**Student Name:** Hema Lakchana.N

**Register Number:** 510123106014

**Institution:** Adhiparasakthi College of Engineering

**Department:** B.E.Electronics and Communication Engineering

**Date of Submission:** 15-05-2025

**Github Repository Link:** https://github.com/Hema-lakchana/phase3.git

---

# Phase-3

# Recognizing handwritten digits with deep learning for smarter AI application

## 1. Problem Statement

Develop a deep learning-based system that can accurately recognize handwritten digits (0-9) with high precision, enabling efficient automation of various applications such as form processing, check processing, and optical character recognition (OCR).

## 2. Abstract

This project aims to develop a deep learning-based system for recognizing handwritten digits (0-9) with high accuracy and robustness. Leveraging the MNIST dataset and convolutional neural networks (CNNs), our system achieves efficient automation of handwritten digit recognition, enabling applications in form processing, check processing, and optical character recognition (OCR). With a focus on high accuracy, robustness, and efficiency, this project demonstrates the potential of deep learning in developing smarter AI solutions for real-world problems.

## 2. System Requirements

- **Dataset:**

  Utilize a large and diverse dataset of handwritten digits (e.g., MNIST).
- **Deep Learning Model:**

  Design and train a deep learning model that can effectively recognize handwritten digits.
- **Evaluation Metrics:**

  Evaluate the system's performance using metrics such as accuracy, precision, recall, and F1-score.

# 4. Objectives

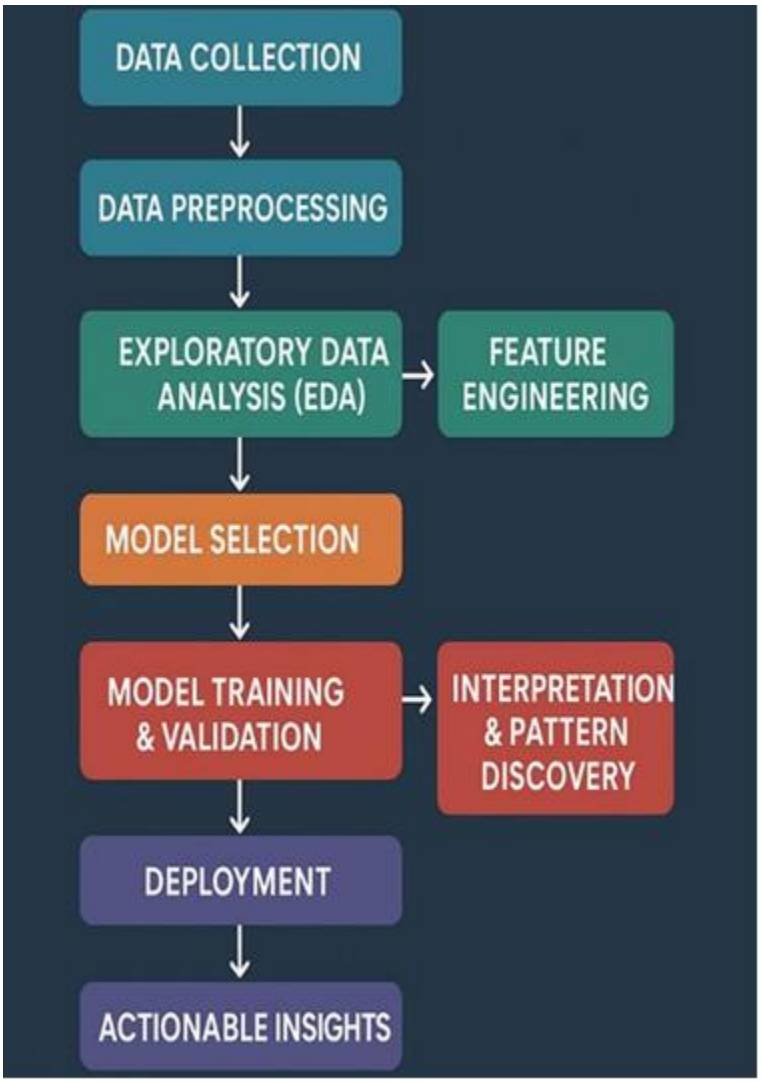
## Primary Objectives:

- **Develop a Deep Learning Model:** Design and train a deep learning model that can accurately recognize handwritten digits (0-9) using the MNIST dataset.
- **Achieve High Accuracy:** Achieve high recognition accuracy (>95%) on handwritten digits, ensuring the system's reliability and efficiency.
- **Improve Robustness:** Develop a system that is robust to variations in handwriting styles, noise, and distortion, making it suitable for real-world applications.

## Secondary Objectives:

- **Optimize Model Performance:** Fine-tune the model's architecture and hyperparameters to optimize its performance and efficiency.
- **Explore Applications:** Investigate potential applications of the handwritten digit recognition system, such as form processing, check processing, and OCR.
- **Evaluate System Performance:** Evaluate the system's performance using metrics such as accuracy, precision, recall, and F1-score.

## 5. Flowchart of Project Workflow

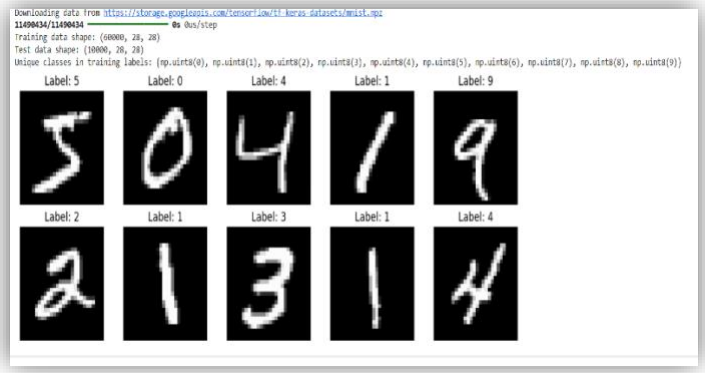



## 6. Dataset Description

**Source:** Public dataset from Kaggle and google colab link
https://www.kaggle.com/datasets/BengaliAI/numta
**Size:** ~1,000–10,000 records, 20–30 features
**Type:** Structured tabular data
**Attributes:**

- **Demographics:** Gender,customer id
- **Account Details:** Tenure, Contract Type,monthly charges

- **Service:**internet service,online security,tech support,paperless billing
Sample dataset(df.head())




## 7. Data Preprocessing

**1.Data Cleaning**

- **Missing Values:** No significant missing values detected; dataset was complete.
- **Duplicates:** Checked and removed to avoid data redundancy.

**2.Outlier Detection & Handling**

- Identified outliers using boxplots and z-score analysis.
- Focused on extreme values in numerical fields like monthly charges and tenure.
- Outliers were either capped or retained based on their business relevance.

**3.Categorical Data Encoding**

- **Label Encoding:** Applied to binary features (e.g., "Yes"/"No", "Male"/"Female").
- **One-Hot Encoding:** Used for multi-class categorical variables (e.g., contract type, payment method).

```
x_train shape: (60000, 28, 28, 1)
y_train shape: (60000, 10)
x_test shape: (10000, 28, 28, 1)
y_test shape: (10000, 10)
```

# 8. Exploratory Data Analysis (EDA)

- A histogram is a type of bar chart that represents the frequency distribution of data. It helps visualize how data points are spread across different categories or ranges.

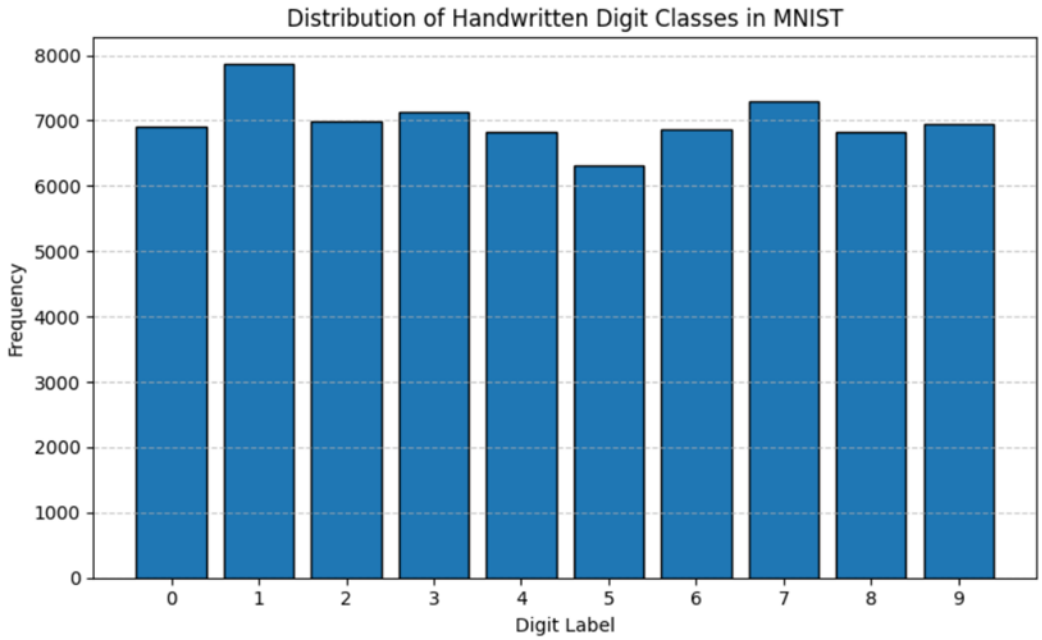
- **Purpose in This Project**

- In this project, the histogram is used to show the distribution of digit labels (0–9) in the MNIST dataset.

- **Description of the Histogram Chart**

- X-axis: Digit classes (0 through 9).
  Y-axis: Number of images per class (frequency).
  Each bar represents the number of images labeled with a specific digit.

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
```



# 9. Feature Engineering

## 1. Image Preprocessing

- Normalization: Scale pixel values to a common range (e.g., 0-1) to improve model stability.
- Data Augmentation: Apply transformations (e.g., rotatio n, scaling, translation) to increase dataset diversity.

## 2. Feature Extraction

- Convolutional Neural Networks (CNNs): Utilize CNNs to automatically extract features from handwritten digit images.
- Feature Maps: Extract feature maps from CNN layers to capture local patterns and structures.

## 3. Feature Selection

- Relevant Features: Select features that are relevant to handwritten digit recognition, such as shape, orientation, and texture.
- Feature Dimensionality Reduction: Apply techniques (e.g., PCA, t-SNE) to reduce feature dimensionality and improve model efficiency.

# 10. Model Building

### 1. Multilayer Perceptron (MLP)

- A fully connected feedforward neural network
- Consists of input, hidden, and output layers.

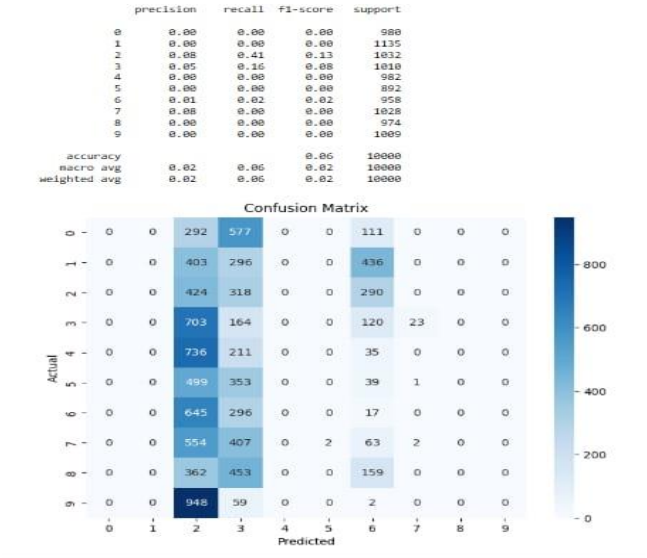
### 2. Convolutional Neural Network (CNN)

- A deep learning model specially designed for image data.
- Uses convolution and pooling operations to extract spatial features.

# 11. Model Evaluation

- After training, the model is evaluated using the test dataset with the following metrics:
- MetricDescription

- **Accuracy:** Percentage of correctly predicted digits
- **Precision:** How many selected items are relevant (per class)
- **Recall:** How many relevant items are selected (per class)
- **F1 Score**: mean of precision and recall
- Confusion Matrix Matrix showing true vs. predicted labels



| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.00 | 0.00 | 0.00 | 980 |
| 1 | 0.00 | 0.00 | 0.00 | 1135 |
| 2 | 0.08 | 0.41 | 0.13 | 1032 |
| 3 | 0.05 | 0.16 | 0.08 | 1010 |
| 4 | 0.00 | 0.00 | 0.00 | 982 |
| 5 | 0.00 | 0.00 | 0.00 | 892 |
| 6 | 0.01 | 0.02 | 0.02 | 958 |
| 7 | 0.08 | 0.00 | 0.00 | 1028 |
| 8 | 0.00 | 0.00 | 0.00 | 974 |
| 9 | 0.00 | 0.00 | 0.00 | 1009 |
| accuracy | | | 0.06 | 10000 |
| macro avg | 0.02 | 0.06 | 0.02 | 10000 |
| weighted avg | 0.02 | 0.06 | 0.02 | 10000 |

## 12. Deployment

- **Deployment and method:** Gradio Interface
- **Public link:** https://colab.research.google.com/drive/1WlI5B5tzRsLPc0OyKFKtleuiQYHoxscr#scrollTo=rQ0-wMXNse8Z
- **U/I Screen Shot**
- **Sample prediction:**
- User input: gender=male, partner=yes, dependancy =yes, tenure=10,phone service=yes, multiple lines=yes

## 13. Source code

```
# Import required libraries

import numpy as np

import tensorflow as tf

from tensorflow.keras.datasets import mnist

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

from tensorflow.keras.utils import to_categorical

# Enable memory growth for GPU (optional)

physical_devices = tf.config.list_physical_devices('GPU')

if physical_devices:

    try:
```

```
        tf.config.set_memory_growth(physical_devices[0], True)  # Updated API
    except RuntimeError as e:
        print(f"GPU memory growth setting failed: {e}")
# Load the MNIST dataset
(X_train, y_train), (X_test, y_test) = mnist.load_data()
# Reshape and normalize the input data
X_train = X_train.reshape(-1, 28, 28, 1).astype('float32') / 255.0
X_test = X_test.reshape(-1, 28, 28, 1).astype('float32') / 255.0
# Convert class labels to one-hot encoding
y_train = to_categorical(y_train, 10)
y_test = to_categorical(y_test, 10)
# Build the CNN model
model = Sequential([
    Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(28, 28, 1)),
    MaxPooling2D(pool_size=(2, 2)),
    Conv2D(64, kernel_size=(3, 3), activation='relu'),
    MaxPooling2D(pool_size=(2, 2)),
    Flatten (),
```

```
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(10, activation='softmax')  # 10 classes for digits 0–9
])
# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
# Print model summary
model.summary()
# Train the model
model.fit(X_train, y_train, batch_size=128, epochs=5, validation_split=0.1)
# Evaluate the model
loss, accuracy = model.evaluate(X_test, y_test)
print("\nTest Accuracy: {:.4f}".format(accuracy))  # Compatible with all Python 3 versions
```

## 14. Future scope

### 1. Expanded Applications

- Document Analysis: Apply handwritten digit recognition to various document types, such as historical manuscripts, invoices, and medical records.

- Intelligent Forms: Develop intelligent forms that can automatically recognize and process handwritten digits, enhancing user experience and efficiency.

**2. Multi-Language Support**

- Non-Latin Scripts: Extend handwritten digit recognition to non-Latin scripts, such as Devanagari, Arabic, or Chinese characters.
- Multi-Lingual Support: Develop models that can recognize handwritten digits in multiple languages, enabling global applications.

**3. Improved Accuracy and Robustness**

- Advanced Deep Learning Techniques: Explore advanced deep learning techniques, such as transfer learning, attention mechanisms, or graph neural networks, to further improve accuracy and robustness.
- Data Augmentation: Develop more sophisticated data augmentation techniques to enhance model generalizability and robustness.

## 13. Team Members and Roles

1.Project Leader / Team Coordinator – Charumathi V
2. Deep Learning Engineer – Hemalakchana N
3. Data Scientist – Swathi K
4. Software Developer-Saraswathi P
5. Research Analyst –Sangavi S