*

Travail à réaliser :

- Charger le jeu de données **CIFAR-10**.
- Filtrer le jeu de données pour ne conserver que les **5 classes choisies**.
- Stocker les images et labels sous forme de tableaux NumPy.
- Créer une structure Pandas contenant les informations essentielles (identifiant, label, nom de la classe).
- Visualiser :
    - la répartition des classes sélectionnées ;
    - des exemples d'images pour chaque classe.
- Mettre en place un **modèle de référence simple** (pixels bruts + MLP ou SVM).
- Évaluer ce modèle avec l'accuracy et une matrice de confusion. Qu'observez-vous ?

2️⃣ Séance 2 – Extraction de caractéristiques avec OpenCV

🎯 *Objectif : améliorer la représentation des images à l'aide de descripteurs visuels simples.*

Dans la séance précédente, les images ont été utilisées sous forme de pixels bruts.
Cette représentation contient beaucoup d'informations, mais aussi beaucoup de bruit,

et elle n'est pas toujours adaptée aux modèles d'apprentissage automatique classiques.

L'objectif de cette séance est de **construire une représentation plus informative et plus compacte des images**, en utilisant des opérations simples de traitement d'images vues en TP OpenCV.

En choisissant des caractéristiques pertinentes (couleur, contours, forme globale), vous facilitez le travail du classifieur. Cette étape est souvent déterminante pour les performances finales, parfois davantage que le choix du modèle lui-même.

Travail à réaliser :

- Implémenter au moins deux familles de caractéristiques parmi les suivantes (issues des outils OpenCV déjà vus) :
    - Caractéristiques de couleur : histogrammes sur les canaux (RGB ou HSV), éventuellement normalisés.
    - Caractéristiques de contours : détection de bords avec Canny, puis statistiques simples (ex. proportion de pixels "bord", moyenne/variance de l'image de bords).
    - Caractéristiques par seuillage + morphologie : seuillage (dont Otsu), puis opérations morphologiques (ouverture/fermeture) et extraction de statistiques (ex. proportion de pixels blancs avant/après).
    - Tiny images : conversion en niveaux de gris puis redimensionnement (ex. 8×8), aplatissement et normalisation.
- Fusionner les caractéristiques choisies (concaténation) afin d'obtenir un vecteur compact par image.
- Construire les matrices de caractéristiques :
    - `F_train` de taille *(N_train, d)*
    - `F_test` de taille *(N_test, d)*
- Réaliser une comparaison visuelle entre classes :
    - distribution d'une ou plusieurs caractéristiques (boxplot / histogrammes / courbes),
    - ou comparaison d'histogrammes moyens par classe.

👉 *Aucune méthode n'est imposée : le choix et la combinaison des caractéristiques font partie du travail.*

3️⃣ Séance 3 – Apprentissage et comparaison de modèles

🎯*Objectif : entraîner et comparer les modèles vus en cours.*

Travail à réaliser :

- Entraîner au moins **deux modèles différents** parmi :
    - MLP,

    - SVM,
    - arbre de décision.
- Utiliser des pipelines Scikit-learn (normalisation + modèle).
- Réaliser une recherche d'hyperparamètres simple et limitée pour **un seul modèle**.
- Évaluer les performances à l'aide :
    - de l'accuracy,
    - de matrices de confusion,
    - de performances par classe.
- Regrouper les résultats dans un tableau Pandas et exporter un fichier CSV.

4️⃣ Séance 4 – Analyse des erreurs et amélioration

🎯 *Objectif : analyser les limites du système et proposer une amélioration simple.*

Travail à réaliser :

- Identifier les principales erreurs de classification :
    - visualisation d'images mal classées,
    - classes les plus souvent confondues.
- Réaliser **une amélioration parmi les suivantes** :
    - réduction de dimension (PCA) avant classification,
    - comparaison des performances selon le type de caractéristiques,
    - ajustement plus fin des hyperparamètres.
- Comparer quantitativement les performances avant et après amélioration. Qu'observez-vous ?
- Rédiger une analyse critique des résultats. Nous attendons également une réflexion autour des métriques d'évaluation.

📍 Livrables attendus

Vous devez rendre :

- un notebook Python propre, structuré et commenté ;
- un fichier CSV contenant les performances des différents modèles ;
- un court rapport (1 page maximum) comprenant :
    - les 5 classes choisies et la justification de ce choix ;
    - le meilleur modèle obtenu ;
    - une analyse des principales confusions observées.