PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM

submitted by

PNT2022TMID53497

Pratheeba R - 2127190801056

Sai Raksha V - 2127190801067

Sandhiya R - 2127190801070

Suriya R -2127190801089

1. INTRODUCTION

1.1 Project Overview

Handwriting recognition is one of the compelling research works going on because every individual in this world has their own style of writing. It is the capability of the computer to identify and understand handwritten digits or characters automatically. Because of the progress in the field of science and technology, everything is being digitalized to reduce human effort. Hence, there comes a need for handwritten digit recognition in many real-time applications. MNIST data set is widely used for this recognition process and it has 70000 handwritten digits. We use Artificial neural networks to train these images and build a deep learning model. Web application is created where the user can upload an image of a handwritten digit. this image is analyzed by the model and the detected result is returned on to UI.

1.2 Purpose

The task of handwritten digit recognition, using a classifier, has great importance and use such as – online handwriting recognition on computer tablets, recognize zip codes on mail for postal mail sorting, processing bank check amounts, numeric entries in forms filled up by hand (for example - tax forms) and so on.

2. LITERATURE SURVEY

2.1 Existing problem

Who would face this kind of problem?	People using Handwritten Digit Recognition in postal mail sorting, bank check processing, form data entry etc.
What are the boundaries of the problem?	Recognition of handwritten digits using Slope Detail (SD) features based on the shape analysis of the digit image and extract slant or slope information. Original documents may have poor quality ad paper deteriorates easily.
What is this issue?	Failure of Handwritten Digit Recognition would lead to financial losses in case of bank check processing, location misidentification in case of postal mail sorting etc.

When does this issue occur?	The handwritten digits are not always of the same size, width, orientation and justified to margins as they differ from writing of person to person, so the general issue would be while **classifying the digits** due to the similarity between digits such as 1 and 7, 5 and 6, 3 and 8, 2 and 5, 2 and 7, etc.
Where is this issue manifesting?	The main problem lies within the ability on developing an efficient algorithm that can recognize handwritten digits, which is submitted by users by the way of a scanner, tablet, and other digital devices.
Why is it necessary to fix the problem?	It becomes very essential to fix this problem to prevent huge losses incurred in various domains which makes use of Handwritten Digit Recognition System.

2.2 References

				Finding
1	Handwritten Digit Recognization Using Convolutional Neural Network	Tarun Dubey , Shubham Patil, Anurag Singh	2022	Comparison of the execution time of the algorithms, increasing the number of epochs without changing the configuration of the algorithm.
2	Handwritten Digit Recognition Using Deep Learning	Surya S, Vismitha N	2021	The proposed system involves constructing the convolutional neural network with appropriate filters that help recognize right features and identify what patterns map to which character.
3	Handwritten digit recognition system using Machine learning	Apaar Chadha, Gaurav Yadav, Keshav Ahlawat	2022	Accuracy of these traditional CNNs can be improved even more by removing the ensemble features and fine tuning the hyper parameters of the pure CNN architecture. This will also reduce the computational complexities and overall cost of implementing the model.
4	Handwritten Digit Recognition using Machine and Deep Learning Algorithms	Ritik Dixit, Rishika Kushwah, Samay Pashine	2021	The handwritten digit recognition is performed using Support Vector Machines (SVM), Multi-Layer

				Perceptron (MLP) and Convolution Neural Network (CNN) models.
5	HandWritten Digit Recognition	Jyoti Shinde, Chaitali Rajput, Prof. Mr Mrunal Shidore, Prof. Milind Rane	2019	Neural network approach is used where in the machine will learn on itself by gaining experiences and the accuracy will increase based upon the experience it gains. The dataset was trained using feed forward neural network algorithm.
6	Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN)	Savita Ahlawat , Amit Choudhary , Anand Nayyar , Saurabh Singh and Byungun Yoon	2020	A CNN architecture is proposed in order to achieve accuracy even better than that of ensemble architectures, along with reduced operational complexity and cost.
7	A Survey on using Neural Network based Algorithms for Hand Written Digit Recognition	Muhammad Ramzan, Shahid Mehmood Awan,Hikmat Ullah Khan, Waseem Akhtar, Ammara Zamir, Mahwish Ilyas	2019	This review is novel as it is focused on HWDR and also it only discusses the application of Neural Network (NN) and its modified algorithms..It presents a Scientometric analysis of HWDR which presents top journals and sources of research content in this research domain.
8	Handwritten Digit Recognition Using Various Machine Learning Algorithms and Models	Pranit Patil, Bhupinder Kaur	2020	Illustration of various Machine learning algorithms such as Support Vector Machine, Convolutional Neural Network, Quantum Computing, K-Nearest Neighbor Algorithm, Deep Learning used in Recognition technique.
9	Handwritten Digit Recognition using Machine Learning Algorithms	S M Shamim, Mohammad Badrul Alam Miah, Angona Sarker, Masud Rana & Abdullah Al Jobair	2019	An approach to off-line handwritten digit recognition based on different machine learning techniques namely, Multilayer Perceptron, Support Vector Machine, Naïve Bayes, Bayes Net, Random Forest, J48 and Random Tree have been used for the recognition of digits.
10	Comparison Study of Handwritten Digit Recognition using Artificial Neural Network and Convolutional Neural Network	Sonia Flora, Anju Kakkad	2019	Comparing the classification accuracy of handwritten digit using artificial neural network and using state of the art deep learning model i.e. convolutional neural network

2.3 Problem Statement Definition

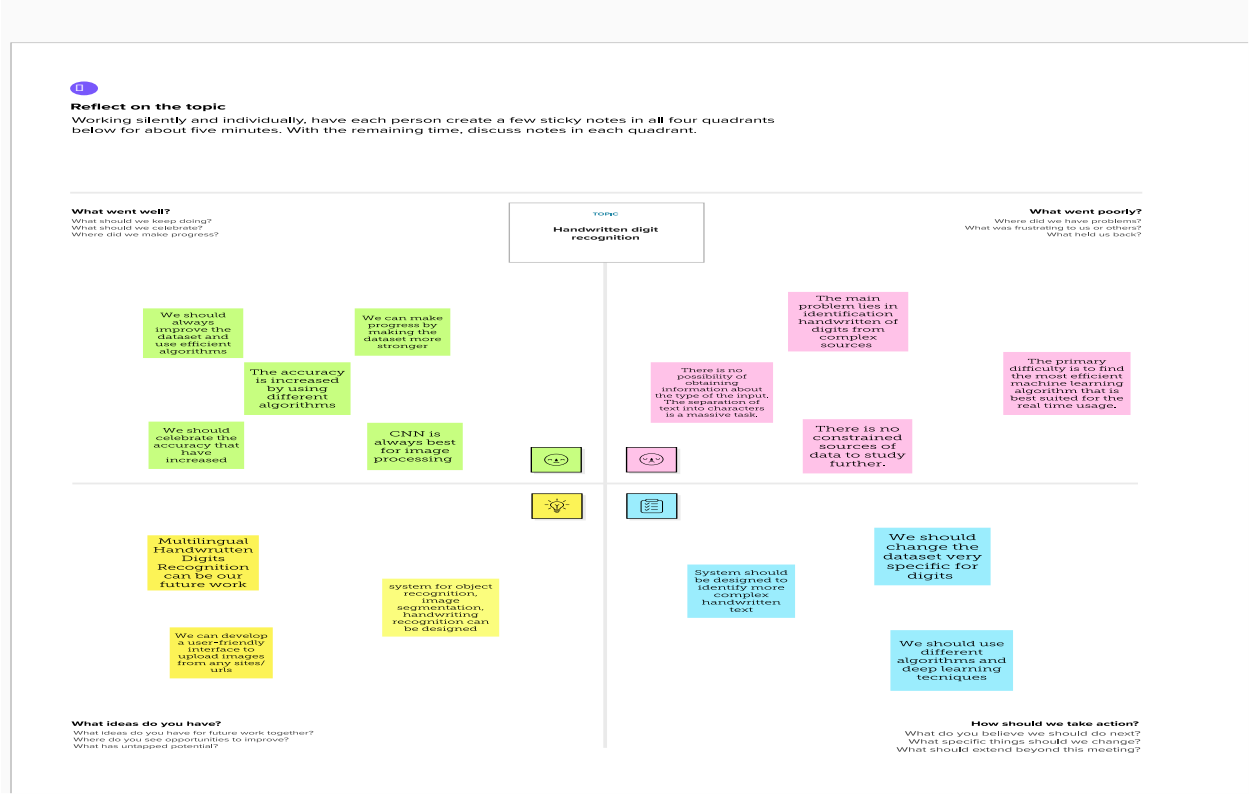
The **handwritten digit recognition** is the capability of computer applications to recognize the human handwritten digits. It is the hard task for the machine because handwritten digits are not perfect and can made many different shapes and sizes. The handwritten digit recognition is a way to tackle this problem which cause the images of digit and recognizes the digit present in the image.

Following are the constraints faced when computers approach to recognize handwritten digits:

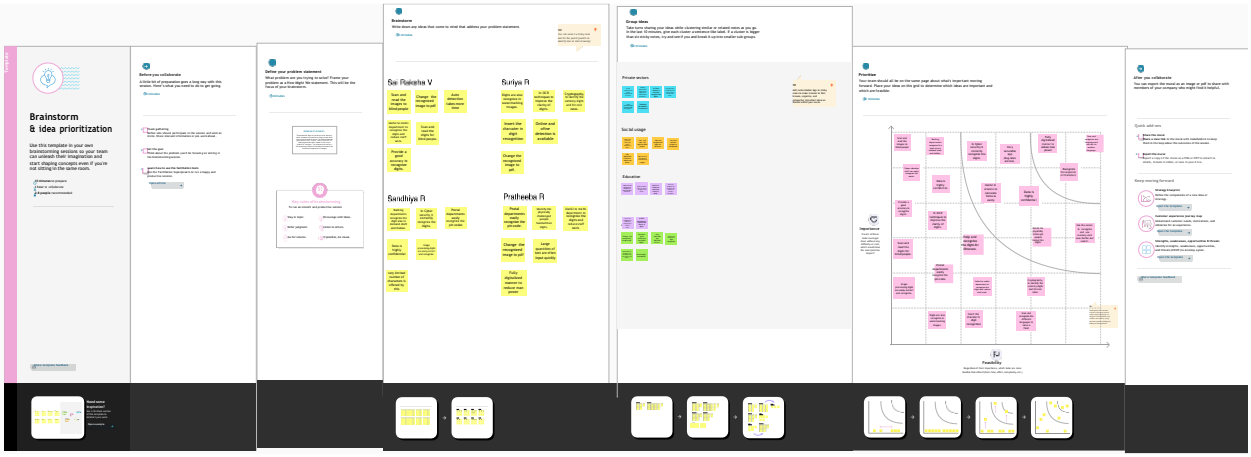
- The Handwritten digits are not always of the same size, width, orientation and justified to margins as they differ from writing of person to person.
- The similarity between digits such as 1 and 7, 5 and 6, 3 and 8, 2 and 7 etc. So, classifying between these numbers is also a major problem for computers.
- The uniqueness and variety in the handwriting of different individuals also influence the formation and appearance of the digits.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming



3.3 Proposed Solution

S.No.	Parameter	Description

1.	Problem Statement (Problem to be solved)	Digit recognition is essential in the modern world. It has the capacity to resolve problems that are getting harder and easier while facilitating human work. One instance is the recognition of handwritten digits. This is a technique that is used globally to identify zip codes or postal codes for mail sorting. A variety of methods can be used to recognise handwritten digits. Because handwritten numerals are not always accurate and can be produced in a variety of ways, the machine has a challenging task. Handwritten digit identification, which uses a picture of a digit to identify the digit represented in the image, offers a solution to this problem.
2.	Idea / Solution description	With 60,000 training photos of handwritten digits from 0 to 9 and 10,000 test images, the MNIS dataset is used to conduct handwritten digit recognition. The MNIST dataset therefore includes 10 distinct classifications. We're going to put into practise a Convolutional Neural Networks model trained application for handwritten digit recognition in this project. The user enters the handwritten digit into a GUI, which recognises it, and the answer is shown instantly.
3.	Novelty / Uniqueness	In the field of handwriting visual recognition, this study presents a practical method for addressing novelty. In addition to identifying any aesthetic differences that could exist inside or across texts, the ideal transcription agent would be able to discriminate between known and unknown characters in an image. The novelty is brought in by making use of tools and algorithms that generates multiple copies of the image with different types of altercations in width, height, skew, etc. This makes the model more accurate and reliable.
4.	Social Impact / Customer Satisfaction	With the handwriting recognition technology come a lot of advantages. In addition to

		reading postal addresses and bank check amounts, it is
		also helpful for reading forms. As a result of how simple it is to compare two texts and establish whether one is a copy, it is also employed in the detection of fraud. This suggested approach ought to be capable of detecting those digits as well, given that users in rural locations will speak their own regional language. Given that it is intended to address real-life issues, it must be completely trustworthy and extremely dependable in all respects, and it must be used by people all over the world.
5.	Business Model (Revenue Model)	The major revenue generating sectors are banking, healthcare, retail, tourism, logistics, transportation, government, manufacturing, and other sectors. All procedures are now quicker and easier to access as a result of digitalization in commercial organisations. Data is becoming an essential component for success as businesses experience technological breakthroughs. When information is transformed into digital form, it can be processed by computers and other computing devices, making it simple to distribute, access, and store. Hence the market value of this technology is very high.
6.	Scalability of the Solution	Scaling of the model can be achieved by expanding the dataset to regional languages. This makes it very useful especially in rural areas where people are prone to writing in the local text. Another method is to use IBM Cloud AI to optimize, train and improve the efficiency of the working model. The high accuracy and reliability makes it more desirable to the market.

3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Upload image	Image upload via files Image upload via folders Image upload via drive Image upload via web Image upload via scan/camera
FR-4	Spelling support	Identifies handwriting of different styles and fonts Spelling check
FR-5	Translation	Handwritten digits from the image are extracted. Conversion of handwritten digits into machine readable form
FR-6	Log out	Log out / sign out.

4.2 Non-Functional requirements

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The proposed system gives good results for images that contain handwritten text written in different styles, different size and alignment with varying background
NFR-2	**Security**	Only authorized people can access the system data and modify the database.
NFR-3	**Reliability**	The Database is frequently updated with handwriting of different styles and size and will roll back when any update fails.

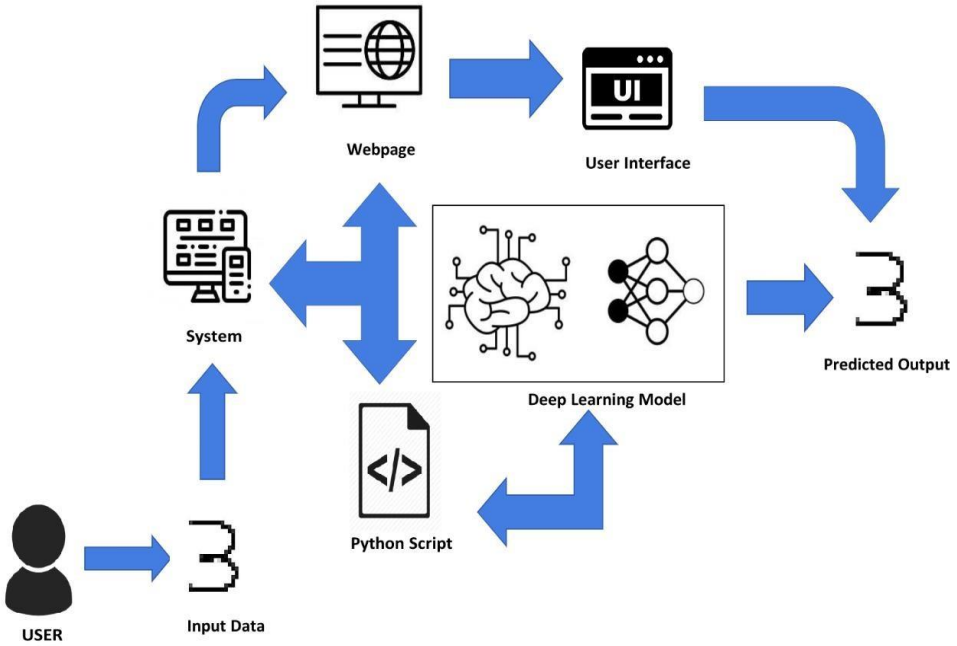
NFR-4	**Performance**	The proposed system is advantageous as it uses fewer features to train the neural network, which results in faster convergence.
NFR-5	**Availability**	The system functionality and services are available for use with all operations.
NFR-6	**Scalability**	The website traffic limit must be scalable enough to support 2 lakhs users at a time

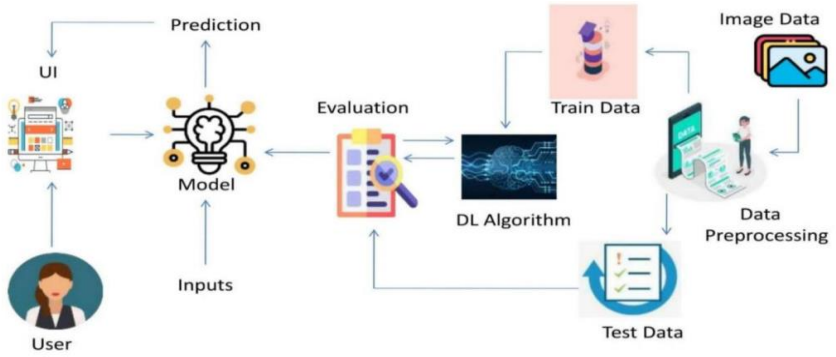
5. PROJECT DESIGN

5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

5.2 Solution & Technical Architecture





5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Web user (customer)	Access web page	USN-1	As a user, anyone can access the web page to upload the handwritten image	I can access my web page through online at any time	High	Sprint-1
	Usage of handwritten data	USN-2	As per the style of the handwriting, it is easy to predict the input	Prediction can be done in an easy way	High	Sprint-2
	Accuracy of the handwriting	USN-3	By using the prediction model, the user can check whether the digit is recognized correctly	Prediction of handwritten digit will be accurate	High	Sprint-3
	View the result	USN-4	As a user, he/she can view the digitalized form of the input	Final result will be displayed	High	Sprint-3
Customer Care Executive	Upload clear image/ draw clearly	USN-5	As a user, he/she need to upload clear and neat image to increase accuracy	Result will be accurate	High	Sprint-3

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	As a user, I can collect the dataset from various resources with different handwritings.	10	High	Pratheeba R Sai Raksha V
Sprint-1	Data Pre-processing	USN-2	As a user, I can load the dataset, handling the missing data, scaling and split data into train and test.	10	High	Sandhiya R Suriya R
Sprint-2	Model Building	USN-3	As a user, I will get an application with ML model which provides high accuracy of recognized handwritten digit.	5	Medium	Pratheeba R Suriya R
Sprint-2	Add CNN layers	USN-4	Creating the model and adding the input, hidden, and output layers to it.	5	Medium	Sai Raksha V Sandhiya R
Sprint-2	Compiling the model	USN-5	With both the training data defined and model defined, it's time to configure the learning process.	5	Medium	Pratheeba R Suriya R
Sprint-2	Cloud Deployment	USN-6	As a user, I can access the web application and make the use of the product from anywhere	5	Medium	Sai Raksha V Sandhiya R
Sprint-3	Building UI Application	USN-7	As a user, I will upload the handwritten digit image to the application by clicking a upload button.	10	High	Pratheeba R Sai Raksha V Sandhiya R Suriya R
Sprint-3	Run the application	USN-8	As a user, I can know the details of the fundamental usage of the application.	5	Medium	Pratheeba R Sai Raksha V Sandhiya R Suriya R
Sprint-3	Digit Recognition	USN-9	As a user, I can see the predicted / recognized digits in the application.	5	Medium	Pratheeba R Sai Raksha V Sandhiya R Suriya R
Sprint-4	Train the model on IBM	USN-10	As a user, I train the model on IBM and integrate flask/Django with scoring end point.	10	High	Pratheeba R Sai Raksha V
Sprint-4	Train & test the model	USN-11	As a user, I will test the application.	10	High	Sandhiya R Suriya R

6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

7.CODING & SOLUTIONING (Explain the features added in the project along with code)

You can get very good results using a very simple neural network model with a single hidden layer. In this section, you will create a simple multi-layer perceptron model that achieves an error rate of 1.74%. You will use this as a baseline for comparing more complex convolutional neural network models.

```
from flask import Flask, render_template, request
from scipy.misc import imsave, imread, imresize
import numpy as np
import keras.models
import re
import base64
```

7.1 OpenCV

The Opencv(Open Source Computer Vision Library) is a python DIL library that is very updated in the work of Image processing. One image file and pixel values can easily come to surface by this library. This library provides a common infrastructure and module related to computer vision technologies. The most important thing about this tool is it is totally free and can be easily modified and changed respective to input by the programmer.

7.2 Tensorflow

TensorFlow is an end-to-end open-source package in python for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML, and developers easily build and deploy ML-powered applications.

7.3 Image Processing

```
    # parse canvas bytes and save as output.png
def parseImage(imgData):
    imgstr = re.search(b'base64,(.*)', imgData).group(1)
    with open('output.png','wb') as output:
        output.write(base64.decodebytes(imgstr))
```

SO once the neural net work has been trained for all ten digits now it is possible to identify the meaning of any hand written digit with the help of the trained neural network. So now when ever a handwritten digit will be given as sample input the system will calculate its global histogram and then feed the global histogram to the neural

network. This time the neural network will take the bias and weight from the already stored text files and use that for detecting the neuron firing sequence. So in the output array it will automatically give the digit whose corresponding match value is detected. And in case none of the neurons is fired that means that it is a new digit which is not available in the trained neural network files which are also called knowledge base library.

8.TESTING

Test Cases

Test Case 1:
Steps to Reproduce :

1. Enter the test image.
2. Wait for Prediction

Type of Test: Functional

Result : Test Case Passed – Working as Expected.

Test Case 2:

Steps to Reproduce :

1. Enter the test image.
2. Wait for the Prediction.

Type of Test: Functional

Result : Test Case Passed [Lower Accuracy.] – Working as Expected.

Regression Testing:

It is defined as a type of software testing to confirm that a recent program or code change has not adversely affected existing features. Regression Testing is nothing but a full or partial selection of already executed test cases that are re-executed to ensure existing functionalities work fine

Digits	Test 1 (Detected)	Test 2 (Detected)	Test 3 (Detected)	Test 4 (Detected)	Test 5 (Detected)
0	60	78	110	116	132
1	50	43	87	97	124
2	72	88	86	114	129
3	23	67	89	109	110
4	45	46	66	78	125
5	56	78	68	90	134
6	76	87	57	104	128
7	45	40	65	99	138
8	45	89	95	134	134
9	55	78	127	130	128

Result: Regression Testing Passed.

9. RESULTS

6.3 Performance Metrics

1. Model Summary

   Model: "sequential" Layer (type)

   Output Shape Param # conv2d (Conv2D) (None, 26, 26, 64) 640 conv2d_1 (Conv2D) (None, 24, 24, 32) 18464 flatten (Flatten) (None, 18432) 0 dense (Dense) (None, 10) 184330

   Total params: 203,434 Trainable params: 203,434 Non-trainable params: 0

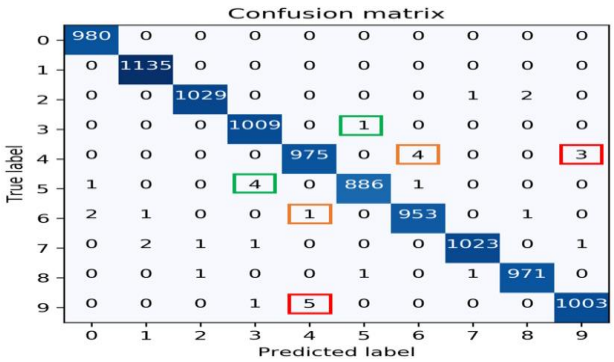
2. Accuracy

   Training Accuracy -0.9979166388511658

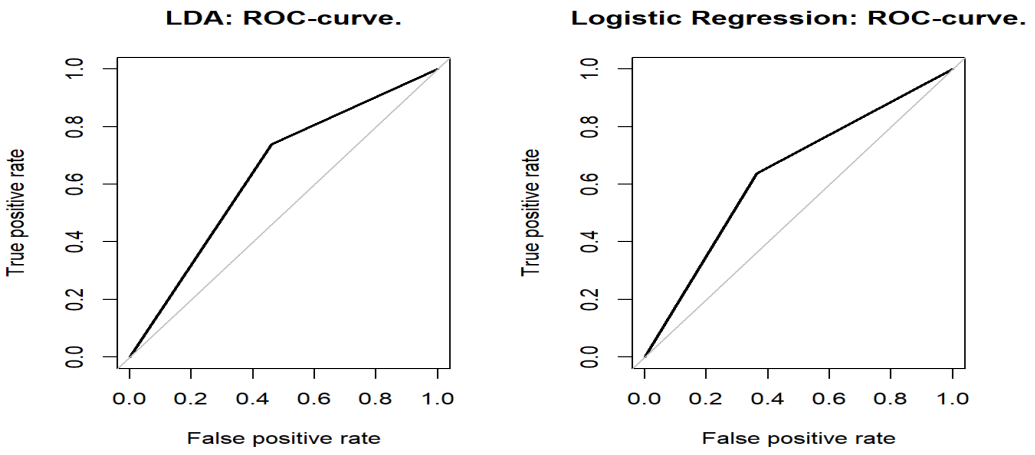
   Validation Accuracy -0.98089998960495

3. Metrics

   Confusion Matrix:



   ROC Curve:



10. ADVANTAGES & DISADVANTAGES

Advantages:

1. The system not only produces a classification of the digit but also a rich description of the instantiation parameters which can yield information such as the writing style.
2. The generative models can perform recognition driven segmentation.
3. The method involves a relatively small number of parameters and hence training is relatively easy and fast.

Disadvantages:

The disadvantage is that it is not done in real time as a person writes and therefore not appropriate for immediate text input.

Furthermore, OCR plays an important role for digital libraries, allowing the entry of image textual information into computers by digitization, image restoration, and recognition methods.

11. CONCLUSION

Given that everyone in the world has their own writing style, handwriting detection is one of the most intriguing research project. It is the computer's capacity to automatically recognise and understand handwritten figures or letters. Because of advances in science and technology, everything is being digitalized to reduce human effort. As a result, handwritten digit identification is required in many real-time applications. The MNIST data collection, which contains 70000 handwritten digits, is employed in this recognition process.

12. FUTURE SCOPE:

The future development of applications based on algorithms of deep and machine learning is practically boundless. In the future, we can work on a denser or hybrid algorithm than the current set of algorithms with more manifold data to achieve solutions to many problems. In the future, the application of these algorithms lies from the public to high-level authorities, as from the

differentiation of the algorithms above and with future development we can attain high-level functioning applications which can be used in the classified or government agencies as well as for the common people, we can use these algorithms in hospitals application for detailed medical diagnosis, treatment and monitoring the patients, we can use it in surveillances system to keep tracks of the suspicious activity under the system, in fingerprint and retinal scanners, database filtering applications, Equipment checking for national forces and many more problems of both major and minor category. The advancement in this field can help us create an environment of safety, awareness and comfort by using these algorithms in day to day application and high-level application (i.e. Corporate level or Government level). Application-based on artificial intelligence and deep learning is the future of the technological world because of their absolute accuracy and advantages over many major problems.

13.APPENDIX

Source Code:

Handwritten Digit Recognition.ipynb:

Import the necessary packages

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from keras.utils import np_utils
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, Dense, Flatten
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.models import load_model
from PIL import Image, ImageOps
```

Load data

```
(X_train, y_train), (X_test, y_test) = mnist.load_data()
```

Data Analysis

```
print(X_train.shape)
print(X_test.shape)
```

```
X_train[0]
```

```
y_train[0]
```

```
plt.imshow(X_train[0])
```

Data Pre-Processing

```
X_train = X_train.reshape(60000, 28, 28, 1).astype('float32')
X_test = X_test.reshape(10000, 28, 28, 1).astype('float32')
number_of_classes = 10
Y_train = np_utils.to_categorical(y_train, number_of_classes)
Y_test = np_utils.to_categorical(y_test, number_of_classes)
Y_train[0]
```

Create model

```
model = Sequential()
model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation="relu"))
model.add(Conv2D(32, (3, 3), activation="relu"))
model.add(Flatten())
model.add(Dense(number_of_classes, activation="softmax"))
model.compile(loss='categorical_crossentropy', optimizer="Adam", metrics=["accuracy"])
```

Train the model

```
model.fit(X_train, Y_train, batch_size=32, epochs=5, validation_data=(X_test,Y_test))
```

Test the model

```
metrics = model.evaluate(X_test, Y_test, verbose=0)
print("Metrics (Test Loss & Test Accuracy): ")
print(metrics)
prediction = model.predict(X_test[:4])
print(prediction)
print(numpy.argmax(prediction, axis=1))
print(Y_test[:4])
```

Save the model

```
model.save("model.h5")
```

Test the saved model

```
model=load_model("model.h5")
img = Image.open("sample.png").convert("L")
img = img.resize((28, 28))
img2arr = np.array(img)
img2arr = img2arr.reshape(1, 28, 28, 1)
results  = model.predict(img2arr)
results = np.argmax(results,axis = 1)
results = pd.Series(results,name="Label")
print(results)
```

recognizer.py:

```
Import
os
        import random
        import string
        from pathlib import Path
        import numpy as np
        from tensorflow.keras.models import load_model
```

```
from PIL import Image, ImageOps


def random_name_generator(n: int) -> str:
	"""
	Generates a random file name.
	Args:
		n (int): Length the of the file name.
	Returns:
		str: The file name.
	"""
	return ''.join(random.choices(string.ascii_uppercase + string.digits, k=n))

def recognize(image: bytes) -> tuple:
	"""
	Predicts the digit in the image.
	Args:
		image (bytes): The image data.
	Returns:
		tuple: The best prediction, other predictions and file name
	"""

	model=load_model(Path("./model/model.h5"))

	img = Image.open(image).convert("L")

	# Generate a random name to save the image file.
	img_name = random_name_generator(10) + '.jpg'
	if not os.path.exists(f"./static/data/"):
		os.mkdir(os.path.join('./static/', 'data'))
	img.save(Path(f"./static/data/{img_name}"))

	# Convert the Image to Grayscale, Invert it and Resize to get better
prediction.
	img = ImageOps.grayscale(img)
	img = ImageOps.invert(img)
	img = img.resize((28, 28))

	# Convert the image to an array and reshape the data to make prediction.
	img2arr = np.array(img)
	img2arr = img2arr / 255.0
	img2arr = img2arr.reshape(1, 28, 28, 1)
```

```
results  = model.predict(img2arr)
best = np.argmax(results,axis = 1)[0]

# Get all the predictions and it's respective accuracy.
pred = list(map(lambda x: round(x*100, 2), results[0]))

values = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
others = list(zip(values, pred))
  # Get the value with the highest accuracy
  best = others.pop(best)

  return best, others, img_name
```

Test Deployed Model.ipynb:

Import necessary libraries

```
import requests
import numpy as np
from PIL import Image, ImageOps
import matplotlib.pyplot as plt
```

Input pre-processing

```
img = Image.open(f"../sample/sample 1.png").convert("L")
img = ImageOps.invert(img)
img = img.resize((28, 28))
img_arr = np.array(img)
img_arr = img_arr / 255.0
img_arr = img_arr.reshape(28, 28, 1)

img2 = Image.open(f"../sample/sample 2.png").convert("L")
img2 = ImageOps.invert(img2)
```

```
img2 = img2.resize((28, 28))
img2_arr = np.array(img2)
img2_arr = img2_arr / 255.0
img2_arr = img2_arr.reshape(28, 28, 1)

img3 = Image.open(f"../sample/sample 3.png").convert("L")
img3 = ImageOps.invert(img3)
img3 = img3.resize((28, 28))
img3_arr = np.array(img3)
img3_arr = img3_arr / 255.0
img3_arr = img3_arr.reshape(28, 28, 1)
```

Get results from the deployed model

```
API_KEY = ""

token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
                               data={"apikey": API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-
type:apikey'})

mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

payload_scoring = {"input_data": [{"fields": [], "values": [img_arr.tolist(), img2_arr.tolist(),
img3_arr.tolist()]}]}

response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/ae43e79c-1fbc-450a-b0b4-
9a54c451033b/predictions?version=2022-11-10',
                                 json=payload_scoring, headers={'Authorization': 'Bearer ' + mltoken})
```

Display results

```
plt.imshow(plt.imread("../sample/sample 1.png"))
plt.axis('off')
plt.show()
print("Result: ", response_scoring.json()['predictions'][0]['values'][0][1])
plt.imshow(plt.imread("../sample/sample 2.png"))
plt.axis('off')
plt.show()
print("Result: ", response_scoring.json()['predictions'][0]['values'][1][1])
```

```
plt.imshow(plt.imread("../sample/sample 3.png"))
plt.axis('off')
plt.show()
print("Result: ", response_scoring.json()['predictions'][0]['values'][2][1])
```

Train and Deploy Model.ipynb:

Import necessary libraries

```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

import keras
from keras.models import Sequential
from keras.layers import Conv2D, Lambda, MaxPooling2D
from keras.layers import Dense, Dropout, Flatten
from tensorflow.keras.layers import BatchNormalization

from keras.preprocessing.image import ImageDataGenerator

from keras.utils.np_utils import to_categorical

from keras.datasets import mnist
```

Load the data

```
(X_train, Y_train), (X_test, Y_test) = mnist.load_data()
print(X_train.shape)
print(X_test.shape)
```

Data pre-processing

```
X_train = X_train / 255.0
X_test = X_test / 255.0
```

```
X_train = X_train.reshape(-1,28,28,1)
X_test = X_test.reshape(-1,28,28,1)

Y_train = to_categorical(Y_train)
Y_test = to_categorical(Y_test)

mean = np.mean(X_train)
std = np.std(X_train)

def standardize(x):
    return (x-mean)/std
```

Create model

```
model=Sequential()

model.add(Conv2D(filters=64, kernel_size = (3,3), activation="relu", input_shape=(28,28,1)))
model.add(Conv2D(filters=64, kernel_size = (3,3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(BatchNormalization())

model.add(Conv2D(filters=128, kernel_size = (3,3), activation="relu"))
model.add(Conv2D(filters=128, kernel_size = (3,3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(BatchNormalization())

model.add(Conv2D(filters=256, kernel_size = (3,3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(BatchNormalization())

model.add(Flatten())
model.add(Dense(512,activation="relu"))

model.add(Dense(10,activation="softmax"))

model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])
model.summary()
```

Define training parameters

```
datagen = ImageDataGenerator(
        featurewise_center=False,
        samplewise_center=False,
```

```
        featurewise_std_normalization=False,
        samplewise_std_normalization=False,
        zca_whitening=False,
        rotation_range=15,
        zoom_range = 0.01,
        width_shift_range=0.1,
        height_shift_range=0.1,
        horizontal_flip=False,
        vertical_flip=False)


train_gen = datagen.flow(X_train, Y_train, batch_size=128)
test_gen = datagen.flow(X_test, Y_test, batch_size=128)


epochs = 10
batch_size = 128
train_steps = X_train.shape[0] // batch_size
valid_steps = X_test.shape[0] // batch_size

es = keras.callbacks.EarlyStopping(
        monitor="val_accuracy",
        patience=10,
        verbose=1,
        mode="max",
        restore_best_weights=True,
    )

rp = keras.callbacks.ReduceLROnPlateau(
        monitor="val_accuracy",
        factor=0.2,
        patience=3,
        verbose=1,
        mode="max",
        min_lr=0.00001,
    )
```

Train the model

```
history = model.fit(train_gen, epochs = epochs,
                    steps_per_epoch = train_steps,
                    validation_data = test_gen,
                    validation_steps = valid_steps,
                    callbacks=[es, rp])
```

Save the model

```
model.save("model.h5")
!tar -zcvf model.tgz model.h5
```

Install necessary packages

```
!pip install watson-machine-learning-client
```

Connect to IBM Watson Machine Learning instance

```
from ibm_watson_machine_learning import APIClient

API_KEY = ""

credentials = {
   "url": "https://us-south.ml.cloud.ibm.com",
   "apikey": API_KEY
}

client = APIClient(credentials)

def guid_from_space_name(client, space_name):
   space = client.spaces.get_details()
   return(next(item for item in space['resources'] if item['entity']['name'] ==
space_name)['metadata']['id'])

space_uid = guid_from_space_name(client, 'Handwritten Digit Recognition')
print("Space UID: ", space_uid)


client.set.default_space(space_uid)
```

Define model specifications for deployment

```
client.software_specifications.list()


software_spec_uid = client.software_specifications.get_uid_by_name("runtime-22.1-py3.9")
software_spec_uid
```

```
model_details = client.repository.store_model(model="model.tgz", meta_props={
    client.repository.ModelMetaNames.NAME: "CNN",
    client.repository.ModelMetaNames.TYPE: "tensorflow_2.7",
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
})

model_id = client.repository.get_model_id(model_details)
model_id
```

Download the deployed model

```
client.repository.download(model_id, "model.tar.gz")
```

HDR cloud deployment.ipynb:

Importing the required libraries

```
!pip install tensorflow --upgrade

import numpy as np
import tensorflow #open source used for both ML and DL for computation
from tensorflow.keras.datasets import mnist #mnist dataset
from tensorflow.keras.models import Sequential #it is a plain stack of layers
from tensorflow.keras import layers #A Layer consists of a tensor- in tensor-out computat ion funct ion
from tensorflow.keras.layers import Dense, Flatten #Dense-Dense Layer is the regular deeply connected r
#faltten -used fot flattening the input or change the dimension
from tensorflow.keras.layers import Conv2D #onvoLutiona l Layer
from keras.optimizers import Adam #opt imizer
from keras. utils import np_utils #used for one-hot encoding
import matplotlib.pyplot as plt   #used for data visualization
```

Load data

(x_train, y_train), (x_test, y_test)=mnist.load_data () *#splitting the mnist data into train and test*
print (x_train.shape) *#shape is used for give the dimens ion values #60000-rows 28x28-pixels*
print (x_test.shape)

x_train[0]

plt.imshow(x_train[5100]) *#ploting the index=image*

np.argmax(y_train[5100])

Reshaping Dataset

#Reshaping to format which CNN expects (batch, height, width, channels)
x_train=x_train.reshape (60000, 28, 28, 1).astype('float32')
x_test=x_test.reshape (10000, 28, 28, 1).astype ('float32')

Applying one hot Encoding

number_of_classes = 10 *#storing the no of classes in a variable*
y_train = np_utils.to_categorical (y_train, number_of_classes) *#converts the output in binary format*
y_test = np_utils.to_categorical (y_test, number_of_classes)

Add CNN Layers

#create model
model=Sequential ()
#adding modeL Layer
model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation='relu'))
model.add(Conv2D(32, (3, 3), activation = 'relu'))
#flatten the dimension of the image
model.add(Flatten())
#output layer with 10 neurons
model.add(Dense(number_of_classes,activation = 'softmax'))
#Compile model

Compiling the Model

```
model.compile(loss= 'categorical_crossentropy', optimizer="Adam", metrics=['accuracy'])
x_train = np.asarray(x_train)
y_train = np.asarray(y_train)
```

Train the model

```
#fit the model
model.fit(x_train, y_train, validation_data=(x_test, y_test), epochs=5, batch_size=32)
```

Observing the metrics

```
# Final evaluation of the model
metrics = model.evaluate(x_test, y_test, verbose=0)
print("Metrics (Test loss &Test Accuracy) : ")
print(metrics)
```

Test The Model

```
prediction=model.predict(x_test[6000:6001])
print(prediction)


plt.imshow(x_test[6000])


import numpy as np
print(np.argmax(prediction, axis=1)) #printing our Labels from first 4 images

np.argmax(y_test[6000:6001]) #printing the actual labels
```

Save the model

```
# Save the model
model.save('models/mnistCNN.h5')
cd models
/home/wsuser/work/models/models
!tar -zcvf hdr_deployment.tgz mnistCNN.h5
mnistCNN.h5
```

```
ls -1


!pip install watson-machine-learning-client --upgrade
```

Cloud deploy

```
from ibm_watson_machine_learning import APIClient
credentials ={
   "url":"https://us-south.ml.cloud.ibm.com",
   "apikey":"Qxwy3byu83al_Lvmk05S2xcRhHqeQiy_4BxWzPcxuB9A"
}
client = APIClient(credentials)
client
client.spaces.get_details()


def guid_from_space_name(client,deploy):
  space = client.spaces.get_details()
  return (next(item for item in space['resources'] if
item['entity']['name']==deploy)['metadata']['id'])

space_uid = guid_from_space_name(client,'hdr')
print("Space UID = " + space_uid)


client.set.default_space(space_uid)



client.software_specifications.list(limit=100)



software_space_uid = client.software_specifications.get_uid_by_name('tensorflow_rt22.1-
py3.9')
software_space_uid



model_details = client.repository.store_model(model='hdr_deployment.tgz',meta_props={
   client.repository.ModelMetaNames.NAME:"Digit Recognition System",
   client.repository.ModelMetaNames.TYPE:"tensorflow_2.7",
   client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_space_uid
```

```
})
```

```
model_details
```

```
model_id = client.repository.get_model_id(model_details)
model_id
```

```
client.repository.download(model_id,'DigitRecog_IBM_model.tar.gz')
```

```
ls
```

Test Model

```
from tensorflow.keras.models import load_model
from keras.preprocessing import image
from PIL import Image
import numpy as np
model = load_model("mnistCNN.h5")
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='1rEQ4QsDyr45SbIYkkmEXGolFpDjMBjlc1KmxrsH2V1U',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'digitrecognition-donotdelete-pr-kvpefjqsoxebrc'
object_key = '4.jpg'

streaming_body_3 = cos_client.get_object(Bucket=bucket, Key=object_key)['Body']
```

```
img = Image.open(streaming_body_3).convert("L") # convert image to monochrome
img = img.resize( (28,28) ) # resizing of input image
img
im2arr = np.array(img) #converting to image
im2arr = im2arr.reshape(1, 28, 28, 1) #reshaping according to our requirement
pred = model.predict(im2arr)
print(pred)


print(np.argmax(pred, axis=1)) #printing our Labels
```

app.py:

```
from flask import Flask,render_template,request
from recognizer import recognize

app=Flask(__name__)

@app.route('/')
def main():
    return render_template("home.html")


@app.route('/predict',methods=['POST'])
def predict():
    if request.method=='POST':
      image = request.files.get('photo', '')
      best, others, img_name = recognize(image)
      return render_template("predict.html", best=best, others=others, img_name=img_name)


if __name__=="__main__":
    app.run()
```

Recognizer.py

```
import os
import random
import string
from pathlib import Path
import numpy as np
from tensorflow.keras.models import load_model
from PIL import Image, ImageOps


def random_name_generator(n: int) -> str:
    """
    Generates a random file name.

    Args:
        n (int): Length the of the file name.

    Returns:
        str: The file name.
    """
    return ''.join(random.choices(string.ascii_uppercase + string.digits, k=n))

def recognize(image: bytes) -> tuple:
    """
    Predicts the digit in the image.

    Args:
        image (bytes): The image data.

    Returns:
        tuple: The best prediction, other predictions and file name
    """

    model=load_model(Path("./model/model.h5"))

    img = Image.open(image).convert("L")

    # Generate a random name to save the image file.
    img_name = random_name_generator(10) + '.jpg'
    if not os.path.exists(f"./static/data/"):
        os.mkdir(os.path.join('./static/', 'data'))
    img.save(Path(f"./static/data/{img_name}"))
```

```
    # Convert the Image to Grayscale, Invert it and Resize to get better prediction.
    img = ImageOps.grayscale(img)
    img = ImageOps.invert(img)
    img = img.resize((28, 28))

    # Convert the image to an array and reshape the data to make prediction.
    img2arr = np.array(img)
    img2arr = img2arr / 255.0
    img2arr = img2arr.reshape(1, 28, 28, 1)

    results  = model.predict(img2arr)
    best = np.argmax(results,axis = 1)[0]

    # Get all the predictions and it's respective accuracy.
    pred = list(map(lambda x: round(x*100, 2), results[0]))

    values = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
    others = list(zip(values, pred))

    # Get the value with the highest accuracy
    best = others.pop(best)

    return best, others, img_name
```

home.html

```
<html>
   <head>
      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
      <title>Handwritten Digit Recognition</title>
      <link rel="icon" type="image/svg" sizes="32x32"
href="{{url_for('static',filename='images/icon.svg')}}" />
      <link rel="stylesheet" href="{{url_for('static',filename='css/main.css')}}" />
      <script src="https://unpkg.com/feather-icons"></script>
      <script defer src="{{url_for('static',filename='js/script.js')}}"></script>
   </head>
   <body>
      <div class="container">
         <div class="heading">
            <h1 class="heading__main">Handwritten Digit Recognizer</h1>
            <h2 class="heading__sub">Easily analyze and detect handwritten digits</h2>
         </div>
         <div class="upload-container">
            <div class="form-wrapper">
```

```
            <form class="upload" action="/predict" method="post" enctype="multipart/form-
data">
                <label id="label" for="upload-image"><i data-feather="file-plus"></i>Select
File</label>
                <input type="file" name="photo" id="upload-image" hidden />
                <button type="submit" id="up_btn"></button>
            </form>
            <img id="loading" src="{{url_for('static',filename='images/loading.gif')}}">
          </div>
        </div>
      </div>
   </body>
</html>
```

Predict.html

```
<html>
   <head>
      <title>Prediction | Handwritten Digit Recognition</title>
      <link rel="stylesheet" href="{{url_for('static',filename='css/predict.css')}}" />
      <link rel="icon" type="image/svg" sizes="32x32"
href="{{url_for('static',filename='images/icon.svg')}}" />
      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
   </head>
   <body>
      <div class="container">
         <h1>Prediction</h1>
         <div class="result-wrapper">
            <div class="input-image-container">
               <img src="{{url_for('static',filename='data/')}}{{img_name}}" />
            </div>
            <div class="result-container">
               <div class="value">{{best.0}}</div>
               <div class="accuracy">{{best.1}}%</div>
            </div>
         </div>
         <h1>Other Predictions</h1>
         <div class="other_predictions">
            {% for x in others %}
            <div class="value">
               <h2>{{x.0}}</h2>
```

```
                <div class="accuracy">{{x.1}}%</div>
              </div>
              {% endfor %}
            </div>
        </div>
    </body>
</html>
```

GitHub & Project Demo Link:

GitHub –

https://github.com/IBM-EPBL/IBM-Project-25624-1659969014

Project Demo Link –

https://drive.google.com/file/d/151HfeeeY5FtlW5XT1v6BC9iCqKoBSSzj/view?usp=sharing