PHASE-2

RECOGNIZING HANDWRITTEN DIGITS]WITH DEEP LEARNING FOR SMARTER AI APPLICATIONS

- Student Name: [R.Pradhish]
- Register Number: [422623104701]
- Institution: [University College Of Engineering Panruti]
- Department: [Computer Science And Engineering]
- Date of Submission: [10.05.2025]
- Github Repository Link: [https://github.com/Pradhish654/Recognizing-handwritten-digits-with-deep-learning-for-smarter-AI-applications.git]

🧩1. PROBLEM STATEMENT:

✔ In many industries including finance, education, and logistics, there is a growing need to convert handwritten numeric data into machine-readable formats quickly and accurately. Manual transcription is inefficient and error-prone. This project aims to build a deep learning-based system that can automatically recognize handwritten digits (0–9) using Convolutional Neural Networks (CNNs). By training the model on the MNIST dataset, the system demonstrates real-time digit recognition capabilities with high accuracy, suitable for smart OCR applications.

✔ Problem Type: Classification (multi-class)

✔ Why it matters: Enables smarter, automated systems for processing handwritten forms, checks, exam sheets, and more, reducing errors and increasing efficiency.

🧠2. PROJECT OBJECTIVES:

- ✔ Develop a CNN model to classify handwritten digits (0–9) from image data.
- ✔ Achieve a classification accuracy of >98% on the MNIST test set.
- ✔ Ensure the model is generalizable and applicable in real-world OCR scenarios.
- ✔ Compare performance with traditional ML models like Logistic Regression and Random Forest.
- ✔ Provide visual explanations (e.g., confusion matrix, activation maps) to interpret model behavior.

📊3. FLOWCHART OF THE PROJECT WORKFLOW

Start

↓

Data Collection (MNIST Dataset)

↓

Data Preprocessing & Normalization

↓

Exploratory Data Analysis (EDA)

↓

Feature Engineering (if applicable)

↓

Model Building:

→ CNN

→ Logistic Regression (for comparison)

↓

Model Evaluation (Accuracy, Precision, Recall)

↓

Visualization of Results

↓

Conclusion & Deployment Ready

🧪4.DATA DESCRIPTION:

Dataset: MNIST Handwritten Digits Dataset

- ✔ Source: Available via Keras Datasets or Kaggle
- ✔ Type: Image (grayscale, 28x28 pixels)
- ✔ Records: 70,000 images (60,000 for training, 10,000 for testing)
- ✔ Features: Pixel intensity values (0–255)
- ✔ Static/Dynamic: Static
- ✔ Target Variable: Digit label (0 to 9)

📈5. DATA PREPROCESSING:

Normalize pixel values to range [0, 1] using x/255.0

- ✔ Reshape images to fit CNN input shape: (28, 28, 1)
- ✔ Convert labels to one-hot encoding using to_categorical()

✔ No missing values or outliers due to clean dataset

✔ Ensure data type consistency (float32 for pixel values)

🚀6. EXPLORATORY DATA ANALYSIS(EDA):

Univariate: Count plots of digit distribution (balanced dataset)

✔ Image Samples: Visualize sample digits for clarity

✔ Pixel Intensity Histogram: Understand grayscale distribution

✔ Insights:

- Dataset is balanced for all digit classes
- Images are uniform size and quality
- No missing or corrupt entries

🧱7. FEATURE ENGINEERING:

Not applicable for CNN model (feature learning is automatic)

✔ For traditional models:

- Flatten images to 1D (784 features)
- Apply PCA (optional) for dimensionality reduction

📦8.MODEL BUILDING:

✔ Models Implemented:

✔ Convolutional Neural Network (CNN):

- Layers: Conv2D → MaxPooling → Dropout → Flatten → Dense
- Activation: ReLU, Softmax
- Optimizer: Adam
- Metrics: Accuracy

✔ Logistic Regression (for baseline comparison):

- Flattened pixel features
- Evaluated with accuracy and confusion matrix

✔ Train-Test Split:

✔ 80:20 on training data (for validation)

✔ Use standard MNIST test set for final evaluation

✔ Metrics Used:

✔ Accuracy, Precision, Recall, F1-Score

📥9. VISUALIZATION OF RESULTS & MODEL INSIGHTS:

✔ Confusion Matrix: Shows classification performance across digits

✔ Accuracy & Loss Curves: Monitors training vs validation

✔ Sample Predictions: Visual comparison of correct and incorrect classifications

✔ Feature Importance (Logistic Model): Visualize pixel influence (optional)

🛠️10.TOOLS AND TECHNOLOGIES USED:

Programming Language: Python

✔ **IDE: Jupyter Notebook / Google Colab**

✔ **Libraries:**

- **mpy, pandas**
- **matplotlib, seaborn**
- **tensorflow.kerasnu**
- **scikit-learn**

11.TEAM MEMBERS AND CONTRIBUTIONS:

Name	Contribution
S.Sangdeena	Project Lead, Model Architecture Design (CNN), Training & Optimization
P.Vaishnavi	Data Preprocessing, Data Normalization, One-Hot Encoding
S.Jothiga	Exploratory Data Analysis (EDA), Data Visualization, Insight Generation
R.pradhish	Model Evaluation, Metrics Analysis, Confusion Matrix, ROC Curve
T.Aligesh	Documentation, GitHub Repository Management, Report Formatting and Submission