Projet Final : Classification de chiffres manuscits (MNIST) avec un réseau de neurones (Multi-Layer Perceptron (MLP))

Author : Badr TAJINI - Introduction to AI - ESIEE-IT 2024-2025

Résumé

Le projet final consiste à explorer les fondamentaux de l'apprentissage automatique en construisant et en évaluant un modèle de classification d'images. L'objectif est de se familiariser avec les étapes clés du développement d'un modèle de `machine learning`, de la préparation des données à l'évaluation des performances, en utilisant un cas d'usage classique : la reconnaissance de chiffres manuscrits à partir du dataset MNIST. Ce projet se focalisera sur la compréhension des concepts et l'expérimentation pratique, sans nécessiter d'expertise préalable en `deep learning`.

Objectifs pédagogiques

1. **Chargement et préparation des données :**
    - Apprendre à charger un dataset d'images standard (MNIST) en utilisant des librairies Python.
    - Comprendre l'importance de la préparation des données, incluant la normalisation.
    - Diviser les données en ensembles d'entraînement et de test pour évaluer les performances du modèle.
2. **Construction d'un modèle de classification :**
    - Découvrir l'architecture d'un réseau neuronal simple, le Perceptron Multi-Couches (MLP).
    - Utiliser une librairie Python pour construire et configurer un modèle MLP avec différentes architectures.
    - Comprendre le rôle des hyperparamètres dans la définition du modèle.
3. **Entraînement du modèle :**

- Apprendre à entraîner un modèle de *machine learning* sur un ensemble de données d'entraînement.
- Observer le processus d'entraînement et comprendre son objectif.
- Expérimenter avec le nombre d'itérations d'entraînement.

4. **Évaluation des performances du modèle :**
   - Utiliser un ensemble de données de test pour évaluer les performances du modèle entraîné.
   - Calculer la précision du modèle en tant que métrique d'évaluation.
   - Comprendre l'importance de l'évaluation pour mesurer la capacité du modèle à généraliser.
5. **Visualisation des résultats et analyse des erreurs :**
   - Utiliser des outils de visualisation pour examiner les prédictions du modèle.
   - Construire et interpréter une matrice de confusion pour identifier les types d'erreurs commises par le modèle.
   - Développer une compréhension qualitative des forces et des faiblesses du modèle.

Conditions du projet

1. **Environnement de développement :** Utilisation de Python avec les librairies `scikit-learn`, `numpy`, `matplotlib` et `seaborn`. L'exécution du code peut se faire dans un environnement local `(VS Code (Jupyter Notebook)` ou Docker `(Jupyter Notebook)` ou en ligne `(e.g., Google Colab)`.
2. **Travail `individuel` ou `binôme` ou en `petite équipe` :** Le travail individuel est encouragé pour une compréhension approfondie. Binôme ou une équipe `(maximum 4 étudiant(e)s)` sont autorisé(e)s pour favoriser la discussion et l'entraide.
3. **Livrables :** Rendu sous forme de `Notebook unique` incluant tous les codes Python pour chaque étape, ainsi qu'un rapport concis expliquant les démarches, les observations et les conclusions pour chaque étape.

> **Exemple : livrable d'une équipe = un notebook pour tout le projet avec toutes les `(cinq)` étapes.**

Phases du projet

Phase 1 : Classification de fruits avec un arbre de décision

> [!IMPORTANT] Les instructions pour finaliser la première étape sont affichées en haut du code joint de cette étape.

- **Objectif :** Introduction aux concepts de base de la classification avec un modèle simple et l'exécution de code fourni.
- **Tâches :**
  1. Essayer de lire attentivement le pseudo-code (`0_pseudocode_classification_fruits.md`) pour comprendre le script Python fourni (`1_classification_fruits.py`)
  2. Exécuter le script Python fourni (`1_classification_fruits.py`) pour entraîner un arbre de décision sur un dataset simple de fruits.
  3. Observer les prédictions du modèle et expérimenter en modifiant les données d'entrée.
  4. Rédiger une courte description de ce que fait le code et des résultats observés.

Phase 2 : Construction d'un modèle MLP sur MNIST - étapes préliminaires

> [!IMPORTANT] Les instructions pour finaliser la deuxième étape sont affichées en haut du code joint de cette étape.

- **Objectif :** Se familiariser avec le dataset MNIST et les étapes de construction d'un modèle Perceptron Multi-Couches (MLP).
- **Tâches :**
  1. Exécuter le script Python fourni (`2_construire_mlp_mnist.py`) pour charger le dataset MNIST et construire différentes instances de modèles MLP.
  2. Examiner la forme des données MNIST et les différents paramètres utilisés pour construire les modèles MLP.
  3. Décrire les étapes nécessaires pour construire un modèle MLP avec scikit-learn et l'impact de certains hyperparamètres.

Phase 3 : Entraînement et évaluation d'un modèle MLP sur MNIST

> [!IMPORTANT] Les instructions pour finaliser la troisième étape sont affichées en haut du code joint de cette étape.

- **Objectif :** Entraîner un modèle MLP sur le dataset MNIST et évaluer ses performances.
- **Tâches :**

1. Exécuter le script Python fourni (`3_entrainer_evaluer_mlp_mnist.py`) pour entraîner un modèle MLP sur les données MNIST et évaluer sa précision.
2. Observer le processus d'entraînement et la précision atteinte par le modèle.
3. Expérimenter en modifiant le nombre d'itérations d'entraînement et l'architecture du modèle, et observer l'impact sur la précision.

Phase 4 : Amélioration de la précision du modèle MLP

> [!IMPORTANT] Les instructions pour finaliser la quatrième étape sont affichées en haut du code joint de cette étape.

- **Objectif :** Explorer des techniques pour améliorer la précision du modèle MLP.
- **Tâches :**
  1. Exécuter le script Python fourni (`4_ameliorer_precision_mlp_mnist.py`) qui explore différentes architectures de modèles, la régularisation et les algorithmes d'optimisation.
  2. Observer l'impact de ces modifications sur la précision du modèle.
  3. Expérimenter en modifiant les hyperparamètres liés à l'architecture, à la régularisation et à l'optimisation pour tenter d'améliorer la précision.

Phase 5 : Visualisation des prédictions du modèle

> [!IMPORTANT] Les instructions pour finaliser la cinquième étape sont affichées en haut du code joint de cette étape.

- **Objectif :** Visualiser les prédictions du modèle et analyser ses erreurs à l'aide d'une matrice de confusion.
- **Tâches :**
  1. Exécuter le script Python fourni (`5_visualiser_predictions_mnist.py`) pour visualiser les prédictions du modèle sur un échantillon d'images de test et afficher la matrice de confusion.
  2. Interpréter les visualisations et la matrice de confusion pour comprendre qualitativement les performances du modèle et les types d'erreurs qu'il commet.
  3. Décrire les observations tirées des visualisations et de la matrice de confusion, en reliant les erreurs aux caractéristiques potentielles des images.

Critères d’évaluation

1. **Exécution correcte du code :** Capacité à exécuter les scripts Python fournis et à comprendre leur fonctionnement.
2. **Expérimentation et observation :** Engagement dans l'expérimentation en modifiant les paramètres et observation attentive des résultats.
3. **Compréhension des concepts :** Qualité des explications et des descriptions des concepts d'apprentissage automatique mis en œuvre.
4. **Analyse des résultats :** Capacité à analyser les résultats des expériences et à tirer des conclusions pertinentes.
5. **Qualité du rapport :** Clarté, concision et pertinence du rapport décrivant les démarches, les observations et les conclusions pour chaque étape.

Livrables

1. **Codes Python :** Les cinq scripts Python utilisés pour chaque étape.
2. **Rapport :** Un document (`Notebook unique`) incluant :
   - **Une introduction au projet et à ses objectifs.**
   - **Pour chaque étape (lire la note en bas [1]) :**
     - Une `description` de l'objectif de l'étape.
     - Une `explication` du code fourni.
     - Un `résumé` des observations et des résultats obtenus lors de l'exécution du code.
     - Une `description` des expérimentations réalisées et de leurs résultats.
     - Les `conclusions` tirées de l'étape.
   - **Une conclusion générale récapitulant les apprentissages du projet.**

> [1] Les instructions pour finaliser chaque étape sont affichées en haut du code joint de chaque étape.

Deadline

- **Rendu Final du projet (Github privé)** : 21/02/2025 à 23h59 (heure de Paris).
  - Vous devez mettre en ligne votre dépôt privé sur Github avec un tag final, et inclure le `Notebook unique` détaillé.
- **Rendu Final des labs (Github privé)** : 21/02/2025 à 23h59 (heure de Paris).
  - **Deux labs** : obligatoire à rendre au choix.

- **Les autres labs** : optionnel, cependant, ils seront comptés commes des bonus dans la note final des labs.

Pondération de l'évaluation

- **Projet** : 50%
- **Labs** : 50%
- **Participation** : 10%

Identification

P.S. : N'oubliez pas de mentionner : votre nom - filière - année académique - école (pour les crédits sur votre Github).

Livraison (traçabilité)

Afin d'assurer la traçabilité de votre travail (pour le projet et les labs), veuillez indiquer votre nom complet et celui de vos coéquipiers dans le formulaire Google suivant : Lien.