TP 3 : Classification d’images par Réseaux de Neurones Convolutifs (CNN)

Réalisée par Sara ADJAHO

Master 2: Data Science

Université : Institut de Mathématiques et des Sciences Physiques (IMSP-Dangbo)

Développement et évaluation d'un réseau de neurones convolutif pour la reconnaissance automatique de chiffres manuscrits sur le jeu de données MNIST

Chargement et Exploration des Données MNIST

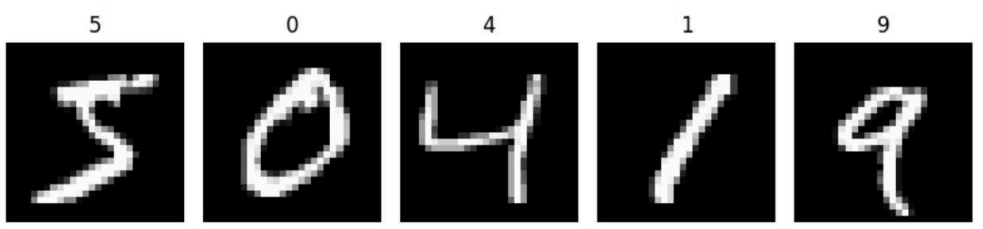
Structure du Jeu de Données

Le jeu de données MNIST est composé de 60 000 images d'entraînement et 10 000 images de test. Chaque image mesure 28×28 pixels en niveaux de gris, représentant des chiffres manuscrits de 0 à 9.

- X_train shape : (60000, 28, 28)
- y_train shape : (60000,)
- X_test shape : (10000, 28, 28)
- y_test shape : (10000,)

Les variables y_train et y_test contiennent les étiquettes correspondantes aux chiffres manuscrits associés à chaque image, permettant l'apprentissage supervisé du modèle.

Visualisation des Exemples



Les images ci-dessus montrent cinq exemples tirés du jeu d'entraînement, illustrant la variabilité de l'écriture manuscrite que le modèle devra apprendre à reconnaître.

Prétraitement des Données

01

Normalisation des Pixels

Conversion des images en float32 et normalisation entre 0 et 1 en divisant par 255, facilitant l'apprentissage et assurant une meilleure stabilité numérique.

02

Redimensionnement des Images

Ajout d'une dimension supplémentaire pour le canal de couleur (28, 28, 1), respectant le format attendu par les couches convolutives du CNN.

03

Encodage One-Hot

Transformation des étiquettes en vecteurs one-hot, étape indispensable pour la classification multiclasses permettant de traiter chaque chiffre comme une catégorie indépendante.

```
x_train = x_train.astype('float32') / 255
x_test = x_test.astype('float32') / 255
x_train = x_train.reshape((-1, 28, 28, 1))
x_test = x_test.reshape((-1, 28, 28, 1))
y_train = to_categorical(y_train)
y_test = to_categorical(y_test)
```

Cette phase de prétraitement garantit la cohérence des données et optimise l'efficacité de l'apprentissage du réseau de neurones.

Architecture du Réseau de Neurones Convolutif

Couches Convolutionnelles

Trois couches Conv2D successives (32, 64, 64 filtres 3×3) avec activation ReLU pour l'extraction progressive de caractéristiques hiérarchiques, des motifs simples aux patterns complexes.

Max Pooling

Réduction des dimensions spatiales 2×2 après chaque convolution pour diminuer les paramètres et limiter le surapprentissage.

Couches Denses

Flatten suivi d'une couche Dense de 128 neurones avec Dropout 50% pour régularisation, puis sortie softmax à 10 neurones.

Ce choix d'architecture offre un compromis optimal entre capacité d'apprentissage, complexité et prévention du surapprentissage grâce aux filtres croissants et techniques de régularisation.

Compilation et Stratégie d'Entraînement

Paramètres de Compilation

Le modèle a été compilé avec des choix stratégiques pour optimiser l'apprentissage :

- **Optimiseur Adam** : Convergence rapide avec ajustement automatique du taux d'apprentissage
- **Fonction de perte** : categorical_crossentropy adaptée à la classification multiclasse
- **Métrique** : accuracy pour évaluer la proportion de prédictions correctes

Configuration d'Entraînement

- Maximum de 50 époques avec batch_size de 64
- Early stopping avec patience de 3 époques sur val_loss
- 20% des données réservées à la validation interne
- Restauration automatique des meilleurs poids

Early Stopping

Le mécanisme d'arrêt anticipé surveille la perte de validation et stoppe l'entraînement lorsqu'elle n'évolue plus après trois itérations, évitant ainsi le surapprentissage tout en garantissant des performances optimales.

Résultats de l'Entraînement

9

Époques Totales

Arrêt automatique après 9 époques grâce à l'early stopping

99.3%

Accuracy Finale

Précision d'entraînement atteinte à la dernière époque

99.0%

Validation Accuracy

Précision stable sur les données de validation

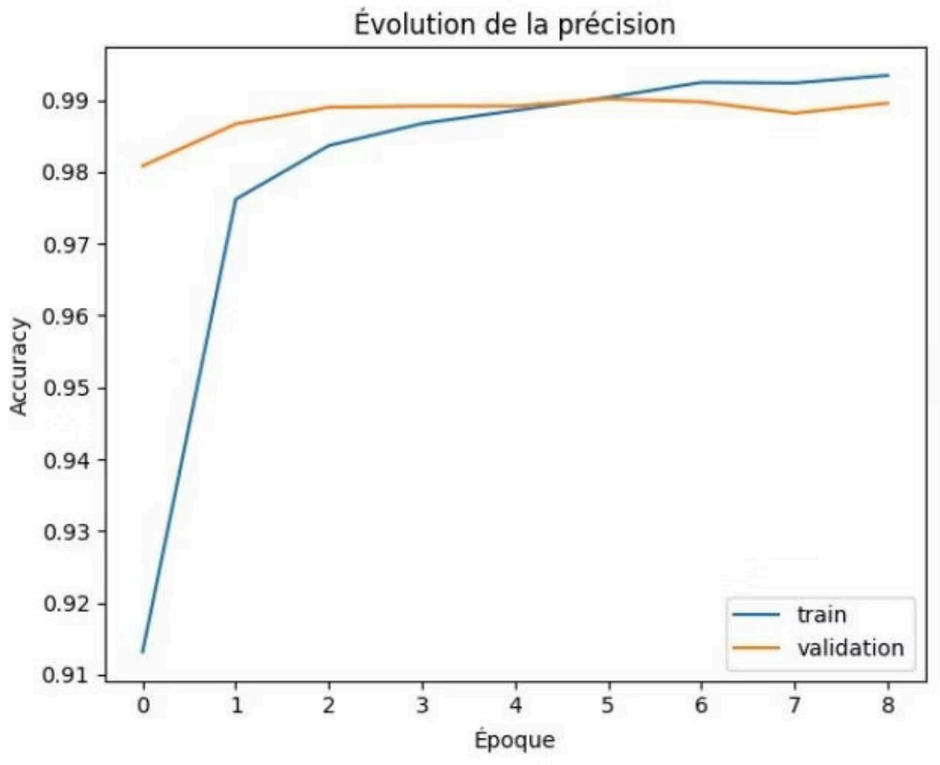
0.018

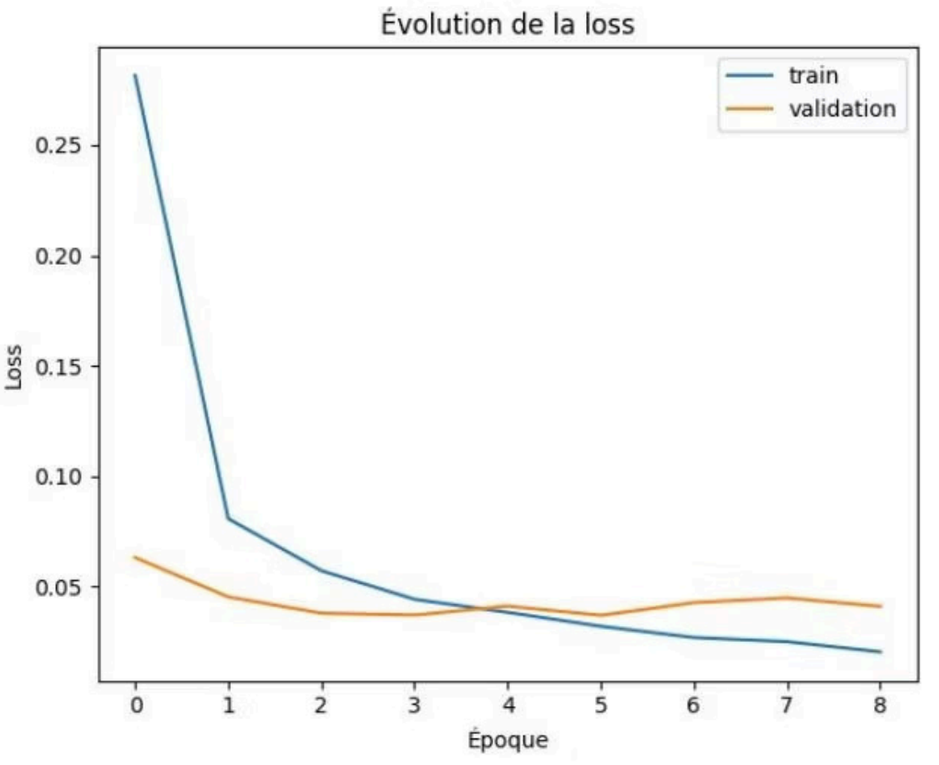
Loss Finale

Perte d'entraînement minimale obtenue

Au cours de l'entraînement, le modèle a montré une amélioration rapide dès les premières itérations. L'accuracy d'entraînement est passée de 80,6% à 99,3% entre la première et la neuvième époque, tandis que l'accuracy de validation a atteint un niveau élevé et stable autour de 98,9% à 99,0% dès la troisième époque. L'écart faible entre les valeurs d'entraînement et de validation montre que le modèle généralise bien et ne présente pas de signe marqué de surapprentissage.

Évolution des Performances





Analyse de la Précision

La précision sur les données d'entraînement augmente rapidement dès les premières itérations, passant d'environ 0,91 à plus de 0,98, avant de se stabiliser autour de 0,99. La précision sur les données de validation suit une tendance similaire et atteint également des valeurs élevées, proches de celles obtenues sur l'ensemble d'entraînement.

Cette convergence des deux courbes indique que le modèle généralise bien et ne présente pas de signe notable de surapprentissage.

Analyse de la Fonction de Perte

La loss d'entraînement diminue fortement au fil des époques, passant de 0,27 à moins de 0,02, tandis que celle de validation reste faible et relativement stable, autour de 0,04.

L'absence d'écart significatif entre les pertes et précisions des deux ensembles témoigne d'un apprentissage équilibré. Ces résultats traduisent une bonne performance du modèle avec une excellente capacité de généralisation.

Évaluation sur le Jeu de Test

Accuracy Test

99,2%

Précision exceptionnelle sur les données de test jamais vues pendant l'entraînement

Loss Test

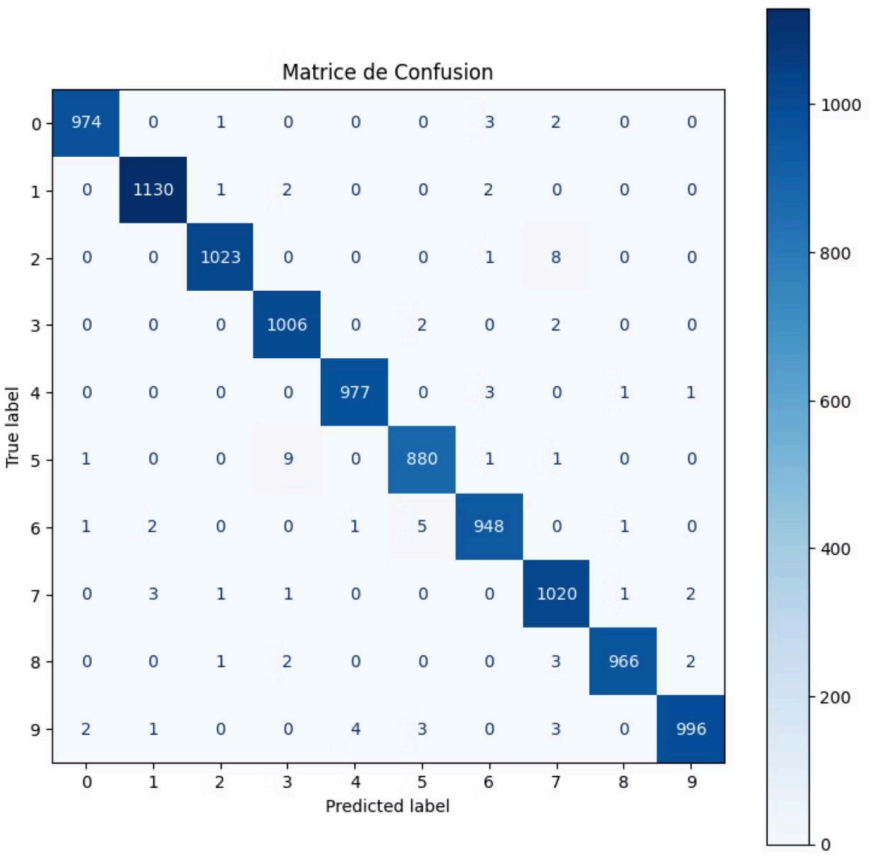
0,0355

Perte très faible confirmant la stabilité et la précision des prédictions

L'évaluation du modèle sur le jeu de test a permis d'obtenir une accuracy de 99,2% et une loss de 0,0355. Ces résultats confirment la très bonne capacité de généralisation du modèle, puisqu'ils sont cohérents avec les performances observées sur le jeu de validation (environ 99%).

Le faible taux de perte indique que les prédictions du réseau sont précises et stables. Ces performances témoignent de la pertinence de l'architecture choisie et de l'efficacité du processus d'apprentissage, tout en montrant que le modèle ne présente pas de surapprentissage significatif.

Matrice de Confusion et Analyse Détaillée



Interprétation des Résultats

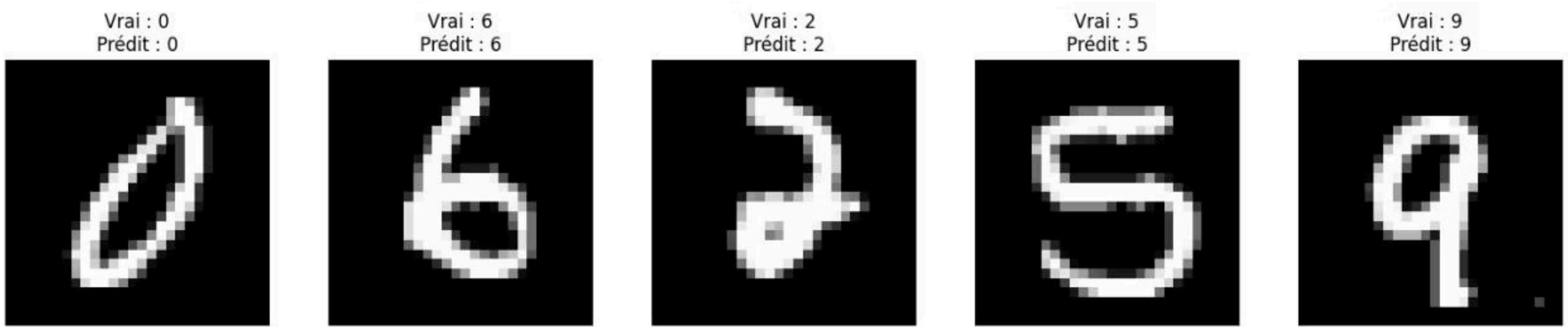
La matrice de confusion permet d'évaluer finement les performances du modèle. La grande majorité des prédictions se situent sur la diagonale principale, traduisant une excellente correspondance entre classes prédites et réelles.

Points clés :

- Pour chaque chiffre, le nombre d'exemples correctement classés est très élevé (>970/1000)
- Les erreurs sont rares et concernent des classes visuellement proches (4/9, 5/6)
- Excellente capacité de discrimination entre les différentes classes
- Confirmation de la forte précision globale et de la généralisation

Cette faible proportion d'erreurs témoigne d'une très bonne capacité du modèle à distinguer les différentes classes de chiffres manuscrits.

Validation Visuelle des Prédictions



Cette étape de validation visuelle permet de vérifier concrètement les performances du modèle en affichant un échantillon aléatoire d'images du jeu de test avec leurs étiquettes réelles et prédictions. Cette approche offre un contrôle qualitatif complémentaire aux indicateurs quantitatifs.

Objectif de la Validation

Observer comment le modèle se comporte sur des données jamais vues pendant l'entraînement et identifier d'éventuelles erreurs de classification.

Résultats Observés

Toutes les prédictions réalisées sur les images sélectionnées se sont révélées exactes, confirmant la fiabilité et la robustesse du modèle.

Conclusion

Contrôle qualitatif validant les excellentes performances quantitatives (accuracy, loss, matrice de confusion) du réseau de neurones convolutif.

Fichier Notebook Disponible

Le notebook Jupyter complet (.ipynb) contenant l'intégralité du code, des visualisations et des résultats détaillés est disponible pour reproduction et analyse approfondie de cette implémentation.

lien github : [saraodile1/MNIST-CNN]