# Reconnaissance de Chiffres MNIST avec CNN et Interface Gradio

## Description

Ce projet implémente un système de reconnaissance de chiffres manuscrits utilisant un réseau de neurones convolutionnel (CNN) entraîné sur le dataset MNIST. L'application inclut une interface web interactive développée avec Gradio qui permet de dessiner des chiffres et d'obtenir des prédictions en temps réel.

## Fonctionnalités

- **Modèle CNN** : Architecture de réseau de neurones convolutionnel optimisée pour la reconnaissance de chiffres
- **Interface interactive** : Zone de dessin pour tracer des chiffres à la main
- **Prédictions en temps réel** : Reconnaissance instantanée avec niveau de confiance
- **Analyse détaillée** : Probabilités pour chaque chiffre (0-9)

## Architecture du Modèle

Le modèle CNN comprend :

- 3 couches de convolution avec activation ReLU
- 2 couches de max pooling
- 1 couche dense cachée (64 neurones)
- 1 couche de sortie avec activation softmax (10 classes)

## Fichiers du Projet

- `mnist_model.py` : Définition de l'architecture du modèle CNN
- `train_mnist.py` : Script d'entraînement du modèle
- `gradio_app_simple.py` : Interface Gradio pour l'application web
- `mnist_cnn_model.h5` : Modèle entraîné sauvegardé
- `README.md` : Documentation du projet

# Installation et Utilisation

## Prérequis

```Bash
pip install tensorflow gradio numpy opencv-python pillow
```

## Entraînement du Modèle

```Bash
python3 train_mnist.py
```

## Lancement de l'Interface Web

```Bash
python3 gradio_app_simple.py
```

L'application sera accessible sur `http://localhost:7862`

# Utilisation de l'Interface

1. **Sélectionner l'outil de dessin** (icône pinceau)
2. **Dessiner un chiffre** de 0 à 9 dans la zone de dessin

3. **Cliquer sur "Prédire"** pour obtenir la reconnaissance
4. **Consulter les résultats** :
    - Chiffre prédit
    - Niveau de confiance en pourcentage
    - Probabilités détaillées pour chaque chiffre

## Performances du Modèle

Le modèle atteint une précision élevée sur le dataset de test MNIST grâce à :

- Architecture CNN adaptée aux images
- Prétraitement approprié des données
- Entraînement sur 60,000 images d'entraînement
- Validation sur 10,000 images de test

## Prétraitement des Images

L'application effectue automatiquement :

- Redimensionnement à 28x28 pixels
- Conversion en niveaux de gris
- Inversion des couleurs (fond noir, chiffre blanc)
- Normalisation des valeurs de pixels (0-1)

## Technologies Utilisées

- **TensorFlow/Keras** : Framework de deep learning
- **Gradio** : Interface web interactive

- **OpenCV** : Traitement d'images
- **NumPy** : Calculs numériques
- **PIL** : Manipulation d'images

## Auteur

Projet développé avec Manus AI pour la reconnaissance de chiffres manuscrits.

## Déploiement Permanent (pour l'utilisateur)

Pour déployer cette application de manière permanente, vous pouvez utiliser des plateformes comme Hugging Face Spaces ou Render. Voici les étapes générales pour un déploiement sur Hugging Face Spaces :

1. **Créer un nouveau Space** : Allez sur Hugging Face Spaces, connectez-vous et créez un nouveau Space. Choisissez le SDK `Gradio` .
2. **Cloner le dépôt** : Clonez le dépôt Git de votre nouveau Space sur votre machine locale.
3. **Ajouter les fichiers du projet** : Copiez tous les fichiers de ce projet ( `mnist_model.py` , `train_mnist.py` , `gradio_app_fixed.py` , `mnist_cnn_model.h5` , `requirements.txt` , `README.md` ) dans le dossier cloné.
4. **Renommer l'application Gradio** : Renommez `gradio_app_fixed.py` en `app.py` (c'est le nom par défaut attendu par Hugging Face Spaces pour les applications Gradio).
5. **Mettre à jour** `requirements.txt` : Assurez-vous que le fichier `requirements.txt` contient toutes les dépendances nécessaires ( `tensorflow` , `gradio` , `numpy` , `opencv-python` , `Pillow` ).
6. **Pousser les modifications** : Poussez tous les fichiers vers le dépôt Git de votre Space.

Bash

```
git add .
git commit -m "Initial commit of MNIST Gradio app"
```

```
git push
```

Hugging Face Spaces détectera automatiquement votre application Gradio et la déploiera. Le lien de votre application sera disponible sur la page de votre Space.

**Note importante** : Le modèle `mnist_cnn_model.h5` doit être présent dans le même répertoire que `app.py` pour que l'application puisse le charger correctement. Le dataset MNIST sera téléchargé automatiquement par TensorFlow lors de l'exécution de `train_mnist.py` ou lors du premier chargement du modèle si les données ne sont pas déjà présentes.