Project Report

Title:

Handwritten Digit Recognition Using Deep Learning (MNIST Dataset)

1. Introduction:

In this project, we build a deep learning model to classify handwritten digits (0-9) using the MNIST dataset. This is a fundamental application of deep learning in the field of image classification. The project demonstrates how neural networks can be used to automatically recognize patterns in images.

2. Problem Statement:

The objective is to create an accurate model that can classify grayscale images of handwritten digits into the correct numeric classes (0-9).

3. Objectives:

- To explore deep learning techniques for image classification.
- To train a model using TensorFlow on the MNIST dataset.
- To visualize model performance and predictions.

4. Tools and Technologies Used:

- **Programming Language:** Python
- **Deep Learning Library:** TensorFlow with Keras
- **Platform:** Google Colab
- **Visualization:** Matplotlib

5. Dataset:

The MNIST dataset consists of 70,000 grayscale images of handwritten digits split as: - 60,000 images for training - 10,000 images for testing Each image is 28x28 pixels and belongs to one of 10 classes (0 to 9).

6. Methodology:

Step 1: Importing Libraries

Essential libraries such as TensorFlow, Matplotlib, and NumPy are imported.

Step 2: Loading the Dataset

Using TensorFlow's built-in datasets module, we load the MNIST dataset.

Step 3: Data Preprocessing

- Normalize pixel values to the range [0, 1].
- Visualize some sample images.

Step 4: Model Building

A simple neural network model is built using Keras Sequential API: - Flatten layer to reshape the input - Dense hidden layer with ReLU activation - Output layer with softmax activation

Step 5: Model Compilation

The model is compiled using: - Optimizer: Adam - Loss function: Sparse Categorical Crossentropy - Metrics: Accuracy

Step 6: Model Training

The model is trained for 5 epochs using both training and validation data.

Step 7: Model Evaluation

Evaluate model performance using test dataset to calculate final accuracy.

Step 8: Visualizations

Graphs of accuracy, validation accuracy, loss, and validation loss are plotted.

Step 9: Predictions

The trained model is used to make predictions on unseen test images. The predicted class is compared to the actual class.

Step 10: Saving the Model

The trained model is saved as `mnist_model.h5` for future use.

7. Results:

- The model achieved approximately 97% accuracy on the MNIST test set.
- Visualizations of training accuracy, validation accuracy, training loss, and validation loss were generated.
- Sample predictions matched the actual labels with high accuracy.

8. Applications:

- Handwritten character recognition systems.
- Automated postal code recognition.
- Bank check digitization.
- Real-time number recognition in various industries.

9. Limitations:

- The model is trained on a relatively clean and simple dataset.
- The approach may not work well on complex or distorted handwriting.

10. Future Work:

- Implement Convolutional Neural Networks (CNN) for better performance.
- Experiment with data augmentation techniques.
- Deploy the trained model as a web or mobile app for real-time predictions.

11. Conclusion:

This project successfully demonstrates how deep learning can be used for simple image classification tasks like handwritten digit recognition. The trained model achieves high accuracy and serves as a solid foundation for more advanced computer vision applications.

12. References:

- TensorFlow Official Documentation
- Keras Tutorials
- MNIST Dataset by Yann LeCun