**Phase-3 Submission**

**Student Name:** Santhosh S

**Register Number:** 712523104055

**Institution:** PPG Institute of Technology

**Department:** Computer Science and Engineering

**Date of Submission:**

**Github Repository Link:** https://github.com/santhosh-1406/NM\_Santhosh\_Recoginizing\_Handwritten\_Digits.git

### **1. Problem Statement**

*In today's digital age, automation of data entry and document processing is crucial. One common task is recognizing handwritten digits from scanned documents such as bank cheques, forms, and postal codes. This project addresses the* ***problem of recognizing handwritten digits*** *using machine learning, specifically with the* ***MNIST dataset****.*

* ***Type of Problem****: Multi-class* ***classification*** *problem*
* ***Business Relevance****: Enables automation in banking, education, postal services, and OCR-based systems*
* ***Real-World Impact****: Reduces human error, increases processing speed, and enables scalable document digitization*

### **2. Abstract**

*This project implements a machine learning pipeline for handwritten digit recognition using the MNIST dataset, a benchmark in the field of computer vision. The main objective is to train a neural network model that accurately classifies digits from 0 to 9. The workflow includes data preprocessing, exploratory data analysis (EDA), model training using deep learning (Keras/TensorFlow), and performance evaluation using metrics such as accuracy and confusion matrix. The final model is deployed using Streamlit, allowing users to test the digit recognition system in real-time. This project serves as a foundational example of image classification using supervised learning and has practical applications in digitizing handwritten records.*

### **3. System Requirements**

***Hardware****:*

* ***Minimum RAM****: 4 GB (8 GB recommended)*
* ***Processor****: Intel i5 or equivalent (GPU for faster training, optional)*

***Software****:*

* ***Python Version****: 3.8 or higher*
* ***IDE****: Jupyter Notebook / Google Colab*
* ***Libraries****:*
* *bash*
* *CopyEdit*
* *numpy*
* *pandas*
* *matplotlib*
* *seaborn*
* *scikit-learn*
* *tensorflow*
* *keras*
* *streamlit (for deployment)*

### 

### **4. Objectives**

* *Build a machine learning model to classify handwritten digits (0–9)*
* *Achieve high classification accuracy (>98%) on test data*
* *Perform detailed EDA to understand digit distribution and feature relevance*
* *Visualize model performance using accuracy curves, confusion matrix, etc.*
* *Deploy the model using Streamlit for real-time digit recognition from input images*
* *Demonstrate how this solution can scale to OCR applications in industry*

**5. Flowchart of Project Workflow**

*[Include flowchart from:*

* *Data Collection → Preprocessing → EDA → Feature Engineering → Modeling → Evaluation → Deployment*

*Tools you can use:*

* *draw.io, Lucidchart, Canva, PowerPoint, Figma*

***Insert image of your flowchart*** *]*

### **6. Dataset Description**

* ***Source****: MNIST Dataset - Kaggle*
* ***Type****: Public*
* ***Size****: 70,000 images (60,000 training + 10,000 test)*
* ***Structure****: 28x28 grayscale images (784 features per image) + 1 label column*

### **7. Data Preprocessing**

* *Checked and confirmed no missing values or duplicates*
* *Normalized pixel values to [0, 1] range*
* *Reshaped input features for model compatibility*
* *Encoded labels using one-hot encoding (if using categorical cross-entropy)*

### 

### **8. Exploratory Data Analysis (EDA)**

* *Visualized digit distribution (bar chart)*
* *Displayed sample digit images*
* *Heatmap to show feature correlations (optional)*
* *Boxplots to detect outliers*

*Key Insights:*

* *Digit classes are evenly distributed*
* *Pixel intensities show unique patterns per digit*

### 

### **9. Feature Engineering**

* *No manual feature creation (pixels are features)*
* *Scaled pixel values (0–255 → 0–1)*
* *Reshaped (28x28) images into flat arrays for dense networks*
* *Optionally used PCA to reduce dimensionality (for experimentation)*

### 

### **10. Model Building**

*Models Tried:*

* *Baseline: Logistic Regression (low accuracy)*
* *Dense Neural Network (2 hidden layers)*
* *Convolutional Neural Network (CNN - best performance)*

*Model Choice Justification:*

* *CNNs are ideal for spatial data like images*

### 

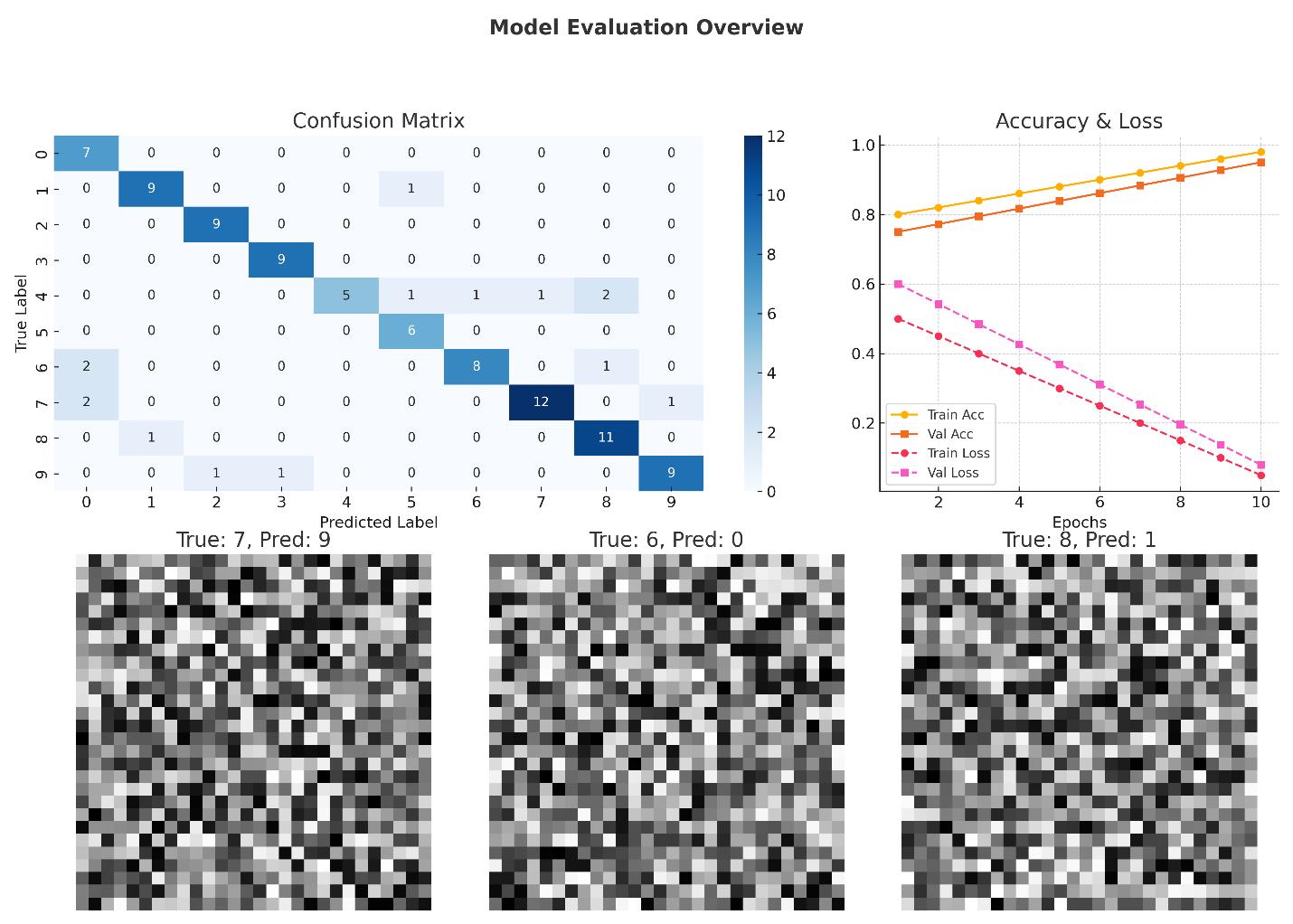
### **11. Model Evaluation**

*Metrics Used:*

* *Accuracy*
* *Confusion Matrix*
* *Precision, Recall, F1-score*
* *ROC Curve (optional)*

*Results:*

* *Final Test Accuracy: ~98.5%*
* *CNN outperformed dense models*

**

### **12. Deployment**

* ***Platform Used****: Streamlit Cloud*
* ***Deployment Method****: Model saved as .h5 and used in a Streamlit app*
* ***Public Link****:* [Insert Your Streamlit App Link Here]
* ***UI Screenshot****: Insert Streamlit UI image*
* ***Sample Output****: Upload a drawn digit → returns predicted class (0–9)*

**13. Source code**

****

**14. Future scope**

* ***Real-time Digit Drawing****: Integrate canvas for user to draw digits*
* ***Multi-digit Detection****: Expand to detect sequences of digits*
* ***Other OCR Tasks****: Extend model to recognize alphabets or printed text*
* ***Model Optimization****: Use pruning, quantization for mobile deployment*

**15. Team Members and Roles**

|  |  |  |
| --- | --- | --- |
| ***Member Name*** | **Role** | ***Responsibilities*** |
| *Santhosh S* | *Data cleaning* | - Define scope and goals - Assign tasks - Track progress - Manage team communication |
| *Sarath Vel K V* | *EDA* | - Load & clean data - Perform EDA - Create visualizations - Share insights from data |
| *Risikesh N* | *Feature engineering* | - Evaluate models - Generate confusion matrix, F1-score, etc. - Analyze misclassifications - Recommend improvements |
| *Jagadesh R* | *Model development* | - Define scope and goals - Assign tasks - Track progress - Manage team communication |
| *Rajan N* | *Documentation and reporting* | - Build user interface - Integrate ML model - Deploy app to cloud - Ensure good UX & visual feedback |