K.L.N College of Information Technology, PottapalayamDepartment of Computer science

Sub.Code : HX 8001

Sub.Name: Professional Readiness for Innovation, Employability and Entrepreneurship

PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION

Submitted by

PNT2022TMID52471

TEAM LEADER : R.ARUNASALAM - 910719104002
TEAM MEMBERS : M.D.DHARSA - 910719104007
TEAM MEMBERS : T.BHARATHI - 910719104005
TEAM MEMBERS : K.SNEKADHARSHINI - 910719104026

Table of contents

CHAPTER 1

1.INTRODUCTION

This research provides a comprehensive comparison between different machine learning and deep learning algorithms for the purpose of handwritten digit recognition while using the Support Vector Machine, Multilayer Perceptron, and Convolutional Neural Network for the same purpose The comparison between these algorithms is carried out on the basis of their accuracy, errors, and testing-training time corroborated by plots and charts that have been constructed using matplotlib for visualization.

1.1PROJECT OVERVIEW

Machine learning and deep learning play an important role in computer technology and artificial intelligence. With the use of deep learning and machine learning, human effort can be reduced in recognizing, learning, predictions and in many more areas Handwritten Digit Recognition is the ability of computer systems to recognise handwritten digits from various sources, such as images, documents, and so on. This project aims to let users take advantage of machine learning to reduce manual tasks in recognizing digits. In coming days, character recognition system might serve as a key factor to create a paperless environment by digitizing and processing existing paper documents. This paper presents a detailed review in the field of Handwritten Character Recognition.

1.2PURPOSE

The handwritten to be recognized is digitized through scanners or camera. The image of the document is segmented into lines , words, and individual character. Each character is recognized using OCR techniques. Finally errors are corrected using lexicons or spelling checkers. Handwritte character recognition is one of the practically important issues in pattern recognition applications of digit recognition includes in postal mail sorting, bank check processing, form dataentry.

CHAPTER 2

2.LITERATURE SURVEY

An early notable attempt in the area of character recognition research is by Grimsdale in 1959.The origin of a great deal of research work in the early sixties was based on an approach known as analysis-by- synthesis method suggested by Eden in 1968. The great importance of Eden's work was that he formally proved that all handwritten characters are formed by a finite number of schematic features, a point that was implicitly included in previous works. This notion was later used in all methods in syntactic (structural) approaches of character recognition. K. Gaurav, Bhatia P. K. [5] Et al, this paper deals with the various pre-processing techniques involved in the character recognition with different kind of images ranges from a simple handwritten form based documents and documents containing colored and complex background and varied intensities. R. Bajaj, L. Dey, S. Chaudhari, they proposed multi classifier connectionist architecture for increasing the recognition reliability and they obtained 89.6% accuracy for handwritten Devanagari numerals**.**

2.1EXISTING PROBLEM

The goal of this project is to create a model that will be able to recognize and determine the handwritten digits from its image by using the concepts of Convolution Neural Network. Though the goal is to create a model which can recognize the digits, it can be extended to letters and an individual's handwriting. The major goal of the proposed system is understanding Convolutional Neural Network, and applying it to the handwritten recognition system.

2.2REFERENCES

1. Handwriting recognition" https://en.wikipedia.org/wiki/Handwriting_recognition
2. "What can a digit recognizer be used for?",
   https://www.quora.com/What-can-a-digit-recognizer-be-used-for.
3. Handwritten Digit Recognition using
   Machine Learning Algorithms", S M Shamim,

Mohammad Badrul Alam Miah, Angona Sarker,

Masud Rana & Abdullah Al Jobair.

4.Handwritten recognition using SVM, KNN, and Neural networks", Norhidayu binti Abdul Hamid, Nilam Nur Binti Amir Sharif.

5. Handwritten Digit Recognition Using
Deep Learning", Anuj Dutt and Aashi Dutt.

2.3PROBLEM STATEMENT DEFINITION

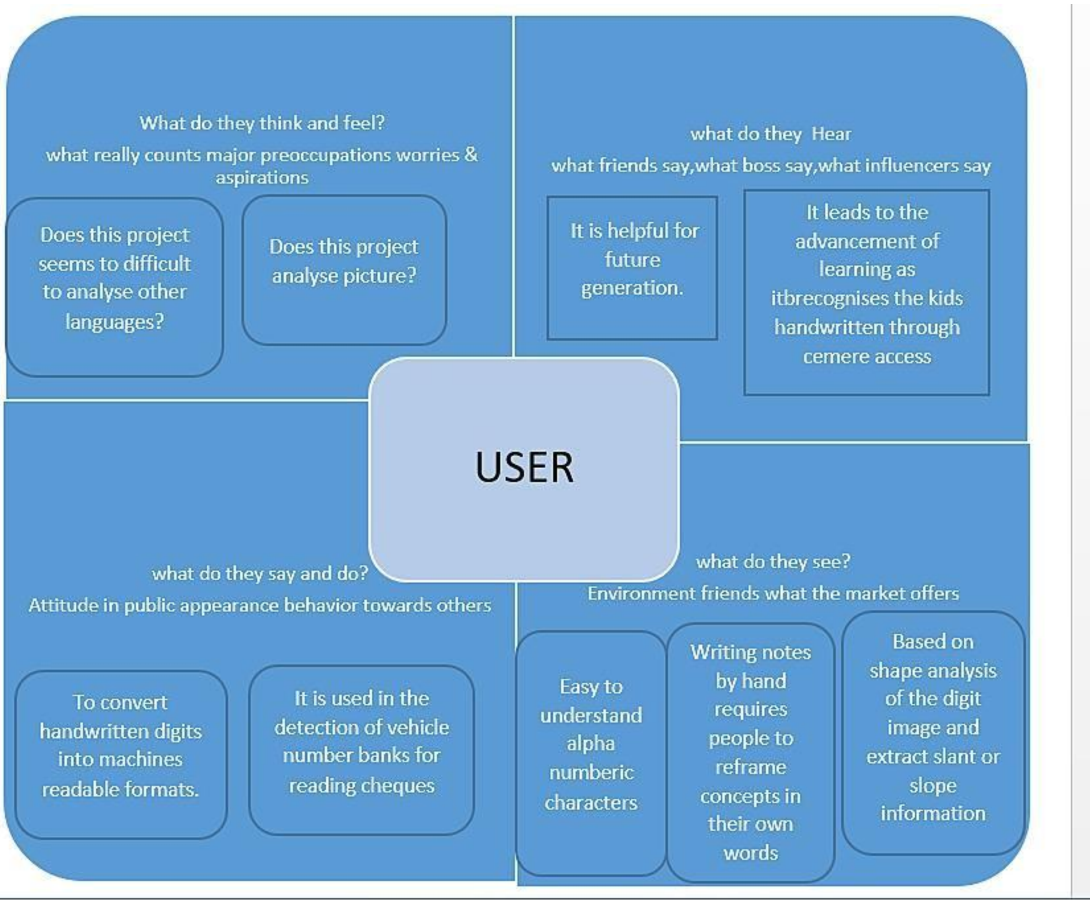
The problem statement is to classify handwritten digits. The goal is to take an image of a handwritten digit and determine what that digit is. The digits range from zero (0) through nine (9).

Who does the problem affect?	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.
What are the boundaries of the Problem?	One of the difficulties in the overall recognition of hand-written digits is the variation and distortion of the hand-written digit collection, because different cultures will employ multiple handwriting kinds and control to extract the characters and identical patterns from their recognized language.

What is the issue?	Digital recognition is also remarkable an important issue.
When does the issue occur?	As the manually written digits aren't of a comparable size, thickness, position and direction, numerous difficulties need to be taken into consideration to decide the problem of handwritten digit recognition. The distinctiveness and collection in the composition styles of numerous people additionally affect the instance and presence of the digits.
Where does the issue occur?	Recognizing handwritten text is a problem that can be traced back to the first automatic machines that needed to recognize individual characters in handwritten documents. Think about, for example, the ZIP codes on letters at the post office and the automation needed to recognize these five digits.

CHAPTER 3
3. IDEATION AND PROPOSED SOLUTION

3.1EMPATHY MAP CANVAS



PAIN

Fears frustrations obstacles

GAINS

Wants/needs measures of success obstacles

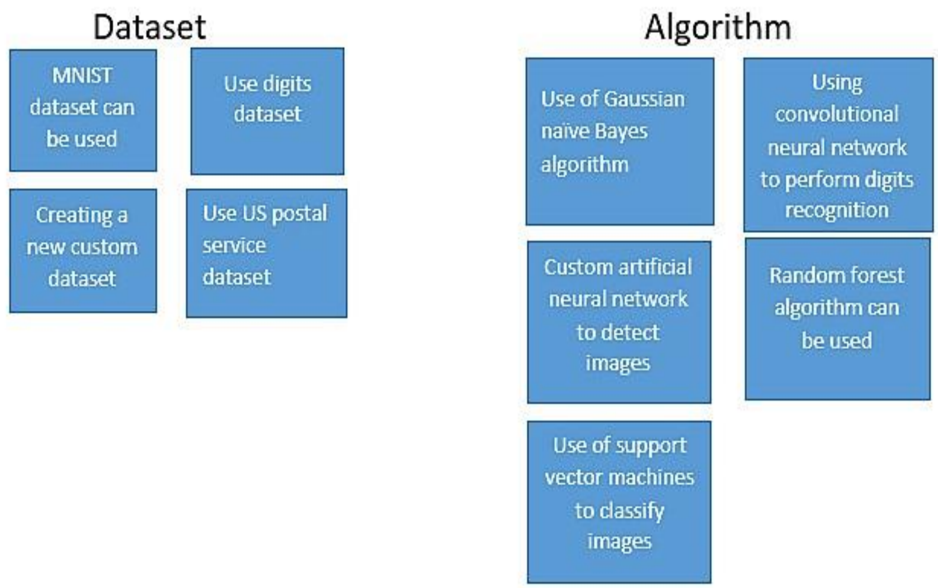
3.2IDEATION & BRAINSTORMING

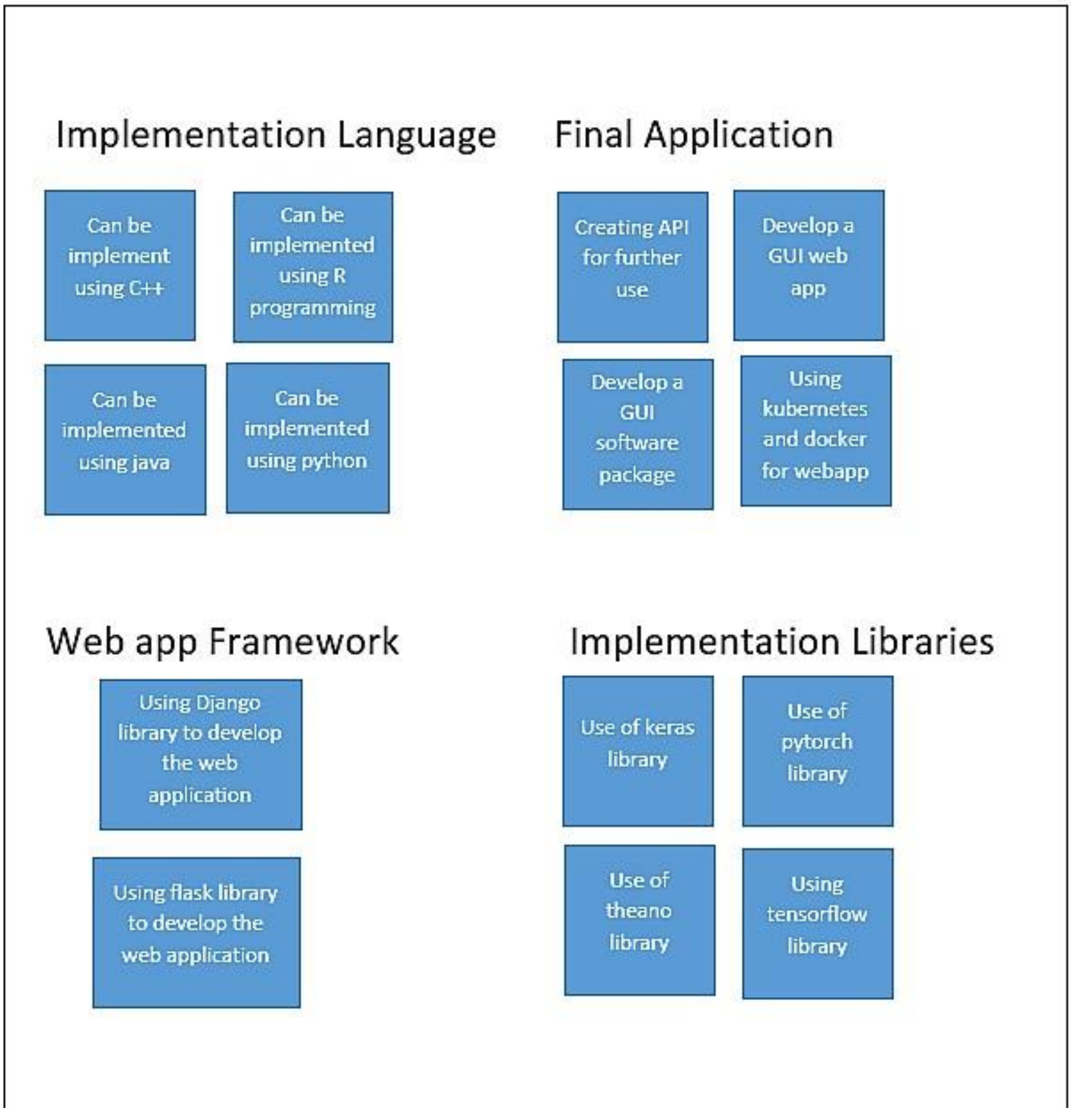
SNEKADHARSHINI.K

Generative models to produce more quality data	Using tensorflow library
Custom artificial neural network to detect images	Training model from scratch
Using kubernetes and docker for web app	Creating a new custom dataset

BHARATHI.T

Using convolutional neural network to perform digit recognition	Offline Recognition
Develop a GUI Web app	Using flask library to develop web application
Can be implemented using C++	Use of pytorch library


Dataset
MNIST dataset can be used
Use digits dataset
Creating a new custom dataset
Use US postal service dataset
Algorithm
Use of Gaussian naïve Bayes algorithm
Using convolutional neural network to perform digits recognition
Custom artificial neural network to detect images
Random forest algorithm can be used
Use of support vector machines to classify images


Implementation Language
Can be implement using C++
Can be implemented using R programming
Can be implemented using java
Can be implemented using python
Final Application
Creating API for further use
Develop a GUI web app
Develop a GUI software package
Using kubernetes and docker for webapp
Web app Framework
Using Django library to develop the web application
Using flask library to develop the web application
Implementation Libraries
Use of keras library
Use of pytorch library
Use of theano library
Using tensorflow library

3.3PROPOSED SOLUTION

SI.NO	Parameter	Description
1	Problem Statement (Problem to be solved)	**Statement**: The handwritten digit recognition is the capability of computer applications to recognize the human handwritten digits. **Description**: It is a hard task for the machine because handwritten digits are not perfect and can be made with many different shapes and sizes.
2	Idea / Solution description	1.It is the capability of a computer to fete the mortal handwritten integers from different sources like images, papers, touch defences. 2.It allows user to translate all those signature and notes into electronic words in a text document format and this data only requires far less physical space than

		the storage of the physical copies.
3	Novelty / Uniqueness	Accurately recognize the digits rather than recognizing all the characters like OCR.
4	Social Impact / Customer Satisfaction	1.Artificial Intelligence developed the app called Handwritten digit Recognizer.

5	Business Model (Revenue Model)	1. This system can be integrated with traffic surveillance cameras to recognize the vehicle's number plates for effective traffic management. 2. Can be integrated with Postal system to identify and recognize the pin-code details easily.
6	Scalability of the Solution	1.Ability to recognise digits in more noisy environments. 2.There is no limit in the number of digits it can be recognized.

3.4 PROBLEM SOLUTION FIT

1.CUSTOMER SEGMENT(S): The Customers who deal with handwritten digits like Banking sectors , schools , colleges ,railways , firms , etc.	**2. JOBS-TO-BE-DONE/PROBLEMS:** Handwrittendigits can be difficult to understand and interpret at times. It may cause errors when dealing with rough handwriting.	**3. TRIGGERS** To obtain the numbers accurately **and quickly.** **4.EMOTIONS BEFORE/AFTER:** Feels frustrated and sad when numbers are not entered.
5.AVAILABLE SOLUTIONS There are no widely used software's to detect handwriting; instead, they check with other people to affirm what number it is.	**6.CUSTOMER CONSTRAINT(S):** They believe that the alternatives will result in errors and faults and will be inconvenient.	**7. BEHAVIOUR** Finding the best software for detecting accurate digits in a more efficient manner
8. CHANNELS OF BEHAVIOUR Using software that is available on the internet. Obtaining assistance from those nearby in order to recognise the digits written by their customers.	**9. PROBLEM ROOT CAUSE** We face numerous challenges in handwritten number recognition. because of different people's jotting styles and the lack of Optic character recognitionThis investigation offers an in-depth comparison of various machine literacy and deep literacy	**10. YOUR SOLUTION** A solution to this problem is the Handwrittendigit recognition system, which uses a picture of a digit and recognises the digit present in theimage. Convolutional Neural Network model built with PyTorch and applied to the MNIST dataset to recognize handwritten digits.

CHAPTER 4
REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

FR No.	Sub Requirement (Story / Sub-Task)
FR-1	Image Data: Handwritten digit recognition refersto a computer's capacity to identifyhuman handwritten digits from a variety of sources, such as photographs, documents, touch screens, etc., and categorise them into ten established classifications (0-9). In the realm of deep learning, this has beenthe subject of countless studies.
FR-2	Website: Web hosting makes the code, graphics, and other items that make up a website accessible online. A server hosts every website you've ever visited. The typeof hosting determines how much space is allotted to a website on a server. Shared, dedicated, VPS, and reseller hostingare the fourbasic varieties.
FR-3	Digit Classifier Model: To train a convolutional network to predict the digit from an image, use the MNISTdatabase of handwritten digits. get the training and validation data first.
FR-4	Cloud: The cloud offers a range of IT services, including virtual storage, networking, servers, databases, and applications. In plain English, cloud computing is described asa virtualplatform that enablesunlimited storage and access to your dataover the internet.
FR-5	Modified National Institute of Standards and Technology dataset: The abbreviation MNIST standsfor the MNISTdataset. It is a collection of 60,000 tinysquare graysc alephotographs, each measuring 28 by 28,comprising handwritten single digits between 0 and 9.

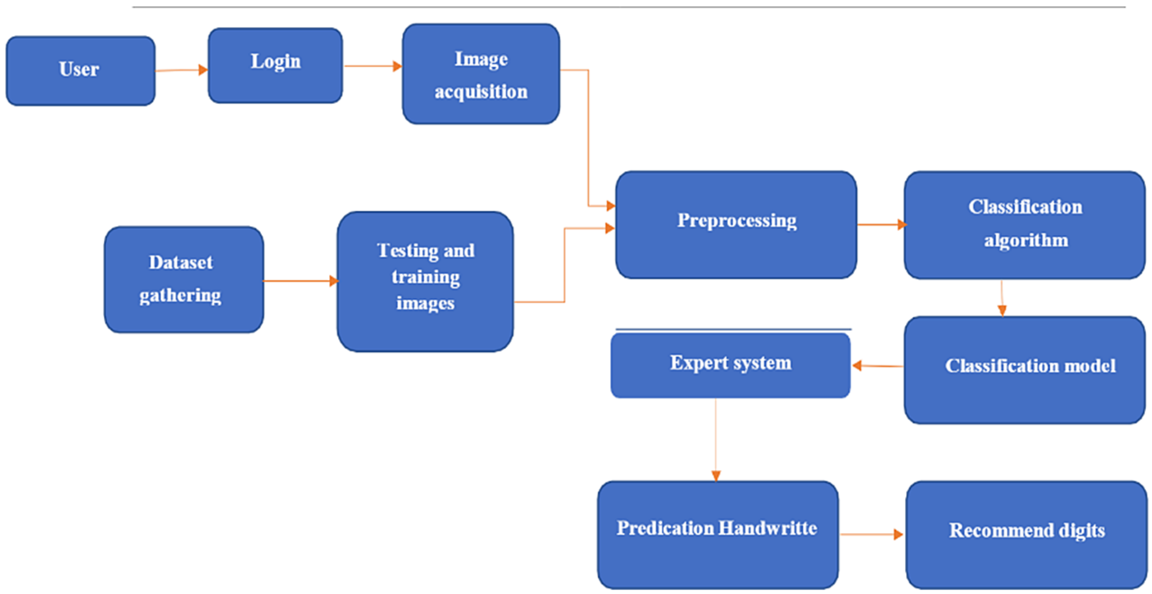
4.2 NON FUNCTIONAL REQUIREMENt

NFR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	One of the very significant problems in pattern recognition applications is the recognition of handwritten characters. Applications for digit recognition includefilling out forms,processing bank checks, and sorting mail.

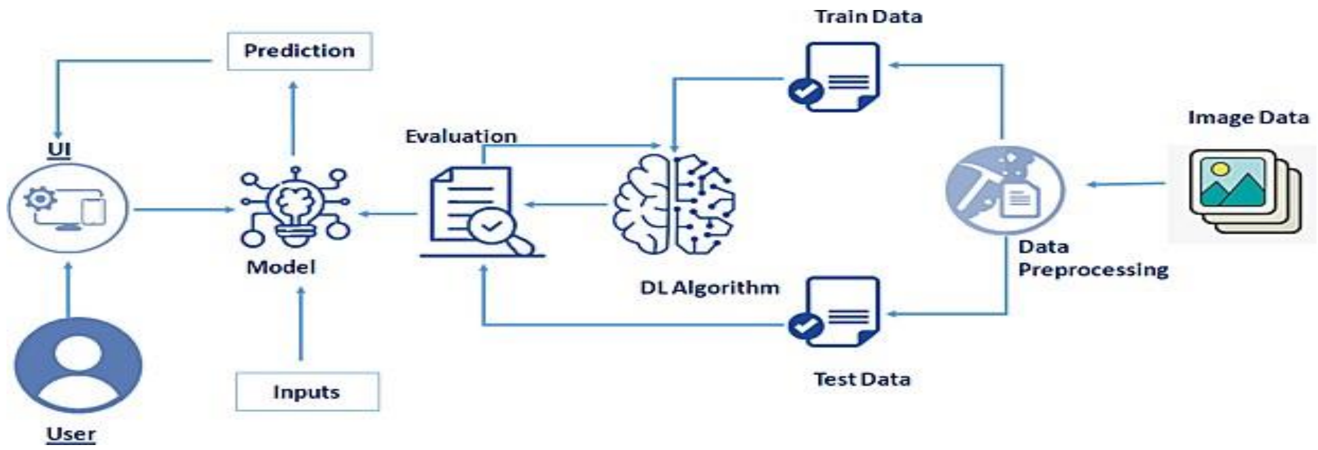
NFR-3	**Reliability**	The samples are used by the neural network toautomatically deduce rules for reading handwritten digits. Furthermore, the network may learn more about handwriting and hence enhance its accuracy by increasing the quantityof training instances. Numerous techniques and algorithms, such asDeep Learning/CNN, SVM, Gaussian Naive Bayes, KNN, Decision Trees, Random Forests, etc., can be used to recognise handwritten numbers.
NFR-4	**Accuracy**	With typed text in high-quality photos, opticalcharacter recognition (OCR) technology offersaccuracy rates of greater than 99%. However,variances in spacing, abnormalities in handwriting, and the varietyof human writingstyles result in less precise character identification.

CHAPTER 5
PROJECT DESIGN

5.1 DATA FLOW DIAGRAM



SOLUTION & TECHNICAL ARCHITECTURE



User Stories

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

CHAPTER 6
PROJECT PLANNING & SCHEDULING

SPRINT PLANNING AND 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 fromvarious resources with different handwritings.	10	Low	Dharsa.M.D Sneka dharshini.K
Sprint-1	Data Preprocessing	USN-2	As a user, I can load the dataset, handling the missing data, scaling and split dataintotr ain and test.	10	Medium	Arunasalam R, Bharathi.T
Sprint-2	Model Building	USN-3	As a user, I will get an application with MLmodel which provides high accuracy of recognized handwritten digit.	5	High	Bharathi.T,Sneka dharshini.K
Sprint-2	Add CNN layers	USN-4	Creating the model and adding the input,hidden, and output layers to it.	5	High	Dharsa.M.D , Arunasalam. R

Sprint	Functional Req uirement (Epic)	User Story Number	User Story/ Task	Story Poits	Priority	Team Members
Sprint-2	Compiling the model	USN-5	With both the training datadefined and model defined, it's time to configure thelearning process.	2	Medium	Sneka dharshini.K,Arunasalam.R
Sprint-2	Train & test themodel	USN-6	As a user, let us train our model with ourimage dataset.	6	Medium	Sneka dharshini.K,Bharathi.T
Sprint-2	Save the model	USN-7	As a user, the model is saved &integratedwith an android application or web application in order to predict something.	2	Low	Dharsa.M.D
Sprint-3	Buildig UI Application	USN-8	As a user, I will upload thehandwritten digitimage to the application by clicking a uploadbutton.	5	High	Bharathi.T
Sprint-3		USN-9	As a user, I can know the details of thefundamental usageof the application.	5	Low	Sneka dharshini.K
Sprint-3		USN-10	As a user, I can see the predicted / recognized digitsin the application.	5	Medium	Arunasalam .R
Sprint-4	Train the model onIBM	USN-11	As a user, I train the model on IBM andintegrate flask/Django with scoring endpoint.	10	High	Dharsa.M.D ,Sneka dharshini.K

Sprint-4	Cloud Deployment	USN-12	As a user, I can access the web applicati onand make the use of the product from anywhere.	10	High	Arunasalam.R, Bharathi.T

Sprint delivery plan:

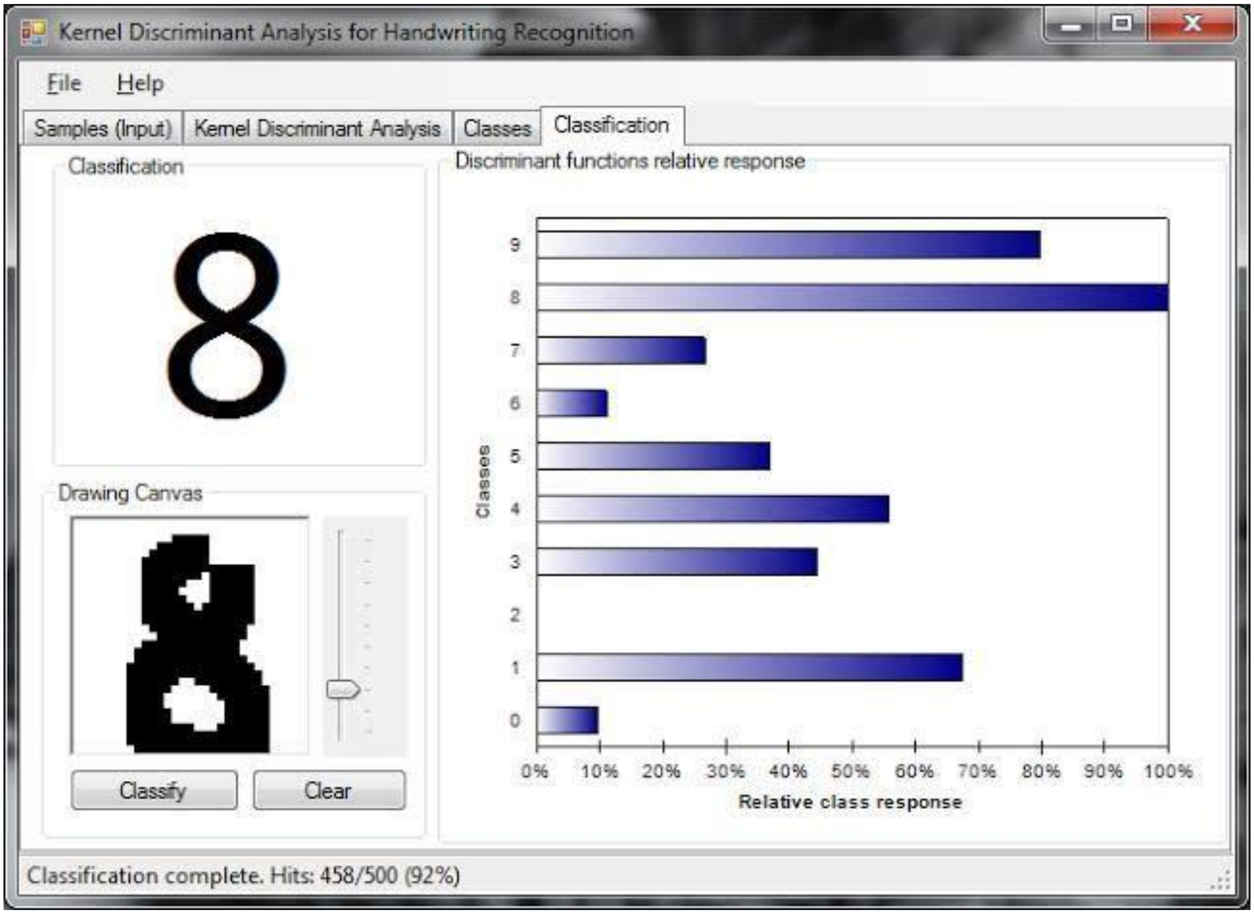
Sprint	Total Story Poins	Duration	Sprint Start Date	SprintEnd Date (Planned)	Story Points Completed (as on Planned EndDate)	Sprint Release Date (Actual)
Sprint –1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint –2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint –3	20	6 Days	07 Nov 2022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

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) periteration unit (story points per day)

Average Velocity= 20 / 6 = 3.33

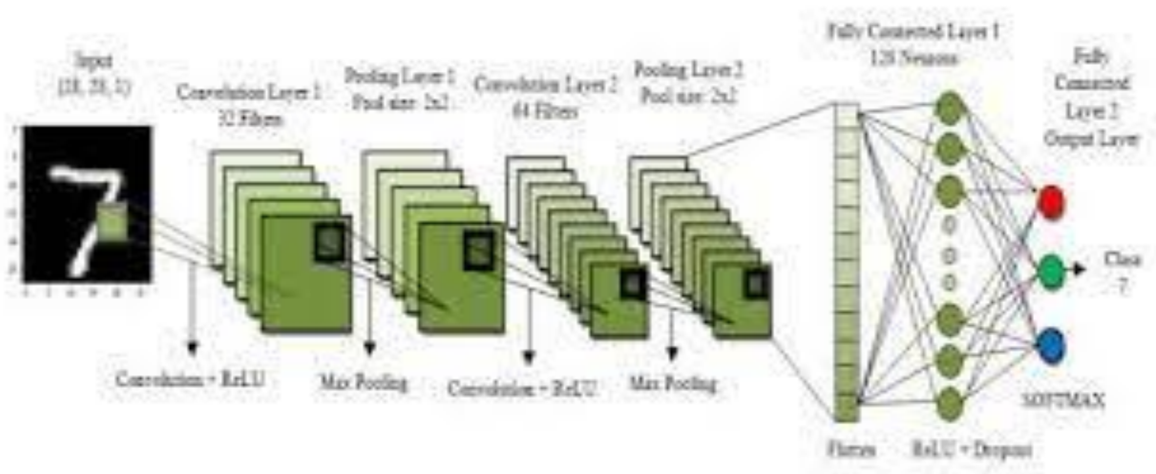
Reports from JIRA



CHAPTER 7
CODING & SOLUTIONING

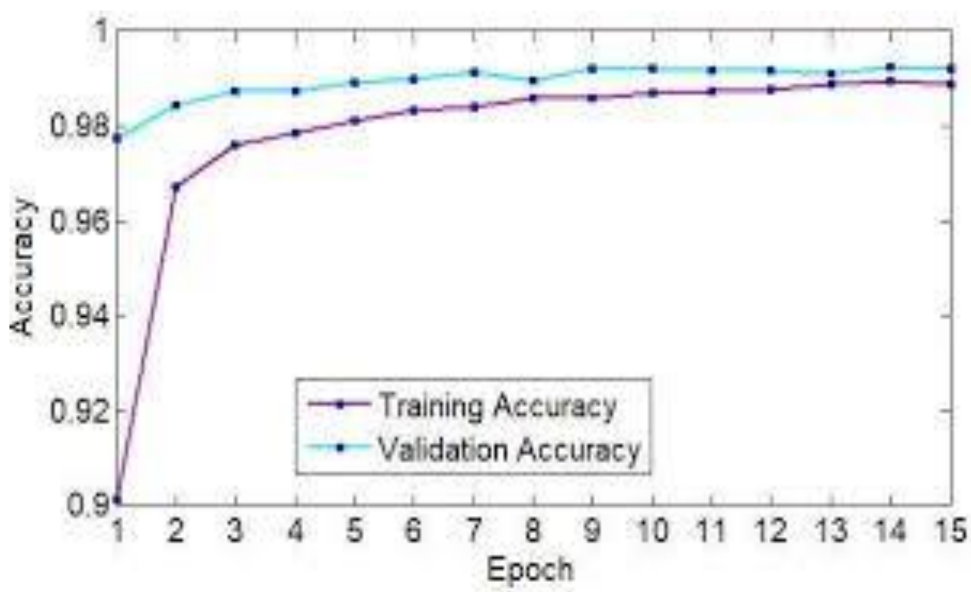
Feature 1

Depending on the features given to the classifier, it accumulates a knowledge base for classification purposes. In case of a binary image, when translated into an array and used as an attribute, no information is given to the classifier about the order of the attributes. It would in fact be irrelevant if all the patterns are shuffled in the same manner and presented to the classifier for classification purposes. If such information is given to the classifier, its performance can be improved. One such feature (Pixel Count Feature) is obtained by counting row-wise, number of black pixels present and doing same column-wise, thus obtaining two profiles. For example the row profile of dimensions (1xN) can be obtained from complemented binary image of pixels, where 0s and 1s represent white and black pixels. The implementation of Handwritten Digit Recognition by Convolutional Neural Network is done using Keras. CNN is a deep learning technique to classify the input image and essentially extracts 'useful' features from the input automatically. CNN is a reliable deep learning algorithm for an automated end-to-end prediction.



Feature 2

Row and Column Pixel Count (PC) Features: Two more features added to this set are variance of number of black pixels in rows and variance of number of black pixels in columns. Another useful feature is the character shape profile, where distance in pixels from an edge of the image to a blackpixel is found. This is done from all four edges, giving four profiles. Here, a lot of feature variation is observed even for similar patterns, which leads to lesser gain in classification accuracy. If only the sign of this difference [- , 0 , +] is taken into consideration, a superior feature is obtained, with a considerable gain in classification accuracy. For two successive pixels in a character profile, a southwest slant is given value 1, a southeast slant is given value 3, and a straight vertical or horizontal line is given value 2.This can be represent Thus the minor irregularities even in case of patterns. The proposed method uses k-nearest neighbor (knn) classification algorithm for classifying the MNIST digit images in test set using the feature vector of training database. The k-nearest neighbor algorithm (k-NN) is a classification technique which classify the objects base on training features space.The functionality of k-NN algorithm is to define the computations until classification is done irrespective of the learning techniques.



CHAPTER 8
TESTING

8.1 TESTCASE

Test case	Feature Type	Component	Test Scenario	Expected Result	Actual Result	Status
1	UI	Home Page	Verify UI elements inthe Home Page	The Home page must be displayedproperly	Working asexpected	PASS
2	UI	Home Page	Check if the UIelements are displayed properly in different screen sizes	The Home page must be displayed properlyin all sizes	The UI is notdisplayed properly in screen size 2560 x 1801 and 768 x 630	FAIL
3	Functional	Home Page	Check if usercan upload their file	The input imageshould be uploaded to theapplication successfully	Working asexpected	PASS
4	Functional	Home Page	Check if user cannot uploadunsupported files	The applicationshould not allowuser to select anon image file	User is able toupload any file	FAIL
5	Functional	Home Page	Check if the page redirectsto the result page once theinput is given	The page shouldredirect to theresults page	Working asexpected	PASS

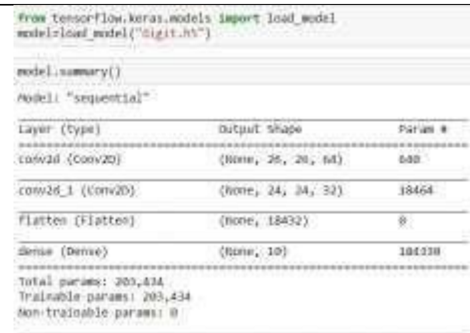
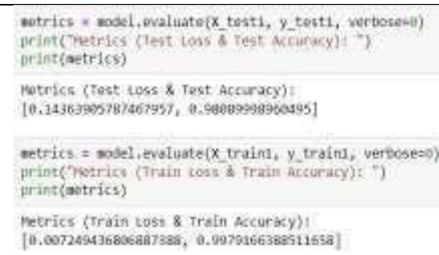
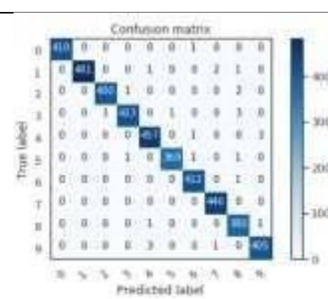
1	Funct ional	Backen d	Check if all theroutes are working properly	All the routes should properlywork	Working ase xpected	PASS
1	Funct ional	Model	Check if the model can handle variousimage sizes	The model shouldresc ale the imageand predict the results	Working ase xpected	PASS
2	Funct ional	Model	Check if themodel predicts thed igit	The model shouldpred ict the number	Working ase xpected	PASS
3	Funct ional	Model	Check if the model can handle complex inputimage	The model shouldpre dict the number in the complex image	The model failst o identify the digit since the model is not built to handlesuch dat a	FAIL
1	UI	Result P age	Verify UI elements inthe Result Page	The Result page must be display edproperly	Working ase xpected	PASS
2	UI	Result P age	Check if the input image isdisplay ed properly	The input image should be displayed properly	The size of thei nput image exceeds the display containe r	FAIL
3	UI	Result P age	Check if theresult is displayed properly	The result shouldbe displayed properly	Working ase xpected	PASS
4	UI	Result P age	Check if the other predictions ar e displayedproperly	The other predictions shouldbe displayed properly	Working ase xpected	PASS

User Acceptance Testing

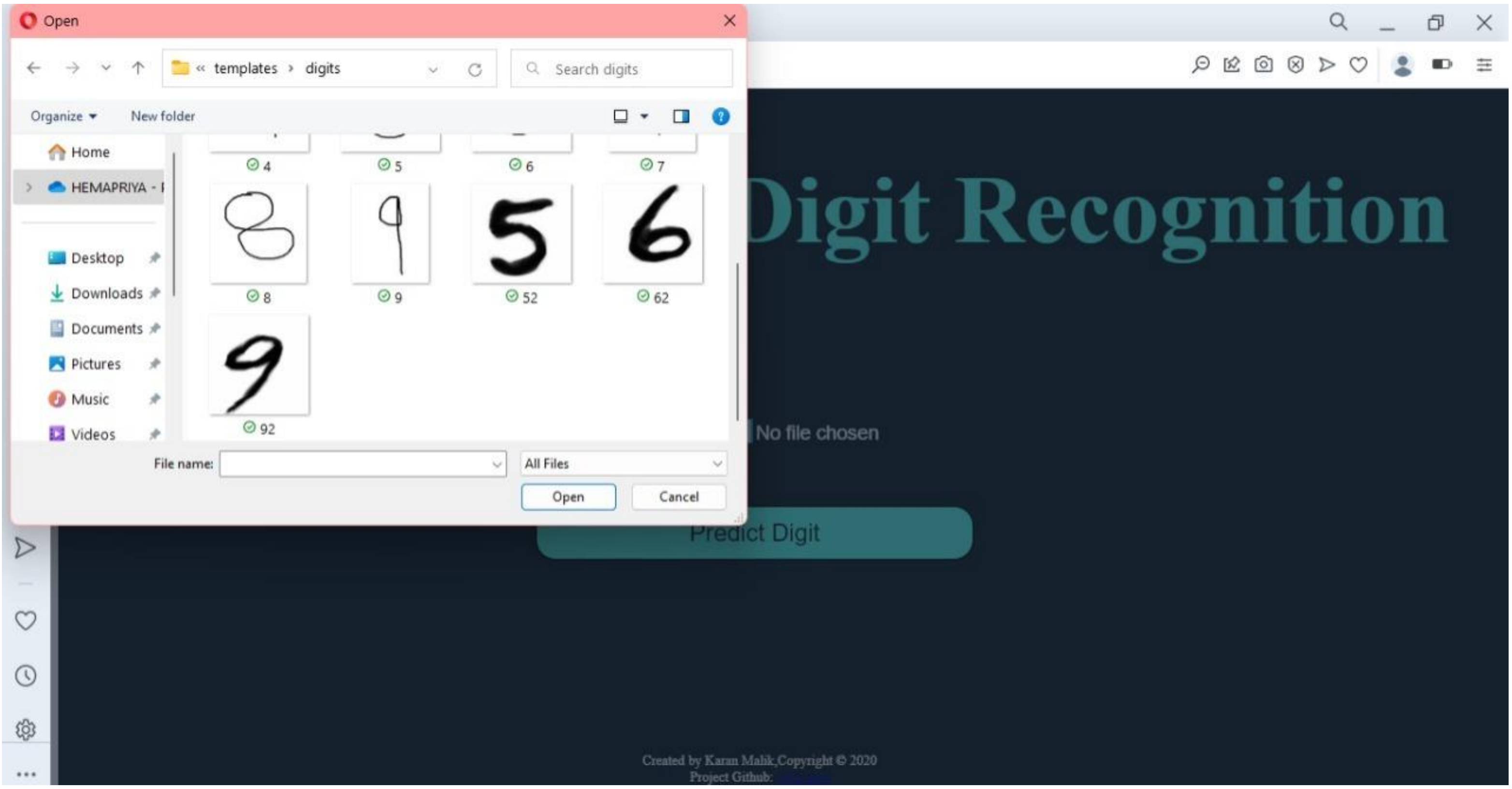
Features	Accuracy
0	80%
2	95%
8	74%
7	70%
4	71%

CHAPTER 9
RESULTS

1. Performance Metrics

S.No.	Parameter	Values	Screenshot
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	 from tensorflow.keras.models import load_model model=load_model("digit.h5") 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 -0.9879166388511658 Validation Accuracy -0.99089998960495	 metrics = model.evaluate(X_test1, y_test1, verbose=0) print("Metrics (Test Loss & Test Accuracy): ") print(metrics) Metrics (Test Loss & Test Accuracy): [0.14363905787467957, 0.98089998960495] metrics = model.evaluate(X_train1, y_train1, verbose=0) print("Metrics (Train Loss & Train Accuracy): ") print(metrics) Metrics (Train Loss & Train Accuracy): [0.007249436806887388, 0.9979166388511658]
3. `	Metrics	**Classification Model:** precision,recall,f1-score,support	
4.	Metrics	Confusion Matrix	

OUTPUT SCREENSHOTS


Open
templates
digits
Search digits
Organize
New folder
Home
HEMAPRIYA - I
Desktop
Downloads
Documents
Pictures
Music
Videos
4
5
6
7
8
9
52
62
92
File name:
All Files
Open
Cancel
Digit Recognition
No file chosen
Predict Digit
Created by Karan Malik,Copyright © 2020
Project Github:


127.0.0.1:5000/predict
GRE Mock Tests
GRE Verbal Practice...
23 Amazing Deep L...
IEEE report
IEEE Citation Guidel...
ieee-format.doc
Azure fundamental...
Feature Engineerin...
Handwritten Digit Recognition
Prediction: 5
Predict Again
Created by Karan Malik,Copyright © 2020
Project Github: Click here
Type here to search
ENG
21:27
08-07-2020

CHAPTER 10
ADVANTAGES & DISADVANTAGES

ADVANTAGES

- Reduces manual work
- More accurate than average human
- Capable of handling a lot of data
- Can be used anywhere from any device

DISADVANTAGES

- Cannot handle complex data
- All the data must be in digital format
- Requires a high performance server for faster predictions
- Prone to occasional errors

CHAPTER 11

CONCLUSION

This project demonstrated a web application that uses machine learning to recognise handwritten numbers. Flask, HTML, CSS, JavaScript, and a few other technologies were used to create this project. The model predicts the handwritten digit using a CNN network. During testing, the model achieved a 99.61% recognition rate. The proposed project is scalable and can easily handle a huge number of users. Since it is a web application, it is compatible with any device that can run a browser. This project is extremely useful in real-world scenarios such as recognizing number plates of vehicles, processing bank cheque amounts, numeric entries in forms filled up by hand (tax forms) and so on. There is so much room for improvement, which can be implemented in subsequent versions

CHAPTER 12

FUTURE SCOPE

This project is far from complete and there is a lot of room for improvement. Some of the improvements that can be made to this project are as follows:

- Add support to detect from digits multiple images and save the results
- Add support to detect multiple digits
- Improve model to detect digits from complex images
- Add support to different languages to help users from all over the world This project has endless potential and can always be enhanced to become better. Implementing this concept in the real world will benefit several industries and reduce the workload on many workers, enhancing overall work efficiency

13. APPENDIX

SOURCE CODE

MODEL CREATION:

```
from keras.datasets import mnist
import matplotlib.pyplot as plt
from keras.utils import np_utils
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D,Dense,Flatten
from tensorflow.keras.optimizers import Adam
(X_train,y_train),(
X_test,y_test) =mnist.load_data()
print(X_train.shape)
print(X_test.shape)
print(y_test.shape)
print(y_train.shape)
print("The label value is ",y_test[10]) #Value in y_test
plt.imshow(X_test[10])
print("The label value is ",y_test[65]) #Value in y_test
plt.imshow(X_test[65])
X_train.shape
X_test.shape
 X_train1 = X_train.reshape(60000, 28, 28, 1).astype('float32')
X_test1 = X_test.reshape(10000, 28, 28, 1).astype('float32')
number_of_classes= 10
y_train1 = np_utils.to_categorical(y_train,number_of_classes)
y_test1 = np_utils.to_categorical(y_test,number_of_classes)
print("After encoding the value",y_test[10] ,"become", y_test1[10])
```

```
print("After encoding the value",y_test[100] ,"become", y_test1[100])
print("After encoding the value",y_test[65] ,"become", y_test1[65])
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"])
model.fit(X_train1, y_train1, batch_size=32, epochs=5, validation_data=(X_test1,y_test1))
metrics = model.evaluate(X_test1, y_test1, verbose=0)
print("Metrics (Test Loss & Test Accuracy): ")
print(metrics)
prediction = model.predict(X_test1[:4])
print(prediction)
import numpy as np
print(np.argmax(prediction, axis=1))
print(y_test1[:4])
model.save("model.h5")
from tensorflow.keras.models import load_model
model=load_model("model.h5")
model.summary()
```

FLASK APP:

```
import numpy as np

import os

from PIL import Image

from flask import Flask, request, render_template, url_for

from werkzeug.utils import secure_filename, redirect
```

```
#from gevent.pywsgi import WSGIServer

from keras.models import load_model

from keras.preprocessing import image

from flask import send_from_directory

UPLOAD_FOLDER = 'D:/ibm/data

app = Flask(__name__)

app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

model = load_model("./models/mnistCNN.h5")

@app.route('/')

def index():

    return render_template('index.html')

@app.route('/predict', methods=['GET', 'POST'])

def upload():

    if request.method == "POST":

        f = request.files["image"]

        filepath = secure_filename(f.filename)

        f.save(os.path.join(app.config['UPLOAD_FOLDER'], filepath))

        upload_img = os.path.join(UPLOAD_FOLDER, filepath)

        img = Image.open(upload_img).convert("L")  # convert image to
monochrome

        img = img.resize((28, 28))  # resizing of input image
```

```
        im2arr = np.array(img)  # converting to image

        im2arr = im2arr.reshape(1, 28, 28, 1)  # reshaping according to our
requirement

        pred = model.predict(im2arr)

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

        return render_template('predict.html', num=str(num[0]))

if___name___== '_main_':

    app.run(debug=True, threaded=False)
```

RECOGNIZER(PYTHON):

```
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
import cv2
def recognize(image: bytes) -> int:
    """
    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/digit.h5"))
    image = cv2.imread(image)
```

```
grey = cv2.cvtColor(image.copy(), cv2.COLOR_BGR2GRAY)
ret, thresh = cv2.threshold(grey.copy(), 75, 255, cv2.THRESH_BINARY_INV)
contours, _ = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
preprocessed_digits = []
for c in contours:
    x,y,w,h = cv2.boundingRect(c)
    cv2.rectangle(image, (x,y), (x+w, y+h), color=(0, 255, 0), thickness=2)
    digit = thresh[y:y+h, x:x+w]
    resized_digit = cv2.resize(digit, (18,18))
     padded_digit = np.pad(resized_digit, ((5,5),(5,5)), "constant", constant_values=0)
    preprocessed_digits.append(padded_digit)
for digit in preprocessed_digits:
    prediction = model.predict(digit.reshape(1, 28, 28, 1))
    best= np.argmax(prediction)
return best, "1.jp
```

FIRST PAGE(HTML)

```
  <!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Handwritten Recognition System</title>
    <link rel="stylesheet" href="style.css">
<style>
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    background-image: url(db-bg.jpg);
    background-repeat: no-repeat;
    background-size: cover;
    height: 100vh;
}

.header {
    background-color: lightskyblue;
    opacity: 0.9;
    font-size: 20px;
    padding: 10px;
```

```
    position: sticky;
}

.navbar {
    text-decoration: none;
}

ul {
    display: flex;
    flex-direction: row;
    justify-content: flex-end;
    gap: 20px;
    list-style-type: none;
}

a {
    color: white;
    letter-spacing: 1px;
    text-decoration: none;
    padding: 10px;
    font-weight: 700;
}

a:hover {
    color: darkblue;
    cursor: pointer;
}

.main {
    margin: 40px;
}

.main-heading {
    color: whitesmoke;
    text-align: center;
    letter-spacing: 1.5;
    margin: 50px 0 0px;
}

.content {
    color: white;
    font-size: 20px;
    font-weight: 500;
    text-align: center;
    line-height: 1.5;
    margin-top: 150px;
}
</style>
</head>

<body>
    <header class="header">
        <nav class="navbar">
            <ul>
```

```
            <li>
                <a href="#">Home</a>
            </li>
            <li>
                <a href="second.html">Recognize</a>
            </li>
        </ul>
    </nav>
</header>

<div class="bg-pic"></div>

<main class="main">
    <h1 class="main-heading">Handwritten Recognition System</h1>

    <p class="content">
        <em>
            Handwritten Text Recognition is a technology that is much needed in this world as of today. This digit
            Recognition system is used to recognize the digits from different sources like emails, bank cheque,
            papers, images, etc. Before proper implementation of this technology we have relied on writing texts
            with our own hands which can result in errors. It's difficult to store and access physical data with
            efficiency. The project presents recognizing the handwritten digits (0 to 9) from the famous MNIST
            dataset. Here we will be using artificial neural networks convalution neural network.
        </em>
    </p>
</main>
</body>

</html>
```

SECOND PAGE (HTML)

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Digit Recognition</title>
    <link rel="stylesheet" href="recognize.css">

<style>
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    background-image: url("bg-img.jpg");
    background-repeat: no-repeat;
```

```
    background-size: cover;
    width: 100%;
    height: 100vh;
}

.header {
    font-size: 20px;
    padding: 10px;
    background-color: lightgray;
    width: 100%;
    opacity: 0.9;
}

.navbar {
    text-decoration: none;
}


ul {
    display: flex;
    flex-direction: row;
    justify-content: flex-end;
    gap: 20px;
    list-style-type: none;
}

a {
    color: black;
    letter-spacing: 1px;
    text-decoration: none;
    padding: 10px;
    font-size: 20px;
    font-weight: 700;
}

a:hover {
    color: darkcyan;
    cursor: pointer;
}

.main {
    margin: 80px;
}

.main-heading {
    color: darkcyan;
    letter-spacing: 1.5px;
    margin-bottom: 20px;
}

.flex-btn {
    display: flex;
    flex-direction: row;
    gap: 20px;
```

```
}

label {
   background-color: darkcyan;
   color: white;
   padding: 10px;
   border: none;
   border-radius: 3px;
   cursor: pointer;
}

.recognize-btn {
   border: none;
   padding: 10px;
   border-radius: 3px;
   background-color: darkcyan;
   color: white;
}

label:hover,
.recognize-btn:hover {
   cursor: pointer;
   background-color: lightblue;
   color: darkblue;
}

input {
   margin-top: 1rem;
}

input[type="file"] {
   z-index: -1;
   position: absolute;
   opacity: 0;
}

input:focus+label {
   outline: 2px solid;
}
</style>
</head>

<body>
    <header class="header">
        <nav class="navbar">
            <ul>
                <li>
                    <a href="first.html">Home</a>
                </li>
                <li>
                    <a href="#">Recognize</a>
                </li>
            </ul>
        </nav>
```

```
    </header>

    <main class="main">
        <h1 class="main-heading">Digit Recognition</h1>
        <br>
        <div class="flex-btn">
            <input type="file" id="file-upload" multiple required />
            <label for="file-upload">Choose</label>
            <div id="file-upload-filename"></div>
            <br><br>
            <button class="recognize-btn">Recognize</button>
        </div>
    </main>

    <script src="recognize.js"></script>

<script>
var input = document.getElementById('file-upload');
var infoArea = document.getElementById('file-upload-filename');

input.addEventListener('change', showFileName);

function showFileName(event) {
    var input = event.srcElement;
    var fileName = input.files[0].name;
    infoArea.textContent = 'File name: ' + fileName;
}
</script>
</body>

</html>
```

GITHUB

https://github.com/IBM-EPBL/IBM-Project-54203-1661777480

VIDEO LINK **https://www.awesomescreenshot.com/video/12680602?key=3d8b6e99c8b112357a438c3c59a30092**