

# Mini-Projet Machine Learning – Classification d'images avec PyTorch

Auteurs : OUARRADI Hanae – SAHI Ines – EL KARIMI Yassine

Filière : MD4

---

## 1. Contexte et objectif

Ce mini-projet a été réalisé dans le cadre du module de Machine Learning. L'objectif est de sélectionner un dataset réel, définir une tâche claire de machine learning, construire un modèle de classification d'images avec PyTorch, puis l'améliorer de manière systématique. Le projet s'inscrit dans l'option classification d'images.

## 2. Dataset

Le dataset principal utilisé est FashionMNIST, accessible via la bibliothèque `torchvision.datasets`. Il contient 70 000 images en niveaux de gris de taille 28x28 pixels, réparties en 10 classes. Le dataset est divisé en 60 000 images d'entraînement et 10 000 images de test. Des images externes personnalisées respectant les mêmes classes ont également été utilisées afin d'évaluer la capacité de généralisation du modèle.

## 3. Prétraitement des données

Les images sont converties en tenseurs PyTorch puis normalisées à l'aide de la moyenne (0.2860) et de l'écart-type (0.3530). Une légère augmentation de données est appliquée sur l'ensemble d'entraînement à l'aide d'un retournement horizontal aléatoire. Les images externes subissent exactement le même prétraitement.

## 4. Modèle de base et améliorations

Un réseau de neurones convolutionnel (CNN) a été implémenté. Il est composé de couches convolutionnelles pour l'extraction de caractéristiques visuelles, suivies de couches fully connected pour la classification finale. Des techniques d'amélioration telles que Batch Normalization, Dropout et un scheduler du learning rate ont été intégrées afin de stabiliser l'apprentissage et d'améliorer les performances.

## 5. Entraînement et évaluation

L'entraînement est réalisé à l'aide de la fonction de perte `CrossEntropyLoss` et de l'optimiseur Adam. Les performances sont évaluées via la métrique d'accuracy sur les jeux d'entraînement et de test. Le meilleur modèle est automatiquement sauvegardé en fonction de l'accuracy obtenue sur le jeu de test.

## 6. Tableau récapitulatif des expériences

Expérience	Architecture	Optimisation	Accuracy Test
Modèle de base	Réseau simple (MLP)	Optimisation standard	~87 %
CNN amélioré	Conv + BN + Dropout	Adam + Scheduler	~92–94 %
Tests externes	CNN amélioré	Modèle figé	Résultats cohérents

## 7. Test sur une image externe

Une image externe unique a été chargée depuis Google Drive et soumise au même prétraitement que les images du dataset FashionMNIST. Le modèle a correctement prédit la classe associée, ce qui démontre sa capacité à généraliser sur des données non vues lors de l'entraînement.

## **8. Conclusion et perspectives**

Ce projet a permis de mettre en œuvre un pipeline complet de Machine Learning avec PyTorch, allant de la préparation des données à l'évaluation sur des images externes. Les résultats obtenus sont satisfaisants pour un CNN simple et optimisé. Des améliorations futures pourraient inclure l'utilisation de modèles plus profonds ou de techniques d'augmentation de données plus avancées.