**CLASSIFICATION DES IMAGES MNIST**

# CHOIX TECHNIQUES

## Bibliothèques Utilisées

**TensorFlow/Keras**: Permettent de créer facilement des CNN profonds et d’ajuster les hyperparamètres comme le nombre de couches, la taille des filtres, et la régularisation.

**Matplotlib**: Pour visualiser l’évolution des performances du modèle, comme les courbes de perte et de précision au cours de l’entraînement.

**NumPy** : est essentiel pour la manipulation des données sous forme de tableaux multidimensionnels. Il est utilisé en arrière-plan pour toutes les opérations numériques dans TensorFlow et Keras, y compris la manipulation des images et des labels.

## Algorithmes Utilisés

**CNN (Convolutional Neural Networks)** : J’ai utilisé un modèle CNN, car ce type de réseau est efficace pour extraire des caractéristiques visuelles via des couches de convolution et de pooling. Les CNN sont généralement le choix préféré pour la classification d'images en raison de leur capacité à capturer des relations spatiales dans les données.

**CNN Profond** : J’ai augmenté la profondeur du réseau, pour que le modèle soit capable de capturer des caractéristiques à différents niveaux d’abstraction. Cela signifie plus de filtres à chaque couche, permettant de mieux distinguer les caractéristiques complexes des images, comme les textures, les motifs, et les contours plus fins.

**Batch Normalization** : Je l’ai intégrée après chaque couche convolutionnelle pour stabiliser l'entraînement des réseaux profonds. En normalisant les sorties intermédiaires, Batch Normalization aide à éviter des problèmes tels que la vanishing gradient (diminution du gradient) ou l'explosion des valeurs, qui peuvent apparaître dans des réseaux plus profonds.

## Optimisateurs

**Adam** : Cet optimiseur adaptatif est populaire pour son efficacité à ajuster les taux d'apprentissage pour chaque paramètre individuellement.

**RMSprop** : Testé avec un learning rate scheduler pour ajuster dynamiquement le taux d'apprentissage, permettant une convergence plus douce si la performance stagne.

# Résultats Obtenus

**Modèle CNN de Base:**

Test Accuracy : 98.59%

Test Loss : 0.0498

**CNN Profond:**

Test Accuracy : 98.98%

Test Loss : 0.0389

**Modèle CNN profond avec Batch Normalization** :

Test Accuracy : 99.3%

Test Loss : 0.022

**Conclusion**: L'ajout de Batch Normalization dans le modèle CNN profond a significativement amélioré les performances en termes de précision et de réduction de la perte, prouvant son efficacité pour stabiliser l'entraînement.

# Pistes d'Amélioration

**Support Vector Machine (SVM)**: Bien que les CNN soient adaptés pour la classification d'images, les SVM peuvent être explorés comme classificateurs en utilisant les caractéristiques extraites des couches convolutionnelles d’un CNN. Cela pourrait améliorer la séparation des classes dans l’espace des caractéristiques.

**Réseaux Récurrents (RNN)**: Bien que principalement utilisés pour des données séquentielles, les RNN (ou LSTM) pourraient être appliqués pour des tâches spécifiques comme la classification d'images avec séquences (par exemple, vidéo ou séries d'images).