Rapport de Projet

IFT3395 Competition 2

1. Header Information / Informations d'En-tête

- **Project Title:** Diabetic Retinopathy Detection (Détection de la Rétinopathie Diabétique)
- **Kaggle Team Name:** [Votre Nom d'Équipe Ici]
- **Team Members:**
  - [Name 1] - [Student Number 1]
  - [Name 2] - [Student Number 2]

---

2. Introduction / Introduction

Description du problème: Le but de ce projet est de classifier automatiquement des images de rétine en 5 niveaux de sévérité de la rétinopathie diabétique (0 à 4). Le défi principal réside dans la nature des données : des images de faiblerésolution (28x28 pixels), un fort déséquilibre de classes (la classe 0 est majoritaire), et la contrainte stricte de ne pas utiliser de données externes ni de modèles pré-entraînés.

Résumé de notre approche: Nous avons adopté une approche en deux étapes.

- **Milestone 1 :** Implémentation "from scratch" en NumPy de modèles de base (k-NN et MLP) pour établir une performance de référence et comprendre les données brutes.
- **Milestone 2 :** Développement d'une solution hybride avancée combinant un CNN léger (architecture ResNet) entraîné avec des techniques modernes de régularisation (Mixup, One Cycle Policy) et un ensemble de modèles classiques (Random Forest, SVM, etc.) via `scikit-learn`. L'ensemble final utilise une recherche aléatoire pour optimiser les poids de chaque modèle.

3.Conception des caractéristiques

Description des méthodes de prétraitement:

1. **Normalisation:** Les pixels des images ont été normalisés dans l'intervalle [0, 1] en divisant par 255.
2. **Aplatissement:** Pour les modèles classiques (k-NN, SVM, Trees), les images 28x28x3 ont été aplaties en vecteurs de dimension 2352.
3. **Extraction de caractéristiques pour l'ensemble:** En plus des pixels bruts, nous avons extrait des statistiques de couleur (moyenne, écart-type, min, max, percentiles) pour chaque canal RGB et pour la version en niveaux de gris de l'image.
4. **Augmentation de données:** Pour le CNN, nous avons utilisé `RandomHorizontalFlip`, `RandomRotation`, et `ColorJitter` pendant l'entraînement.

Justification:

- **Pourquoi des caractéristiques manuelles?:** Bien que les CNN apprennent leurs propres caractéristiques, les modèles comme Random Forest bénéficient de statistiques globales explicites

(comme la luminosité moyenne ou la variance de couleur) qui peuvent capturer des anomalies globales dans l'œil sans nécessiter de cohérence spatiale.
- **Pourquoi l'augmentation?:** Avec seulement ~1000 images d'entraînement, le sur-apprentissage (overfitting) est inévitable. L'augmentation force le modèle à apprendre des caractéristiques invariantes (la maladie ne change pas si l'image est retournée).

---

4. Algorithmes

Milestone 1 (NumPy Only):

- **k-NN:** Implémentation vectorisée calculant les distances L2.
- **MLP:** Réseau de neurones à 2 couches caché avec activation ReLU et sortie Softmax, entraîné par rétropropagation manuelle (SGD).

Milestone 2 (PyTorch + Scikit-learn):

- **ResNet-like:** Un réseau convolutionnel personnalisé utilisant des blocs résiduels (**ResBlocks**).
- **Ensemble Learners:** ExtraTreesClassifier, RandomForestClassifier, HistGradientBoostingClassifier, et SVC (Support Vector Classifier).

---

5. Méthodologie

Explication du pipeline d'apprentissage:

1. **Data Loading:** Chargement des fichiers `.pkl`.
2. **Stratification:** Utilisation de `StratifiedKFold` (5 folds) pour garantir que chaque pli contient la même proportion de classes que le jeu de données total, ce qui est crucial vu le déséquilibre (Classe 0 dominante).
3. **Entraînement:**
    - Le CNN est entraîné sur 30 époques pour chaque pli.
    - Les modèles `sklearn` sont entraînés sur les caractéristiques extraites (pixels + stats).
4. **Ensembling:** Les prédictions (probabilités OOF - Out Of Fold) sont combinées. Une recherche aléatoire (Random Search) de 2000 itérations trouve la combinaison linéaire optimale des poids des 5 modèles pour maximiser l'accuracy de validation.

Techniques de régularisation:

- **Mixup:** Nous avons implémenté Mixup, qui mélange deux images et leurs labels (ex: 0.6 * ImageA + 0.4 * ImageB). Cela encourage le modèle à avoir des frontières de décision linéaires et fluides entre les classes, réduisant la mémorisation du bruit.
- **Data Augmentation & TTA:** Augmentation classique à l'entraînement et "Test Time Augmentation" (moyenne des prédictions de l'image originale et de son miroir) à l'inférence.
- **Label Smoothing:** Au lieu de viser des cibles [0, 1] strictes (one-hot), nous utilisons des cibles adoucies (ex: [0.1, 0.9]) pour éviter que le modèle ne soit trop confiant.

Méthodes d'optimisation:

- **One Cycle Policy:** Pour le CNN, nous utilisons l'ordonnanceur `OneCycleLR`. Au lieu d'un taux d'apprentissage (LR) fixe ou décroissant, le LR augmente rapidement jusqu'à un maximum puis

redescend.
    - *Justification:* Cela permet au modèle de traverser rapidement les plateaux de perte et de converger vers des minima plus larges et plus généralisables ("Super-convergence").
- **WeightedRandomSampler:** Pour combattre le déséquilibre de classes, nous échantillonnons les images rares plus souvent dans chaque batch.

Pourquoi ces choix?:

- **ResNet:** Nous avons choisi une architecture résiduelle car les CNN standards profonds souffrent du problème de disparition de gradient. Les "skip connections" permettent d'entraîner des réseaux plus profonds même sur de petites images (28x28), capturant des caractéristiques plus complexes sans dégrader la performance. C'est un standard robuste, préférable à des architectures trop complexes comme EfficientNet pour cette résolution.
- **Mixup & OneCycle:** Ces techniques sont issues des meilleures pratiques récentes (inspirées de fast.ai et des compétitions Kaggle) pour maximiser la performance quand les données sont limitées.

---

6. Résultats

Analyse détaillée et Progression:

Phase	Model	Validation Accuracy	Notes
Milestone 1	k-NN (k=5)	~0.4537	Baseline simple. Souffre de la haute dimensionnalité.
Milestone 1	MLP (Custom)	~0.4907	Meilleur que k-NN, capable d'apprendre des relations non-linéaires.
Milestone 2	Random Forest / ExtraTrees	~0.5000 - 0.5300	Robuste, bonne gestion du bruit.
Milestone 2	Custom CNN (ResNet)	~0.4900 - 0.5200	Apprend les motifs spatiaux, mais instable sans régularisation.
Milestone 2	**Ensemble (Final)**	**~0.5259 - 0.5300+**	La combinaison des forces variées donne le meilleur résultat.

Nous observons une progression claire : les modèles paramétriques (MLP) battent les non-paramétriques (k-NN), et les modèles profonds/ensemblistes (M2) surpassent les approches simples de M1. Le CNN seul n'est pas toujours le meilleur à cause du manque de données, mais il apporte une diversité essentielle à l'ensemble.

---

7. Discussion

Avantages:

- **Robustness:** L'ensemble réduit la variance. Si le CNN se trompe, les Random Forests peuvent corriger.
- **Gestion du déséquilibre:** L'utilisation de `WeightedRandomSampler` et de `StratifiedKFold` assure que les classes rares (comme la classe 1) ne sont pas ignorées.

- **Généralisation:** L'utilisation de Mixup et TTA améliore significativement la capacité du modèle à généraliser sur des données non vues.

Désavantages:

- **Complexité:** Maintenir 5 modèles différents est lourd.
- **Temps d'inférence:** TTA double le temps de prédiction, ce qui pourrait être problématique en temps réel.
- **Taille d'image:** Travailler en 28x28 limite la capacité à voir des micro-anévrismes typiques de la rétinopathie.

Idées d'amélioration:

- Utiliser des images de plus haute résolution si possible.
- Explorer des techniques d'attention (comme les Vision Transformers) si plus de données étaient disponibles.
- Pré-entraînement auto-supervisé sur les données non étiquetées (si disponibles).

---

8. Références

1. **Learning Rate Schedulers (One Cycle Policy):**
    - Kaggle Notebook - *Learning Rate Schedulers*: https://www.kaggle.com/code/snnclsr/learning-rate-schedulers
    - *Explication visuelle des stratégies de learning rate et de leur impact sur la convergence.*
2. **Mixup Augmentation:**
    - Kaggle Notebook - *Mixup Data Augmentation*: https://www.kaggle.com/code/kaushal2896/data-augmentation-tutorial-basic-cutout-mixup
    - *Démonstration pratique de l'implémentation de Mixup pour régulariser les réseaux de neurones.*
3. **ResNet & Image Classification:**
    - PyTorch Tutorials - *Training a Classifier*: https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html
    - *Guide officiel pour construire et entraîner des modèles de type ResNet.*