Phase-1

RECOGNIZING HANDWRITTEN DIGITS WITH DEEP LEARNING FOR SMARTER AI APPLICATION

Student Name: REVETHA K

Register Number: 71772317141

Institution: Government College of Technology, Coimbatore

Department: Computer Science and Engineering

Date of Submission: 13.04.2025

1.Problem Statement

In many sectors, handwritten digits remain a common method of data input—such as prescriptions in healthcare, cheque amounts in banking, or roll numbers in education. Manually interpreting and digitizing these inputs is time-consuming and prone to human error. The lack of automation limits the integration of traditional handwritten data into modern digital systems. Hence, building an intelligent system that can accurately recognize handwritten digits is crucial for enhancing efficiency, accessibility, and automation in AI-powered applications across domains.

2.Objectives of the Project

Through this project, we aim to:

- *Build a deep learning model, specifically using Convolutional Neural Networks (CNNs), to recognize handwritten digits ranging from 0 to 9.*

- *Train and evaluate this model using both standard datasets and custom handwritten samples.*
- *Demonstrate the model's ability to integrate with real-world applications such as MedTrack (medicine tracking), cheque readers, and academic data entry tools.*
- *Create a simple interface that allows users to input handwritten digits and receive predictions in real time.*
- *Showcase the role of AI in bridging traditional handwritten input methods with modern digital systems.*

3.Scope of the Project

We Plan to Build:

- *A CNN-based digit classifier*
- *A preprocessing pipeline for input image standardization*
- *Real-world testing using custom digit samples*
- *A basic interface (CLI or GUI) for testing predictions*
- *Integration modules for domain-specific use cases*

Constraints:

- *Our current focus is on single-digit recognition (0–9)*
- *We are using the MNIST dataset along with a small custom dataset*
- *Implementation is limited to Python and associated frameworks*
- *Web/mobile deployment is optional and subject to time constraints*

4.Data Sources

*Primary Dataset**:** MNIST, a well-known public, static dataset containing 60,000 training and 10,000 testing images of handwritten digits which is availed from [Kaggle/UCI].*

Secondary Dataset: We are generating our own digit samples using scanned or photographed handwritten numbers for testing purposes. This helps us evaluate how well the model performs on real-world inputs beyond the MNIST dataset.

5.High-Level Methodology

- **Data Collection** – *We download the MNIST dataset from TensorFlow/Keras. We also collect handwritten digit samples using notebooks, scanners, or mobile phone cameras.*

- **Data Cleaning** – *All images are converted to grayscale, resized to 28x28 pixels, and normalized. We also remove any poor-quality or inconsistent samples from the custom dataset.*

- **Exploratory Data Analysis (EDA)** – *We use tools like Matplotlib and Seaborn to explore class distribution, pixel intensity patterns, and sample images. This helps us understand the characteristics of both the standard and custom datasets.*

- **Feature Engineering** – *We apply normalization and basic transformations. In some cases, we use data augmentation to increase the variety in training samples.*

- **Model Building –** *We implement a CNN using TensorFlow/Keras (or PyTorch) and experiment with multiple layers, optimizers, and activation functions to improve performance.*

- **Model Evaluation** – *We assess model accuracy and loss using training and validation data. We also evaluate the model on our custom dataset and visualize results using a confusion matrix.*

- **Visualization & Interpretation** – *We plot training progress (loss/accuracy graphs), display prediction results, and visualize EDA insights to interpret model performance.*

- **Deployment** – *We plan to build a simple CLI or web-based interface using Streamlit or Flask to allow users to upload/test handwritten digit images. Mobile deployment may be considered in the future using TensorFlow Lite.*

6.Tools and Technologies

- **Programming Language** – *Python*
- **Notebook/IDE** – *Pycharm/ Google Colab/ Jupyter Notebook/ VS Code*
- **Libraries**:
  - *Data Handling: NumPy, Pandas*
  - *Image Processing: OpenCV, PIL*
  - *Modeling: TensorFlow, Keras or PyTorch*
  - *Visualization: Matplotlib, Seaborn*
- **Optional Tools for Deployment** – Streamlit or Flask for building simple interfaces, TensorFlow Lite or ML Kit for future mobile deployment

7.Team Members and Roles

1. Gokula Lakshmi P – Model Building
2. Revetha K – Model Building
3. Janani S – Front End Development
4. Srinithija S – Back End Development