# MIST HANDWRITTEN DIGIT CLASSIFICATION

Low Level Design

Santhosh Kumar K S

OCTOMER 16, 2023
LETSGROWMORE

# 1. INTRODUCTION

## What is Low Level Design Document?

Low-level design refers to the process of specifying and defining the detailed design of a software system. This Low level Design focuses on the implementation details of a system and is concerned with how the system will be built and how it will function at a detailed level. It provides the foundation for high-level design, which defines a system's overall architecture and design.

## Scope of Low Level Design Document?

Low-level design (LLD) is a component-level design process that follows a step-by step refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms.

# 2. PROJECT DESCRIPTION

## PROBLEM STATEMENT

Develop a system that classifies the input image into digits.

## PROPOSED SOLUTION

Machine learning is a field in computer science aiming to imitate the human learning process. Deep Learning is a branch of machine learning where deep learning system imitates the biological neuron. Here we will develop Deep Learning models to classify the handwritten digit image.

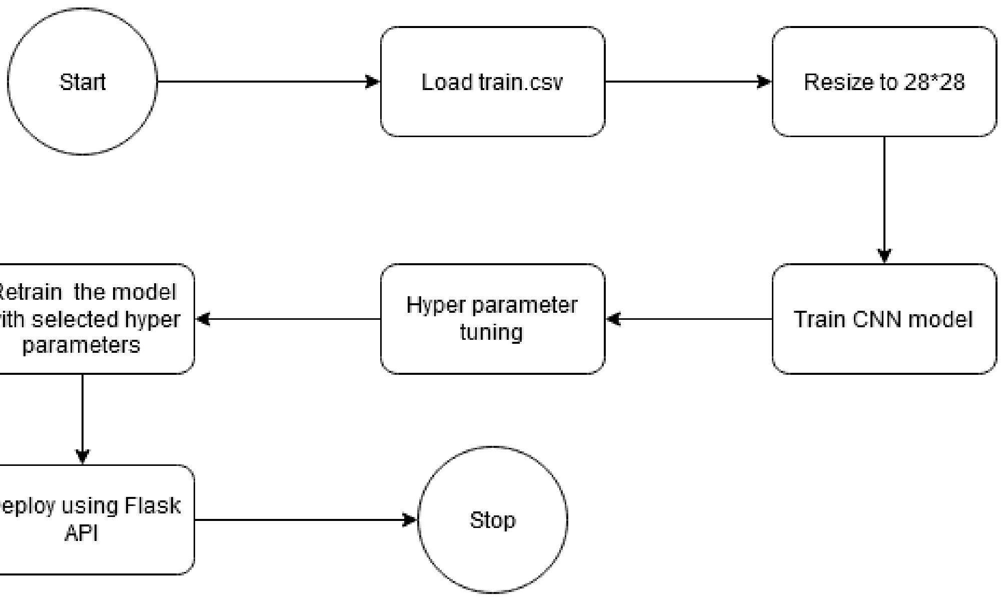
## DATA INFORMATION

The dataset was taken from Kaggle. This dataset is in CSV format. Each row contains 784 values, 1 for label and 784 values for image. There are 10 labels starting from 0 to 9.

## TOOLS USED

Python programming language and frameworks such as Numpy, OpenCV is used for image processing and Tensorflow library is used for model building. Flask is used to deploy trained model.

## 3. ARCHITECTURE



## 4. Architecture Description.

### Image Resize

In this project dataset is in the form of CSV format each row contains 785 values, 1 for label and remaining values indicate image. So we resize from (1,784) to (28*28).

.

### Train Test Split and Train the Models

In this we split the train and test will in the ratio of 80:20. And we trained CNN.

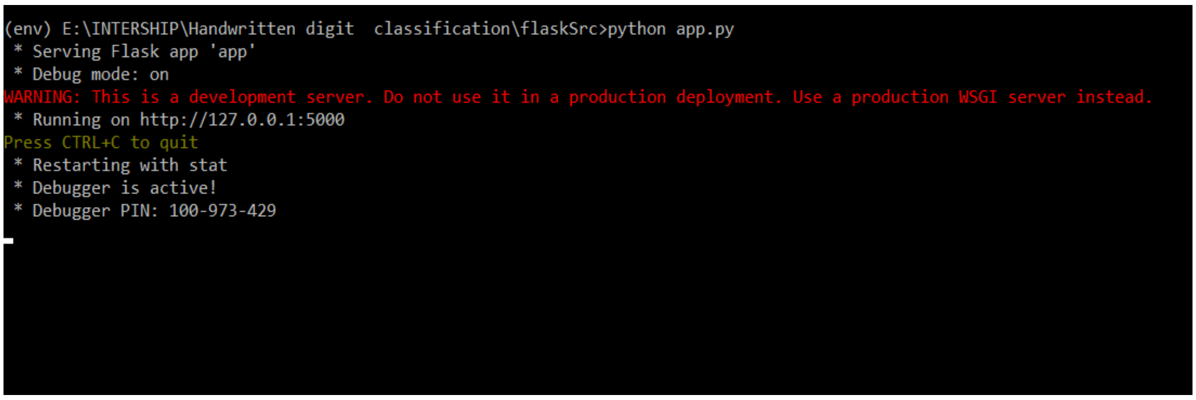
### Hyper Parameter Tuning and Train the model with best hyper parameter

We tuned hyper parameter for batch size, optimizers, learning rate.
We used randomized search for choosing appropriate hyper parameters. And trained the model with best hyper parameters.

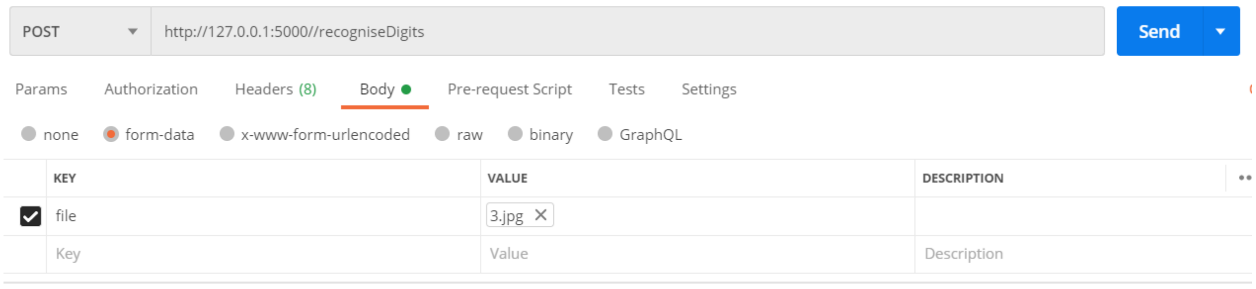
### Deploying using Flask

We designed UI and deployed model using Flask framework.

## 5.Screenshots



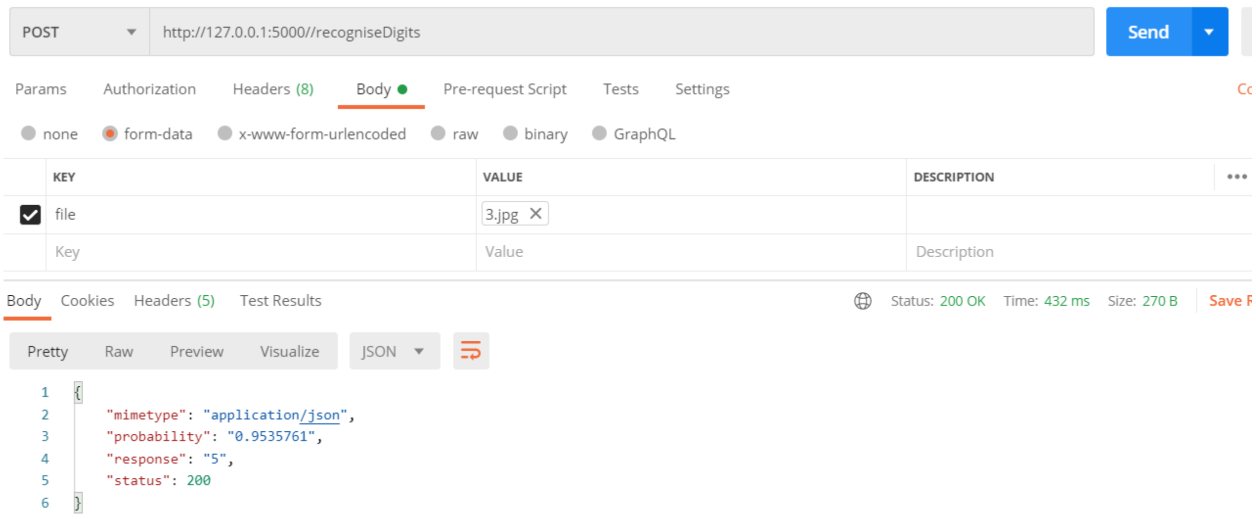
```
(env) E:\INTERSHIP\Handwritten digit  classification\flaskSrc>python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 100-973-429
```



POST http://127.0.0.1:5000//recogniseDigits

Body: form-data

| KEY | VALUE | DESCRIPTION |
|---|---|---|
| file | 3.jpg | |
| Key | Value | Description |



POST http://127.0.0.1:5000//recogniseDigits

Body: form-data

| KEY | VALUE | DESCRIPTION |
|---|---|---|
| file | 3.jpg | |
| Key | Value | Description |

Status: 200 OK  Time: 432 ms  Size: 270 B

```
{
    "mimetype": "application/json",
    "probability": "0.9535761",
    "response": "5",
    "status": 200
}
```