Phase-3

Student Name: M.R.Tameema Naazmi

Register Number: 712523121024

Institution: PPG Institute Of Technology

Department: Biomedical Engineering

Date of Submission: 16-05-2025

Github Repository Link:
https://github.com/Tameema14naazmi/NM_Deep-Learning-AI.git

1. Problem Statement:

Handwritten digit recognition is a critical task in computer vision, with applications ranging from automated form processing to postal code recognition. This project aims to develop a deep learning model capable of accurately identifying digits from images of handwritten numbers.

2. Abstract

This project presents a deep learning-based approach to recognizing handwritten digits using the MNIST dataset. A Convolutional Neural Network is trained on image data to achieve high accuracy in classification. The project involves data preprocessing, model training, evaluation, & deployment, culminating in an end-to-end AI solution .

3. System Requirements

Hardware:

- ➢ Minimum 8GB RAM
- ➢ GPU (NVIDIA preferred) for faster training (optional)

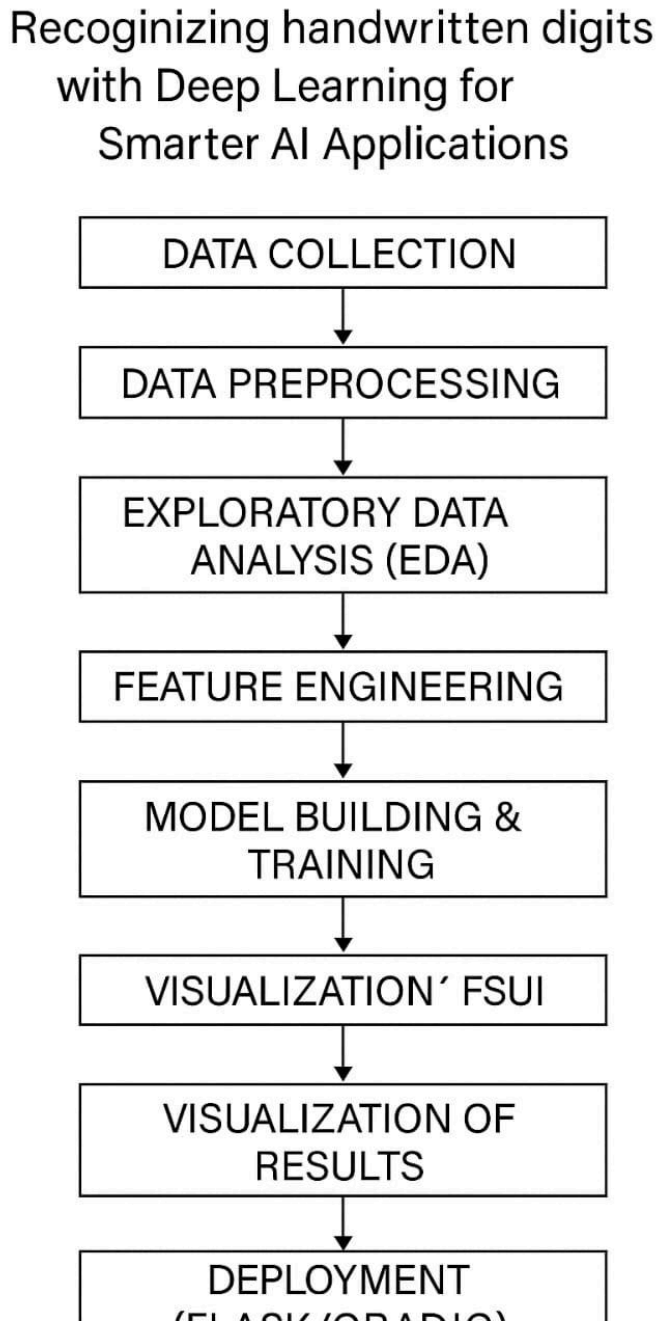
Software:

- ➢ Python 3.x
- ➢ Jupyter Notebook / VS Code
- ➢ TensorFlow / Keras / PyTorch
- ➢ NumPy, Pandas, Matplotlib, Seaborn
- ➢ Flask or Streamlit for deployment

4. Objectives

To build and deploy a deep learning model that can accurately classify handwritten digits from 0 to 9, thereby demonstrating the application of neural networks in real-world computer vision tasks.

5. Flowchart of Project workflow



6. Dataset Description

Name: MNIST (Modified National Institute of Standards and Technology)

Source: https://www.kaggle.com/datasets/oddrationale/mnist-in-csv

Details:

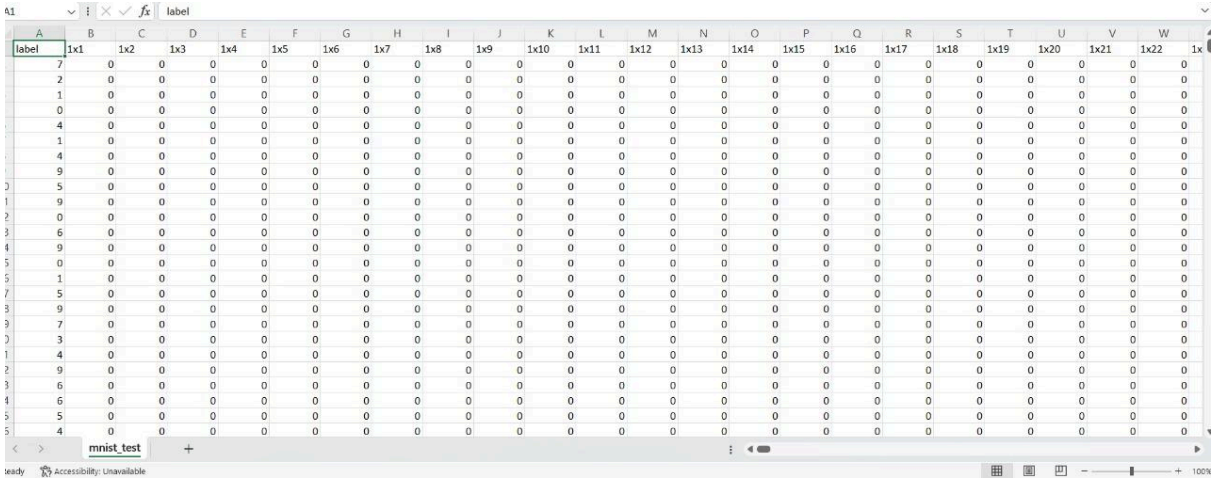
60,000 training images

10,000 test images

Grayscale, 28x28 pixels

Labels: Digits (0 to 9)

SAMPLE DATASET:



7. Data Preprocessing

Normalization of pixel values (0–255 to 0–1)

Reshaping images for CNN input

One-hot encoding of labels

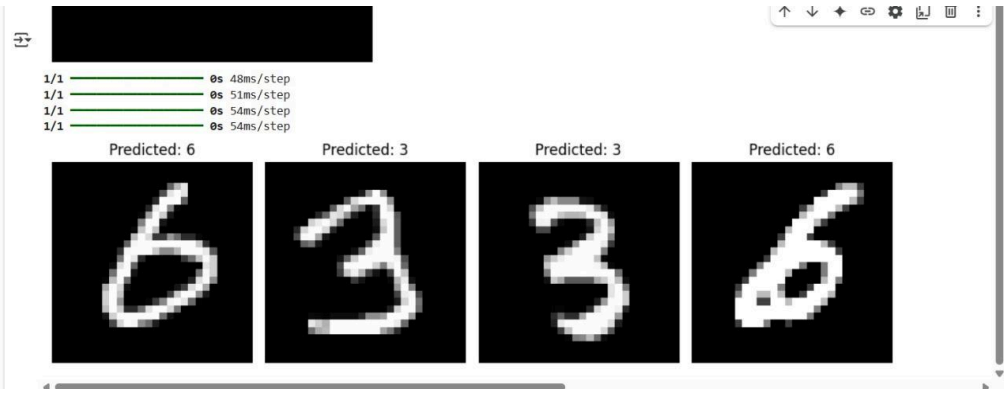
Splitting into training and validation sets

8. Exploratory Data Analysis(EDA)

Visualizing digit distribution

Displaying sample images

Checking image quality and balance



9.Feature Engineering

Adding dropout layers to prevent overfitting

Image augmentation (optional)

Converting labels to categorical format & Padding or resizing if needed

10. Model Building

Use a CNN architecture:

Conv2D → MaxPooling → Flatten → Dense

Final layer: 10 neurons (Softmax activation)

Compile with Adam optimizer and categorical cross-entropy loss

Train for 10–20 epochs

11. ModeArchitecture:

Metrics: Accuracy, Precision, Recall, F1 Score

Confusion Matrix

Validation Accuracy vs Training Accuracy

Loss curves

12. Deployment

Using Flask/Streamlit to create a web interface

Uploading an image to predict digits

Hosting options: Heroku, Render, or local server

13. Source code

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from tensorflow.keras.utils import to_categorical

from tensorflow.keras.models import Sequential


from tensorflow.keras.layers import Dense, Flatten

from tensorflow.keras.optimizers import Adam


# Load dataset

train_df = pd.read_csv("mnist_train.csv"
```

```
test_df = pd.read_csv("mnist_test.csv")


# Split features and labels
X_train = train_df.iloc[:, 1:].values
y_train = train_df.iloc[:, 0].values
X_test = test_df.iloc[:, 1:].values
y_test = test_df.iloc[:, 0].values


# Normalize the pixel values (0–255 to 0–1)
X_train = X_train / 255.0
X_test = X_test / 255.0


# One-hot encode labels
y_train = to_categorical(y_train, 10)
y_test = to_categorical(y_test, 10)
```

```
# Build a simple neural network
model = Sequential([
    Dense(128, activation='relu', input_shape=(784,)),
    Dense(64, activation='relu'),
    Dense(10, activation='softmax')  # 10 output classes
])
# Compile the model
model.compile(optimizer=Adam(), loss='categorical_crossentropy',
metrics=['accuracy'])


# Train the model
model.fit(X_train, y_train, epochs=10, batch_size=128, validation_split=0.1)


# Evaluate on test data
test_loss, test_acc = model.evaluate(X_test, y_test)
print(f"\nTest accuracy: {test_acc:.4f}")
```

#

```
Show 4 sample predictions
num_samples = 4
sample_indices = np.random.choice(len(X_test), num_samples, replace=False)


plt.figure(figsize=(10, 3))
for i, idx in enumerate(sample_indices):
    img = X_test[idx].reshape(28, 28)


    pred = model.predict(np.expand_dims(X_test[idx], axis=0))
    label = np.argmax(pred)


    plt.subplot(1, num_samples, i + 1)
    plt.imshow(img, cmap='gray')
    plt.title(f"Predicted: {label}")
    plt.axis('off')
plt.tight_layout()
plt.show()
```

14. Future scope

Expand to multi-digit recognition

Integration with real-time camera input

Use more complex datasets like EMNIST or USPS

Deployment on mobile devices & Improving accuracy with advanced model

13. Team Members and Roles

Team Leader: M.R. Tameema Naazmi

Responsible for overall project management, coordination among team members, setting milestones, and ensuring timely delivery.

M. Ashwini – Data analyst

Handles data preprocessing, visualization, and analysis. Prepares the dataset for training and testing.

K. Monisha – Deep Learning Engineer

Designs and develops the Convolutional Neural Network (CNN). Trains and optimizes the model for high accuracy.

G. Sharmila Devi – Software Developer

Develops the user interface and deploys the trained model using a web or mobile platform.

Pratiksha – QA Analyst

Tests the model's performance and accuracy…