MNIST Handwritten Digit Classification with Artificial Neural Networks

Author: Daniel Muthama
Program: Data and AI 2025
Date: 15/7/2025

1. Introduction

This project demonstrates the implementation of an Artificial Neural Network (ANN) using TensorFlow/Keras to classify handwritten digits from the MNIST dataset. The MNIST dataset consists of 70,000 grayscale images (28×28 pixels) of digits (0–9), split into 60,000 training and 10,000 test samples.

Objectives:

✔ Preprocess image data for classification
✔ Design and train an ANN with input, hidden, and output layers
✔ Evaluate model performance using accuracy, confusion matrix, and classification report

2. Methodology

2.1 Data Loading & Exploration

- The dataset was loaded using `tensorflow.keras.datasets.mnist`.
- Training set: `(60000, 28, 28)`
- Test set: `(10000, 28, 28)`
- A sample of 9 images was visualized to verify correct labeling.

2.2 Data Preprocessing

- Normalization: Pixel values scaled to [0, 1] (`X_train / 255.0`).
- One-Hot Encoding: Labels converted to categorical format (`to_categorical`).

2.3 Model Architecture

A Sequential ANN was built with:

- Input Layer: `Flatten` (28×28 → 784 neurons)
- Hidden Layers:
  - `Dense(128, activation='relu')` + `Dropout(0.3)`
  - `Dense(64, activation='relu')` + `Dropout(0.3)`
- Output Layer: `Dense(10, activation='softmax')`

Total Parameters: 109,386

2.4 Training

- Optimizer: `adam`
- Loss Function: `categorical_crossentropy`
- Batch Size: `128`
- Epochs: `10`
- Validation Split: `10%`

Training Results:

- Training Accuracy: ~98%
- Validation Accuracy: ~97%

2.5 Evaluation

- Test Accuracy: ~97%
- **Confusion Matrix:**
  - High accuracy across all digits (0–9).
  - Minor misclassifications (e.g., 4 ↔ 9, 3 ↔ 5).
- **Classification Report:**
  - Precision, Recall, F1-Score consistently above 0.96 for all classes.

3. Results & Discussion

3.1 Performance Metrics

Metric	Value
Test Accuracy	97.2%
Training Loss	0.08
Validation Loss	0.12

3.2 Key Observations

✅ The model achieved high accuracy (97.2%) on unseen test data.

✅ Dropout layers (0.3) helped prevent overfitting.

✅ Misclassifications mostly occurred between visually similar digits (e.g., 4 ↔ 9).

3.3 Limitations & Improvements

- Limitation: Simple ANN may struggle with more complex variations.
- Improvement:
  - Use Convolutional Neural Networks (CNNs) for better feature extraction.
  - Apply data augmentation to improve generalization.

4. Conclusion

The ANN successfully classified handwritten digits with 97.2% accuracy. The project demonstrated:

✔ Effective data preprocessing

✔ Proper model design & training

✔ Comprehensive performance evaluation

The trained model was saved as `mnist_ann_model.keras` and verified for consistency upon reloading.

Appendix:

- Code Repository: [GitHub Link]

- Colab Link: [Colab Link]

- **References**:

  - TensorFlow Documentation

  - MNIST Dataset Official Page