A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM

A PROJECT REPORT

Submitted by :

S SWETHA	**511319104074 (Team Leader)**
M NANDHINI	**511319104052**
R LIKITHA	**5113191040441**
D K MOHANA SUNDARI	**511319104050**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

KINGSTON ENGINEERING COLLEGE, VELLORE

ANNA UNIVERSITY: CHENNAI 600 025

Executed By:

IBM

CONTENT

ABSTRACT

Image recognition is widely used in the field of computer vision today. As a kind of image recognition, digit recognition is widely used. Today, the online recognition technology in digit recognition is relatively mature while the offline recognition technology is not. This paper mainly introduces an offline recognition system for handwritten digits based on convolutional neural networks. The system uses the MINST dataset as a training sample and pre-processes the picture with the Opencv toolkit. Then it uses LeNet-5 in the convolutional neural network to extract the handwritten digit image features, repeatedly convolution pooling, and pull the result into a one-dimensional vector. And finally find the highest probability point to determine the result to achieve handwritten digit recognition with the Softmax regression model. The application of this system can greatly reduce labor costs and improve work efficiency, which is of great significance in many fields.

CHAPTER 1

1.INTRODUCTION

1.1 OVERVIEW

The handwritten digit recognition is the ability to recognize human handwritten digits by computers. It is considered to be a hard task for the machine because handwritten digits are not perfect and can be made with many different techniques. A solution to this problem is handwritten digit recognition which uses the image of a digit and thereby recognizes the digit present in it.For that we implemented Deep learning approach to overcome this issue the handwritten digit recognition problem becomes one of the most notorious problems in machine literacy and computer vision operations. numerous machine literacy ways have been employed to break the handwritten number recognition problem. This paper focuses on Neural Network(NN) approaches. The three most popular NN approaches are deep neural network(DNN), deep belief network(DBN) and convolutional neural network(CNN). In this paper, the three NN approaches are compared and estimated in terms of numerous factors similar as delicacy and performance. Recognition delicacy rate and performance, still, isn't the only criterion in the evaluation process, but there are intriguing criteria similar as prosecution time. Random and standard dataset of handwritten number have been used for conducting the trials. The results show that among the three NN approaches DNN is the most accurate algorithm, still the prosecution time of DNN is similar with the other two algorithms.

1.2 PURPOSE

The handwritten digit recognition is the ability of computers to recognize human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different flavors. The handwritten digit recognition is the solution to this problem which uses the image of a digit and recognizes the digit present in the image.

CHAPTER 2

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

In previously, the handwritten digit are recoginized by machince by using various methods, but it is very hard to find the accurate digits by the machince because of the digits fonts and styles.

2.2 REFERENCES

Author Name : B. El Kessab1, C. Daoui1, B. Bouikhalene2, M. Fakir2 and K. Moro2

Abstract: This paper deals with an optical character recognition (OCR) system of handwritten digit, with the use of neural networks (MLP multilayer perceptron). And a method of extraction of characteristics based on the digit form, this method is tested on the MNIST handwritten isolated digit database (60000 images in learning and 10000 images in test). This work has achieved approximately 80% of success rate for MNIST database identification.

Author Name : Ali Abdullah Yahya 1,*, Jieqing Tan

Abstract: An enormous number of CNN classification algorithms have been proposed in the literature. Nevertheless, in these algorithms, appropriate filter size selection, data preparation, limitations in datasets, and noise have not been taken into consideration. As a consequence, most of the algorithms have failed to make a noticeable improvement in classification accuracy. To address the shortcomings of these algorithms, our paper presents the following contributions: Firstly, after taking the domain knowledge into consideration, the size of the effective receptive field (ERF) is calculated. Calculating the size of the ERF helps us to select a typical filter size which leads to enhancing the classification accuracy of our CNN. Secondly, unnecessary data leads to misleading results and this, in turn, negatively affects classification accuracy. To guarantee the dataset is free from any redundant or irrelevant variables to the target variable, data preparation is applied before implementing the data classification mission. Thirdly, to decrease the errors of training and validation, and avoid the limitation of datasets, data augmentation has been proposed. Fourthly, to simulate the real-world natural influences that can affect image quality, we propose to add an additive white Gaussian noise with s = 0.5 to the MNIST dataset. As a result, our CNN algorithm achieves state-of-the-art results in handwritten digit recognition, with a recognition accuracy of 99.98%, and 99.40% with 50% noise.

Author Name : Shruti Luthra Dinkar Arora

Abstract:

Large number of techniques for keyword extraction have been proposed for better matching of documents with the user’s query but most of them deal with tf-idf to find the weight age of query terms in the entire document but this can result in improper result as if a term has a low term frequency in overall document but high frequency in a certain part of the document then that term can be ignored by

traditional tf-idf method. Through this paper, the keyword extraction is improved using a hybrid technique in which the entire document is split into multiple domains using a master keyword and the frequency of all unique words is found in every domain . The words having high frequency are selected as candidate keywords and the final selection is made on the basis of a graph which is constructed between the keywords using Word Net. The experiments, conducted on various documents show that proposed approach outperforms other keyword extraction methodologies by enhancing document retrieval.

2.3 PROBLEM STATEMENT DEFINITION

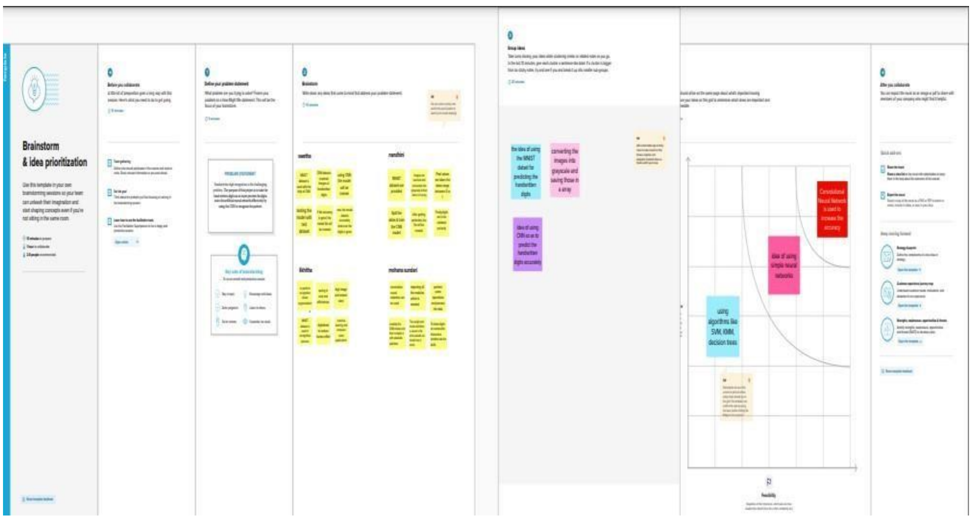
The goal of this project is to create a model that will be able to recognize and determine the handwritten digits from its image by using the concepts of Convolution Neural Network. Though the goal is to create a model which can recognize the digits, it can be extended to letters and an individual's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

CHAPTER 3

3.IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

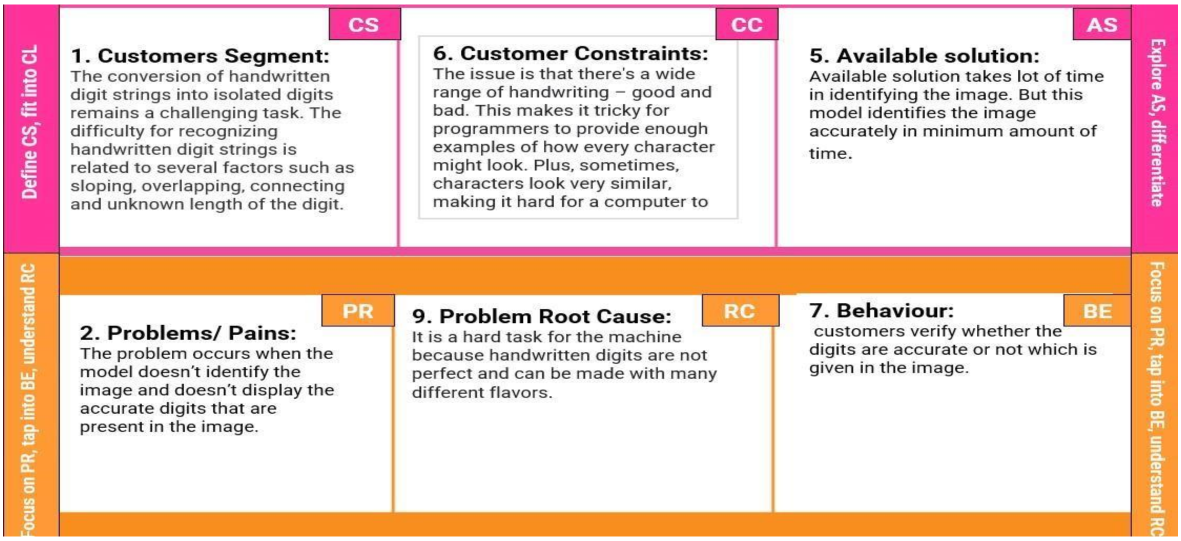
3.2 IDEATION & BRAINSTORMING

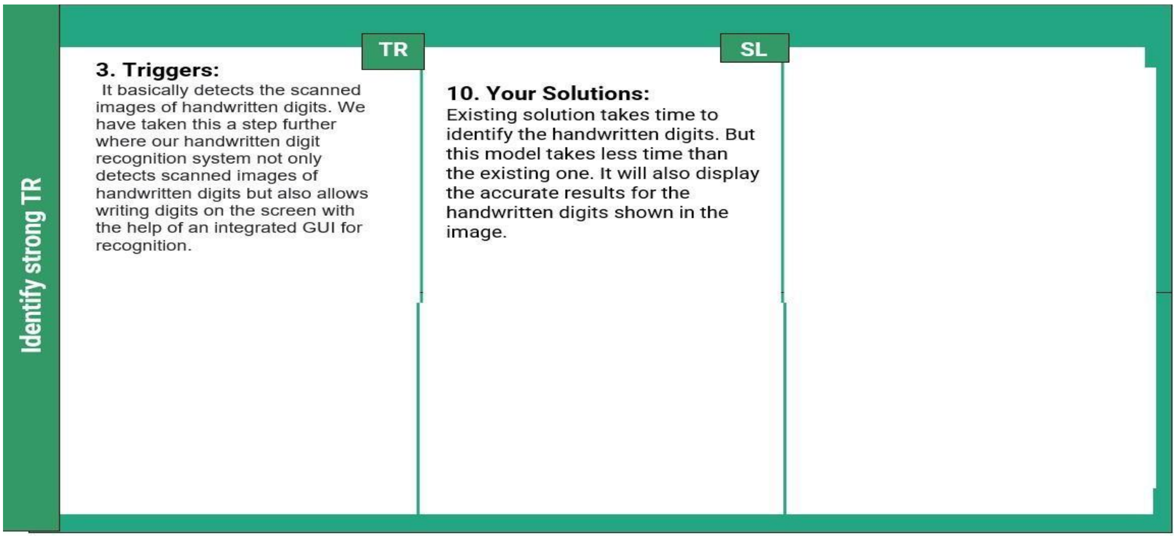


3.3 PROPOSED SOLUTION

We have used CNN (Convolutional Neural Networks) method ,It is an Deep learning Algorithms . A solution to this problem is handwritten digit recognition which uses the image of a digit and thereby recognizes the digit present in it.For it better recoginition.

3.4 PROPOSED SOLUTION FIT





CHAPTER 4

4.REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

Following are the functional requirements of the proposed solution.

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	User Login	Login through Google Login through Email
FR-4	Choose package	Selection of desired package
FR-5	Generate the daily plan	Daily plans will be generated by dietician
FR-6	Manage progress report	Gathering information from database and generating report
FR-7	Query	The user can ask for changes in plan

4.2 NON-FUNCTIONAL REQUIREMENTS

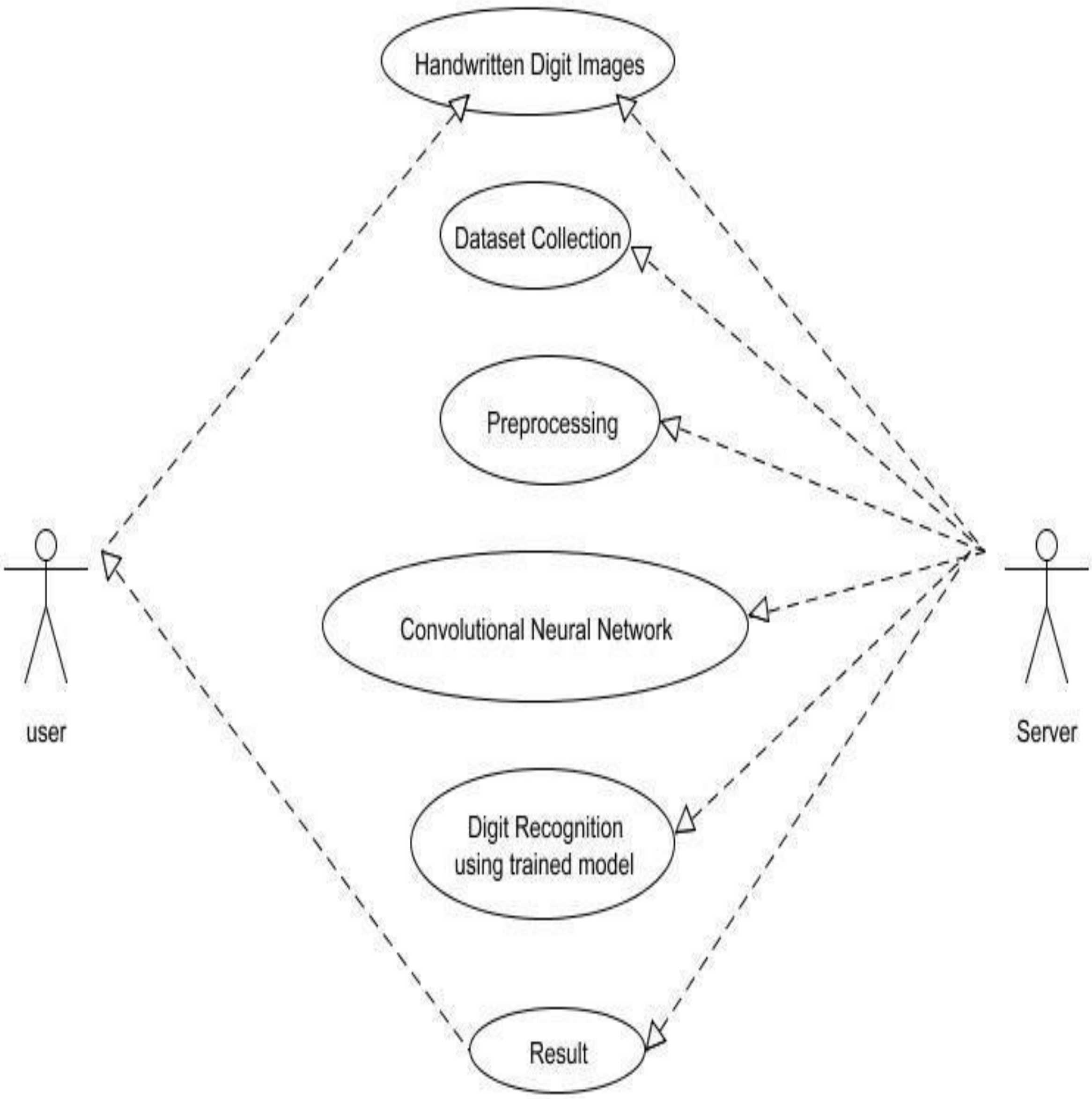
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Easy to use with interactive User Interface
NFR-2	Security	User can access only their personal information and not that of other users.
NFR-3	Reliability	The average time of failure shall be 7 days.
NFR-4	Performance	The results has to be shown within 10 sec
NFR-5	Availability	The dietician shall be available to users 24 hours a day, 7 days a week.
NFR-6	Scalability	Supports various food items

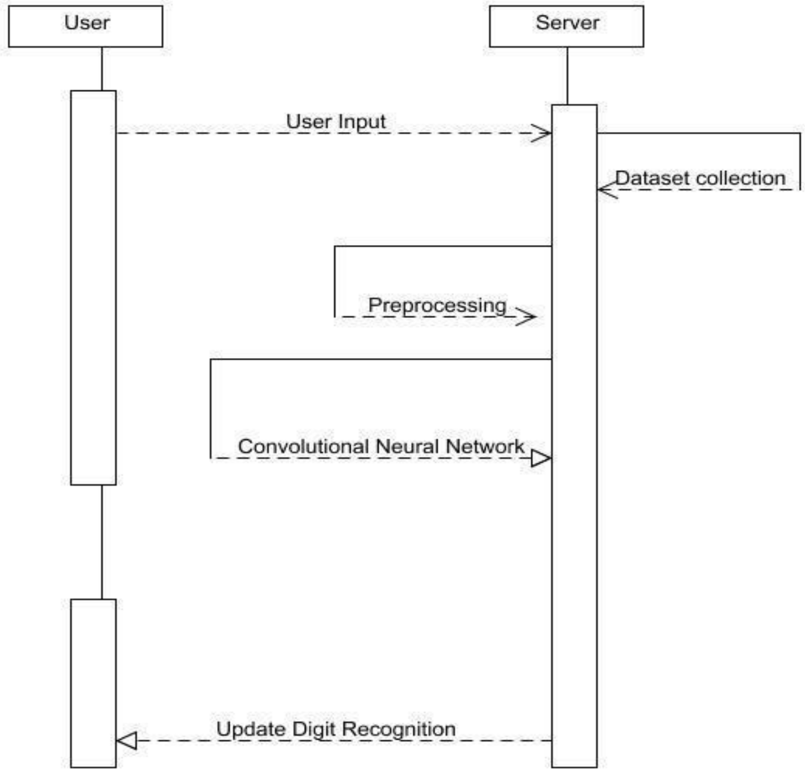
CHAPTER 5

5. PROJECT DESIGN

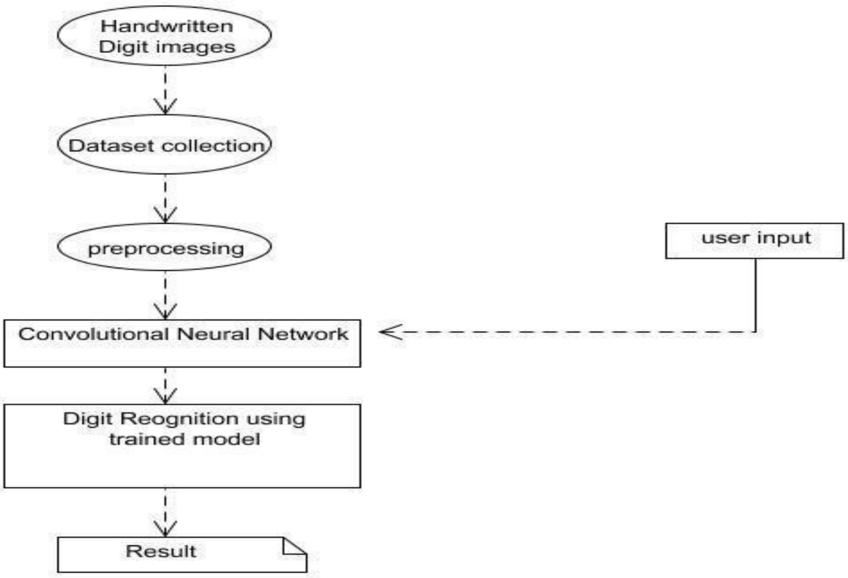
5.1 DATA FLOW DIAGRAMS USECASE DIAGRAM:



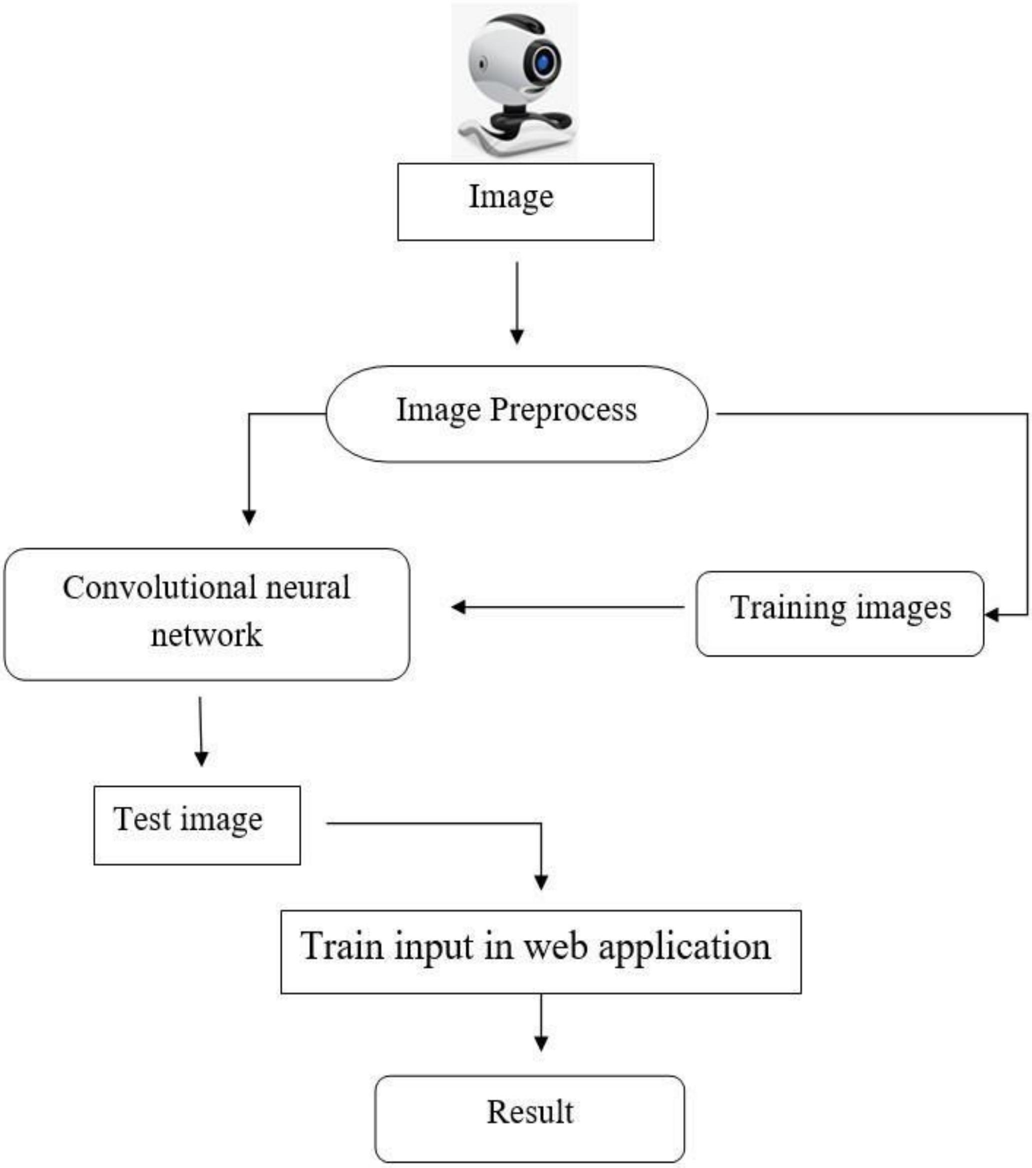
SEQUENCE DIAGRAM:



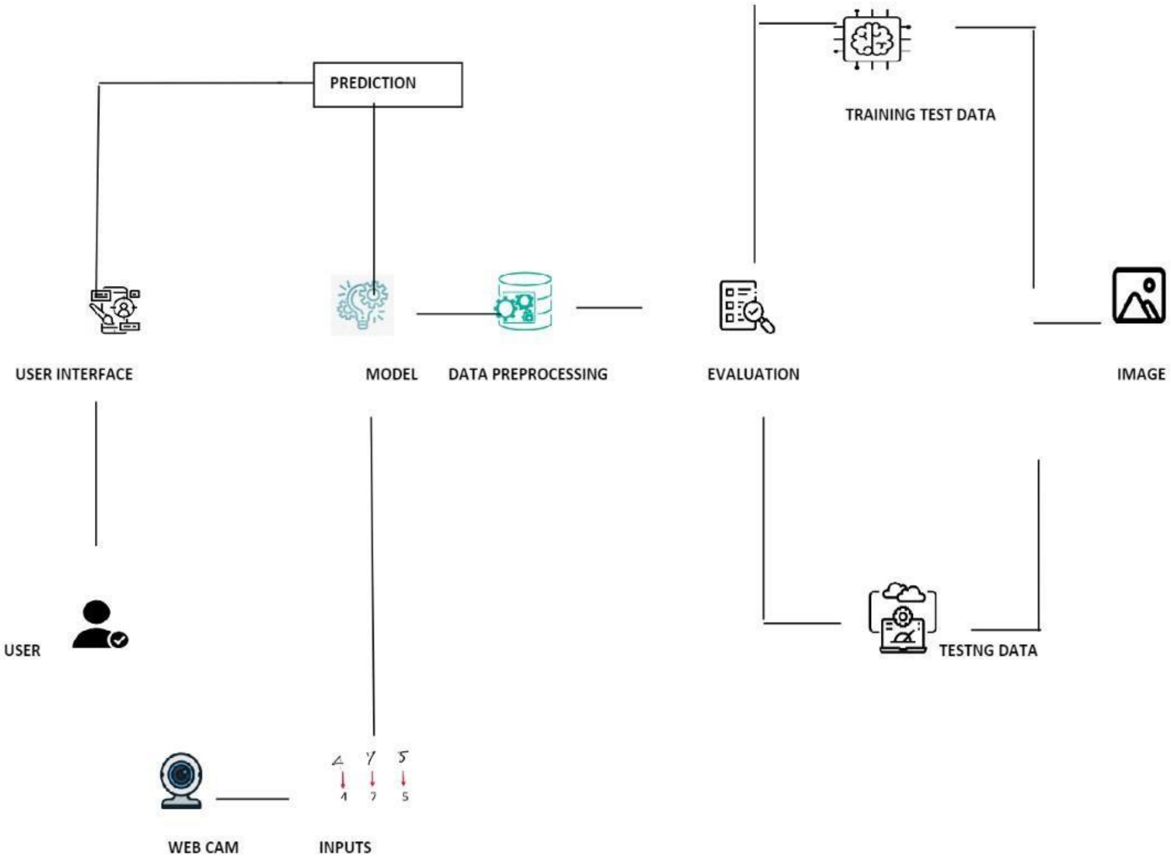
ACTIVITY DIAGRAM:



5.2 SOLUTION AND TECHNICAL ARCHITECTURE



TECHNICAL ARCHITECTURE:



5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password and confirming my password.	I can access my account / dashboard	High	Sprint1

		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint1
		USN-3	As a user, I can register for the application through google	I can register & access the dashboard with google Login	Low	Sprint2
		USN-4	As a user, I can register for the application through email and password	I can register into the application with email & password	Medium	Sprint1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can login into the application with email & password	High	Sprint1
	Dashboard	USN-5	As a user, I can access any of the options available there.	I can access my resource	High	Sprint3
Administrator	prediction	USN-1	Here the model will predict the image using deep learning algorithms Such as CNN	In this I can have correct prediction on the particular algorithms.	High	Sprint3

CHAPTER 6

6. PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

Sprint Schedule, and Estimation

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task
Sprint-1	registration	USN-1	As a user, I can register for the application by entering my email, password and confirming my password.
Sprint-1	Login	USN-2	As a user, I can log into the application by entering email & password.
Sprint-2	Registration	USN-3	As a user, I can register for the application through google.
Sprint-3	Dashboard	USN-4	As a user, I can access any of the options available there.
Sprint-3	prediction	USN-5	Here the model will predict the image using deep learning algorithms Such as CNN
Sprint-3	Feature Extraction	USN-6	As a user, I can input any of the image of handwritten digit in the upload field and will get the results of the image.
Sprint-4	Train & Deployment of model in IBM cloud	USN-7	As a user, I can access the web application and make the use of the product from anywhere.

6.2 SPRINT DELIVERY SCHEDULE

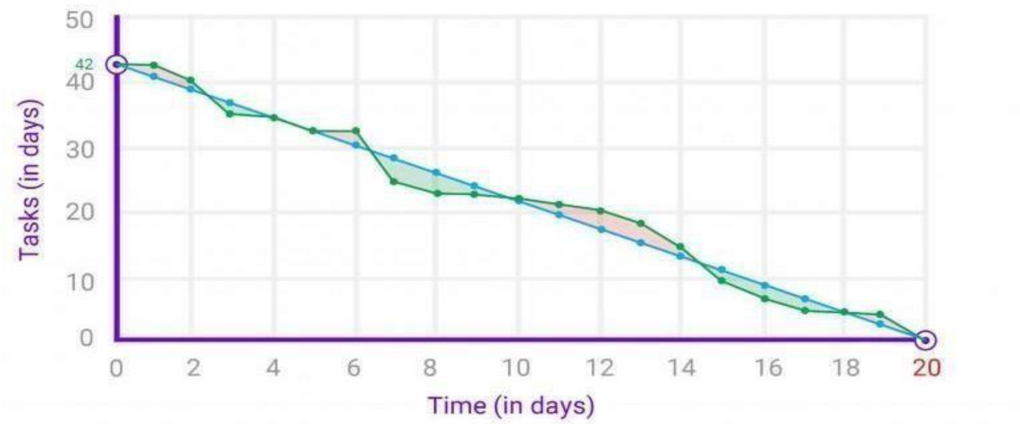
Sprint	Total Story Points	Duration	Sprint start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 oct 2022	5	29 oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	8	05 Nov 2022
Sprint-3	20	6 Days	07 Oct 2022	12 Nov 2022	8	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	16	19 Nov 2022

6.3 REPORTS FROM JIRA 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) per iteration unit (story points per day) Average Velocity = 20 / 6 = 3.33

BURNDOWN CHART:

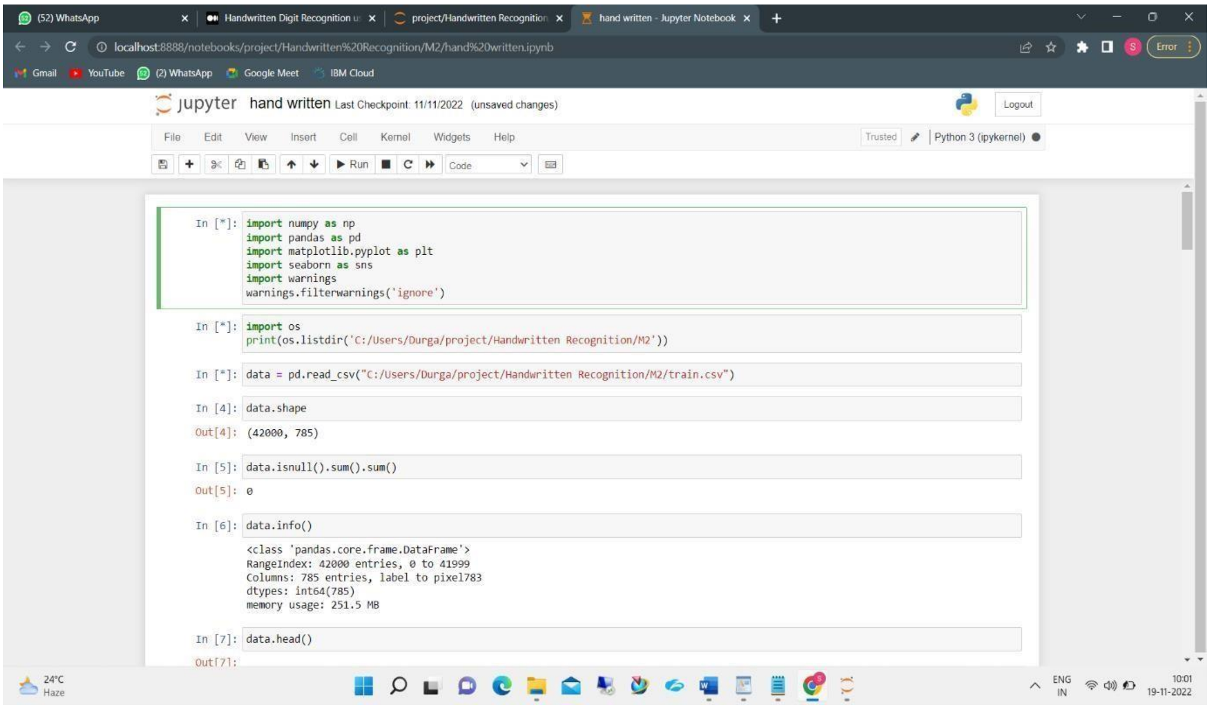
A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

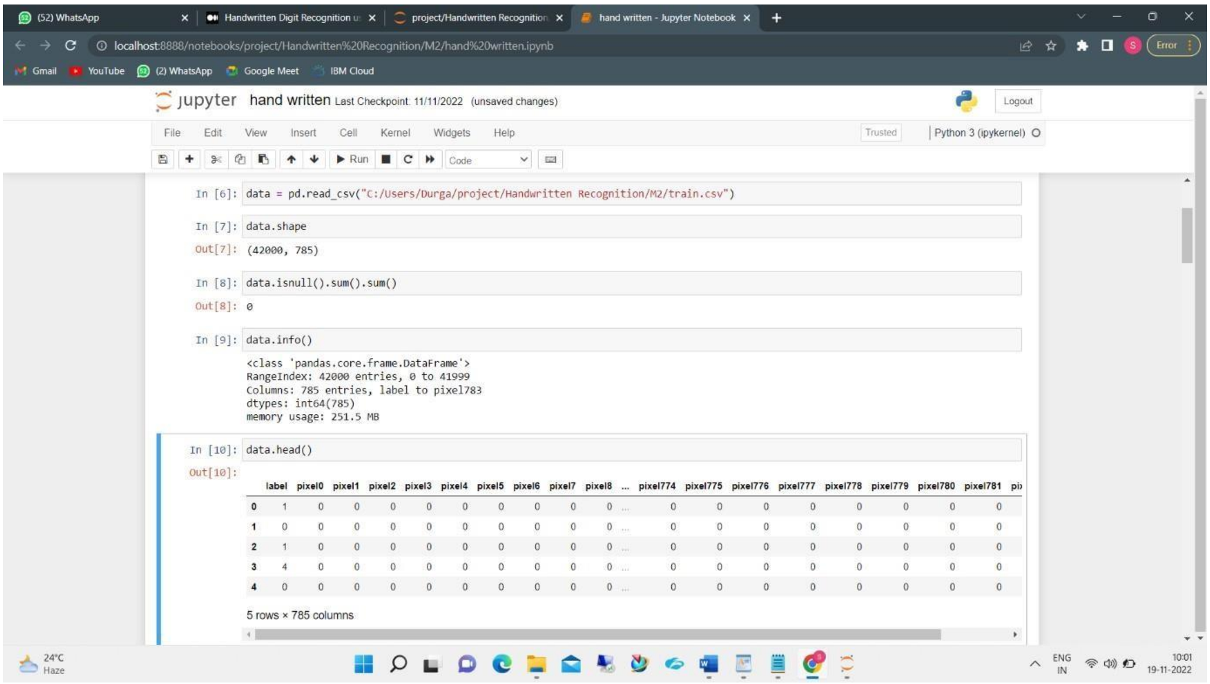


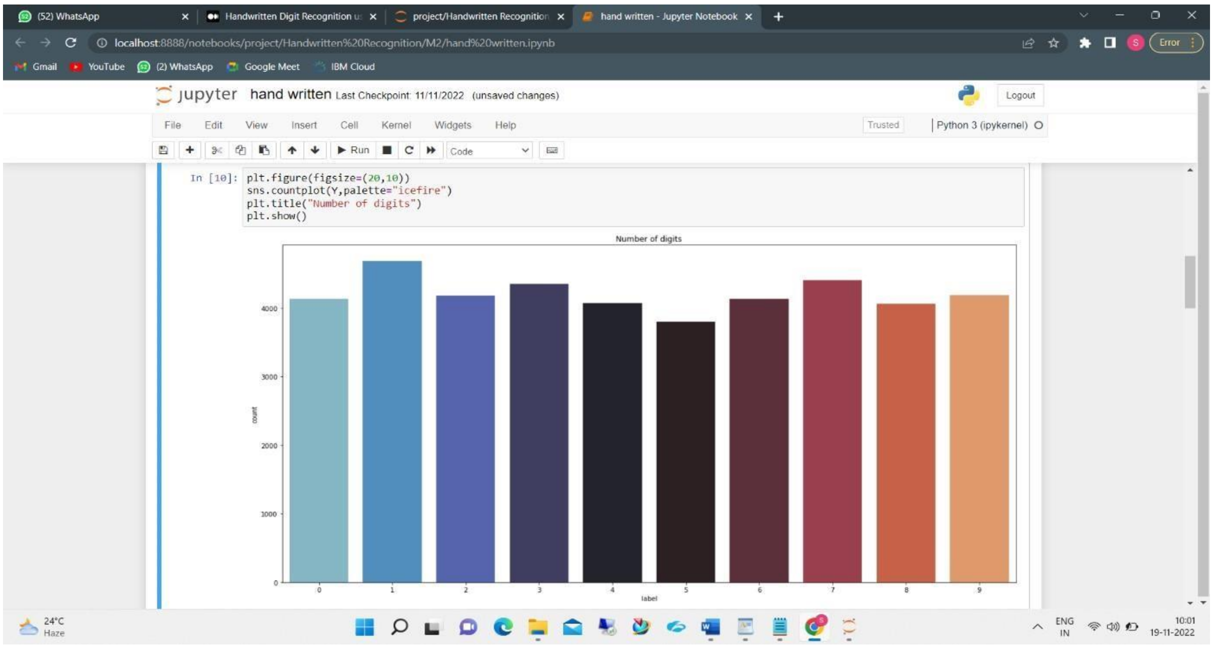
CHAPTER 7

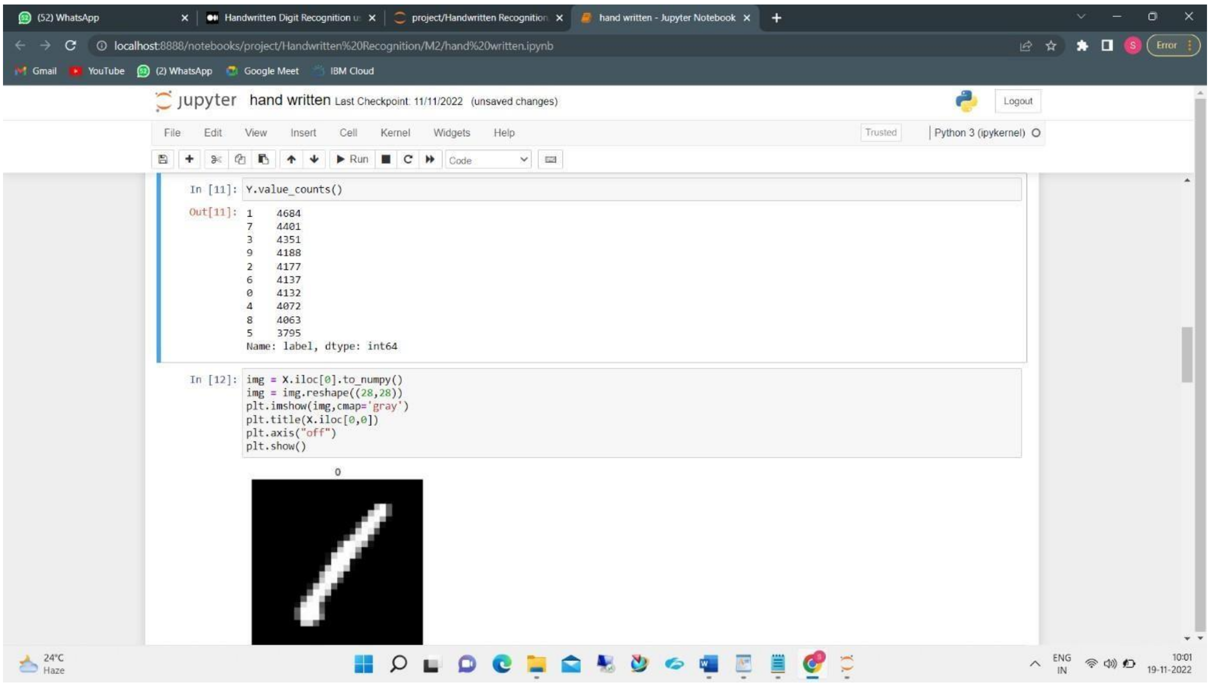
7.CODING AND SOLUTIONING

7.1 FEATURE 1

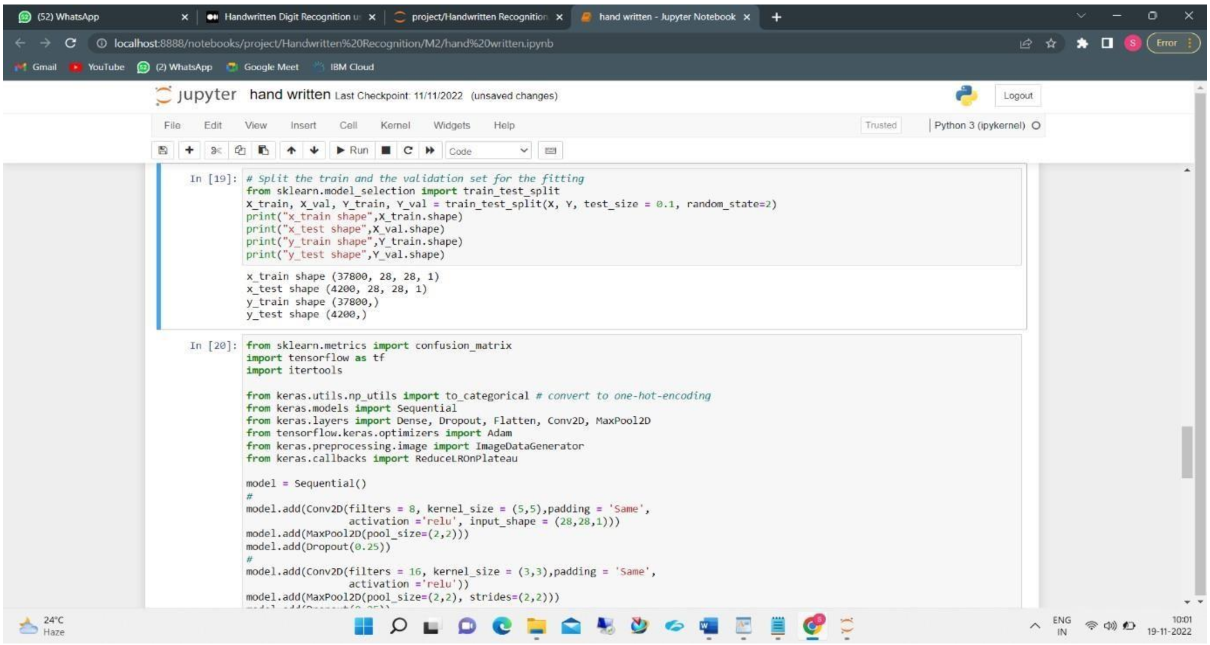




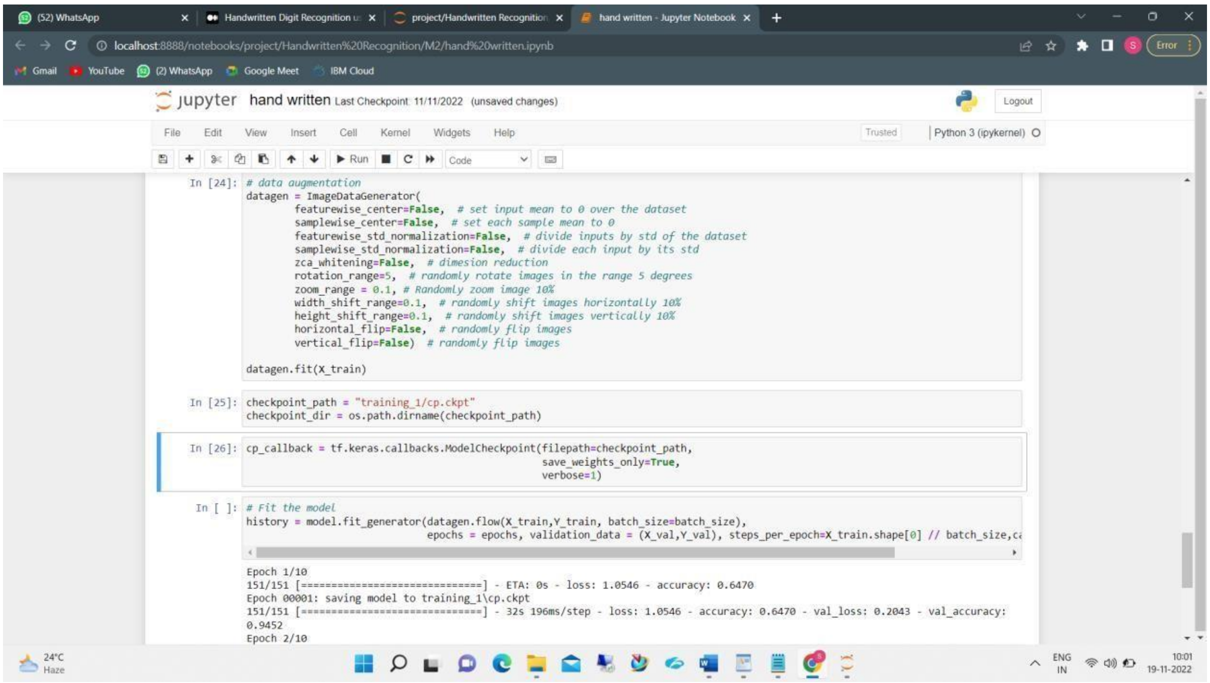

jupyter hand written Last Checkpoint: 11/11/2022 (unsaved changes)
In [10]:
plt.figure(figsize=(20,10))
sns.countplot(Y,palette="icefire")
plt.title("Number of digits")
plt.show()
Number of digits
count
label
0
1
2
3
4
5
6
7
8
9
1000
2000
3000
4000


jupyter hand written Last Checkpoint: 11/11/2022 (unsaved changes)
In [11]: Y.value_counts()
Out[11]:
1 4684
7 4401
3 4351
9 4188
2 4177
6 4137
0 4132
4 4072
8 4063
5 3795
Name: label, dtype: int64
In [12]:
img = X.iloc[0].to_numpy()
img = img.reshape((28,28))
plt.imshow(img,cmap='gray')
plt.title(X.iloc[0,0])
plt.axis("off")
plt.show()
0

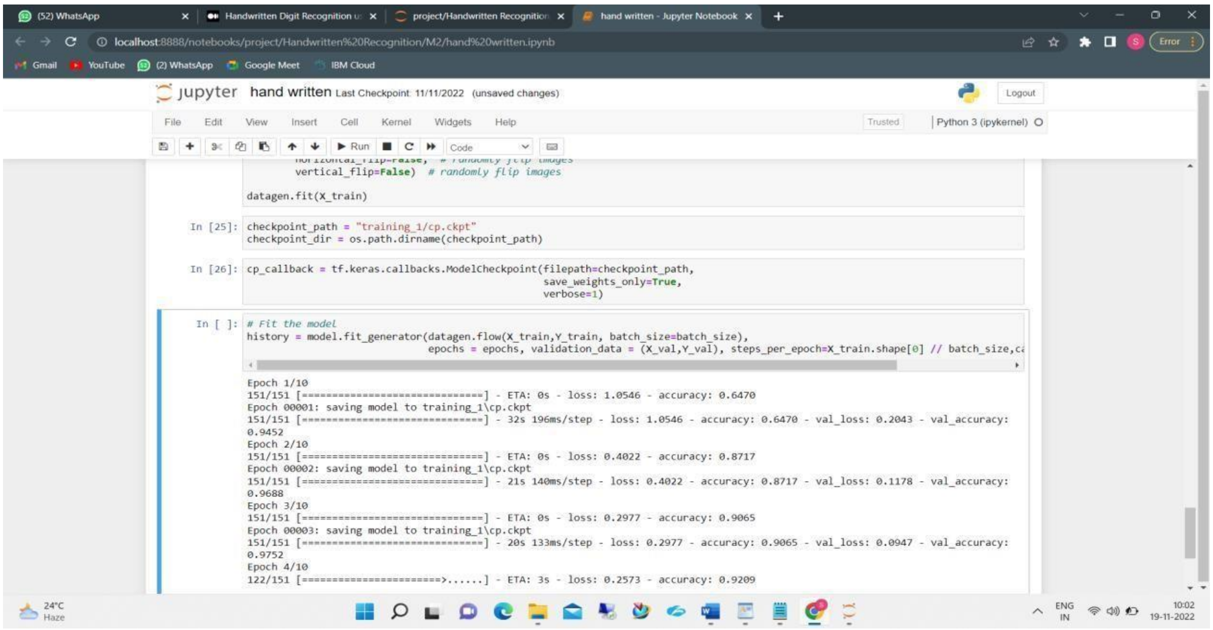
7.2 FEATURE 2

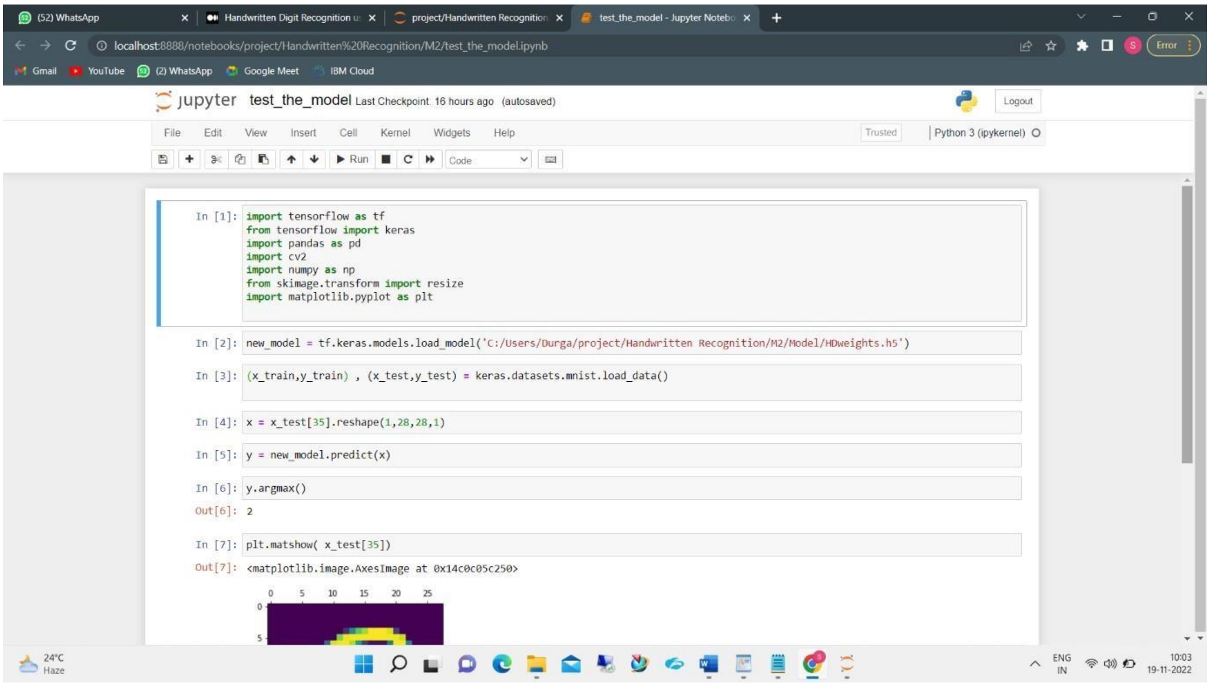


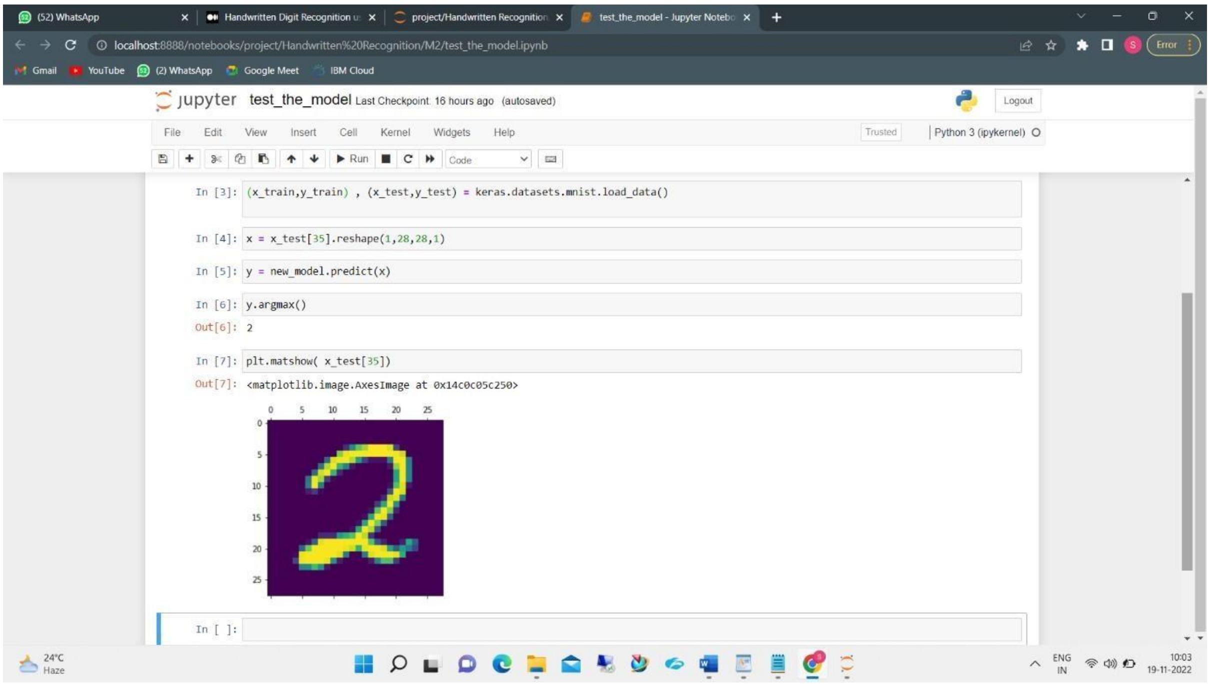
hand written Last Checkpoint: 11/11/2022 (unsaved changes)
In [19]: # Split the train and the validation set for the fitting
from sklearn.model_selection import train_test_split
X_train, X_val, Y_train, Y_val = train_test_split(X, Y, test_size = 0.1, random_state=2)
print("x_train shape",X_train.shape)
print("x_test shape",X_val.shape)
print("y_train shape",Y_train.shape)
print("y_test shape",Y_val.shape)
x_train shape (37800, 28, 28, 1)
x_test shape (4200, 28, 28, 1)
y_train shape (37800,)
y_test shape (4200,)
In [20]: from sklearn.metrics import confusion_matrix
import tensorflow as tf
import itertools
from keras.utils.np_utils import to_categorical # convert to one-hot-encoding
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D
from tensorflow.keras.optimizers import Adam
from keras.preprocessing.image import ImageDataGenerator
from keras.callbacks import ReduceLROnPlateau
model = Sequential()
#
model.add(Conv2D(filters = 8, kernel_size = (5,5),padding = 'Same',
activation ='relu', input_shape = (28,28,1)))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Dropout(0.25))
#
model.add(Conv2D(filters = 16, kernel_size = (3,3),padding = 'Same',
activation ='relu'))
model.add(MaxPool2D(pool_size=(2,2), strides=(2,2)))



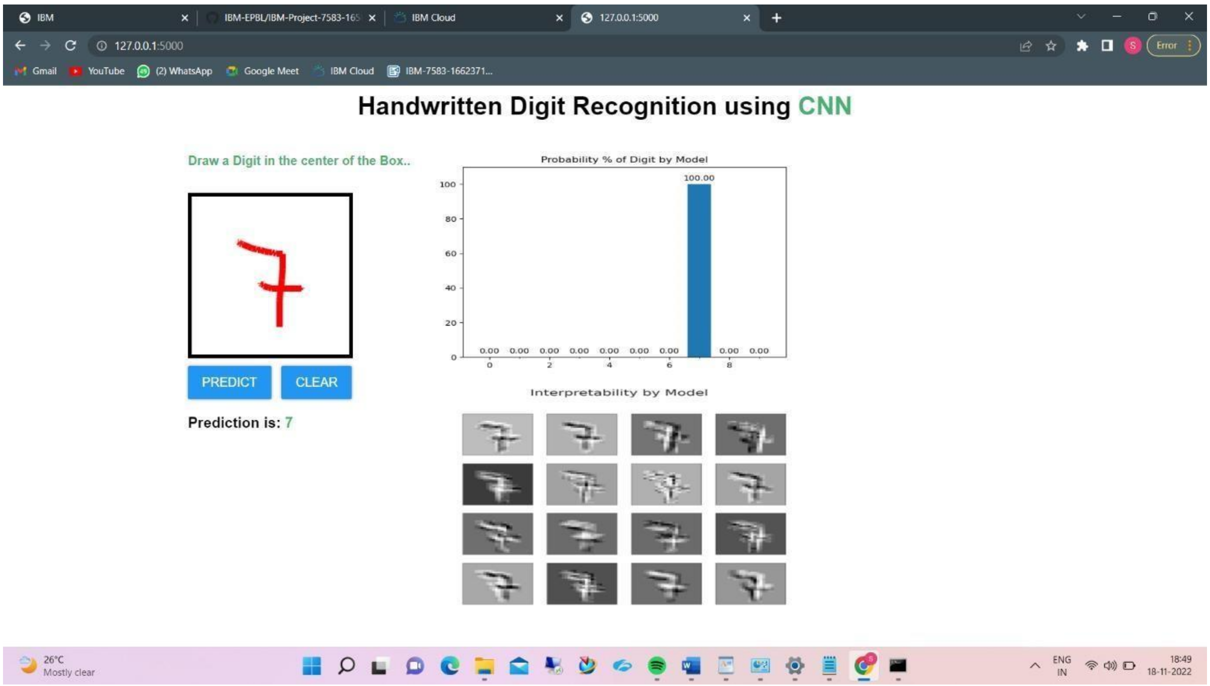
hand written Last Checkpoint: 11/11/2022 (unsaved changes)
In [24]: # data augmentation
datagen = ImageDataGenerator(
featurewise_center=False, # set input mean to 0 over the dataset
samplewise_center=False, # set each sample mean to 0
featurewise_std_normalization=False, # divide inputs by std of the dataset
samplewise_std_normalization=False, # divide each input by its std
zca_whitening=False, # dimesion reduction
rotation_range=5, # randomly rotate images in the range 5 degrees
zoom_range = 0.1, # Randomly zoom image 10%
width_shift_range=0.1, # randomly shift images horizontally 10%
height_shift_range=0.1, # randomly shift images vertically 10%
horizontal_flip=False, # randomly flip images
vertical_flip=False) # randomly flip images
datagen.fit(X_train)
In [25]: checkpoint_path = "training_1/cp.ckpt"
checkpoint_dir = os.path.dirname(checkpoint_path)
In [26]: cp_callback = tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_path,
save_weights_only=True,
verbose=1)
In [ ]: # Fit the model
history = model.fit_generator(datagen.flow(X_train,Y_train, batch_size=batch_size),
epochs = epochs, validation_data = (X_val,Y_val), steps_per_epoch=X_train.shape[0] // batch_size,ca
Epoch 1/10
151/151 [==============================] - ETA: 0s - loss: 1.0546 - accuracy: 0.6470
Epoch 00001: saving model to training_1\cp.ckpt
151/151 [==============================] - 32s 196ms/step - loss: 1.0546 - accuracy: 0.6470 - val_loss: 0.2043 - val_accuracy: 0.9452
Epoch 2/10

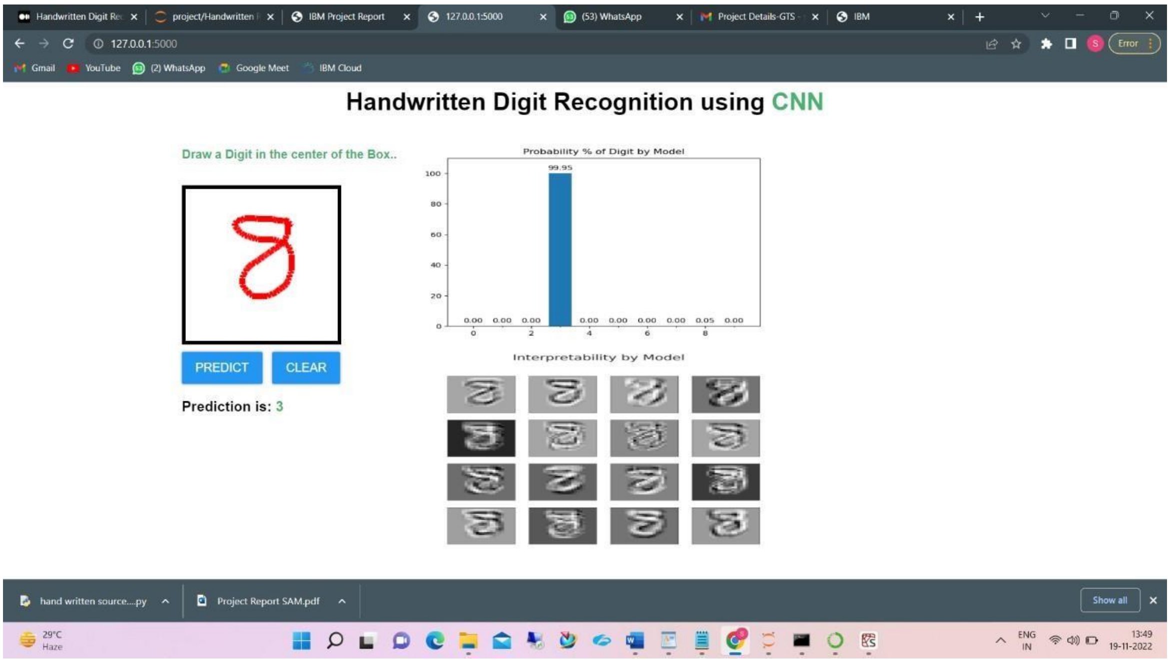

hand written Last Checkpoint: 11/11/2022 (unsaved changes)
vertical_flip=False) # randomly flip images
datagen.fit(X_train)
In [25]: checkpoint_path = "training_1/cp.ckpt"
checkpoint_dir = os.path.dirname(checkpoint_path)
In [26]: cp_callback = tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_path,
save_weights_only=True,
verbose=1)
In [ ]: # Fit the model
history = model.fit_generator(datagen.flow(X_train,Y_train, batch_size=batch_size),
epochs = epochs, validation_data = (X_val,Y_val), steps_per_epoch=X_train.shape[0] // batch_size,
Epoch 1/10
151/151 [==============================] - ETA: 0s - loss: 1.0546 - accuracy: 0.6470
Epoch 00001: saving model to training_1\cp.ckpt
151/151 [==============================] - 32s 196ms/step - loss: 1.0546 - accuracy: 0.6470 - val_loss: 0.2043 - val_accuracy: 0.9452
Epoch 2/10
151/151 [==============================] - ETA: 0s - loss: 0.4022 - accuracy: 0.8717
Epoch 00002: saving model to training_1\cp.ckpt
151/151 [==============================] - 21s 140ms/step - loss: 0.4022 - accuracy: 0.8717 - val_loss: 0.1178 - val_accuracy: 0.9688
Epoch 3/10
151/151 [==============================] - ETA: 0s - loss: 0.2977 - accuracy: 0.9065
Epoch 00003: saving model to training_1\cp.ckpt
151/151 [==============================] - 20s 133ms/step - loss: 0.2977 - accuracy: 0.9065 - val_loss: 0.0947 - val_accuracy: 0.9752
Epoch 4/10
122/151 [=======================>......] - ETA: 3s - loss: 0.2573 - accuracy: 0.9209

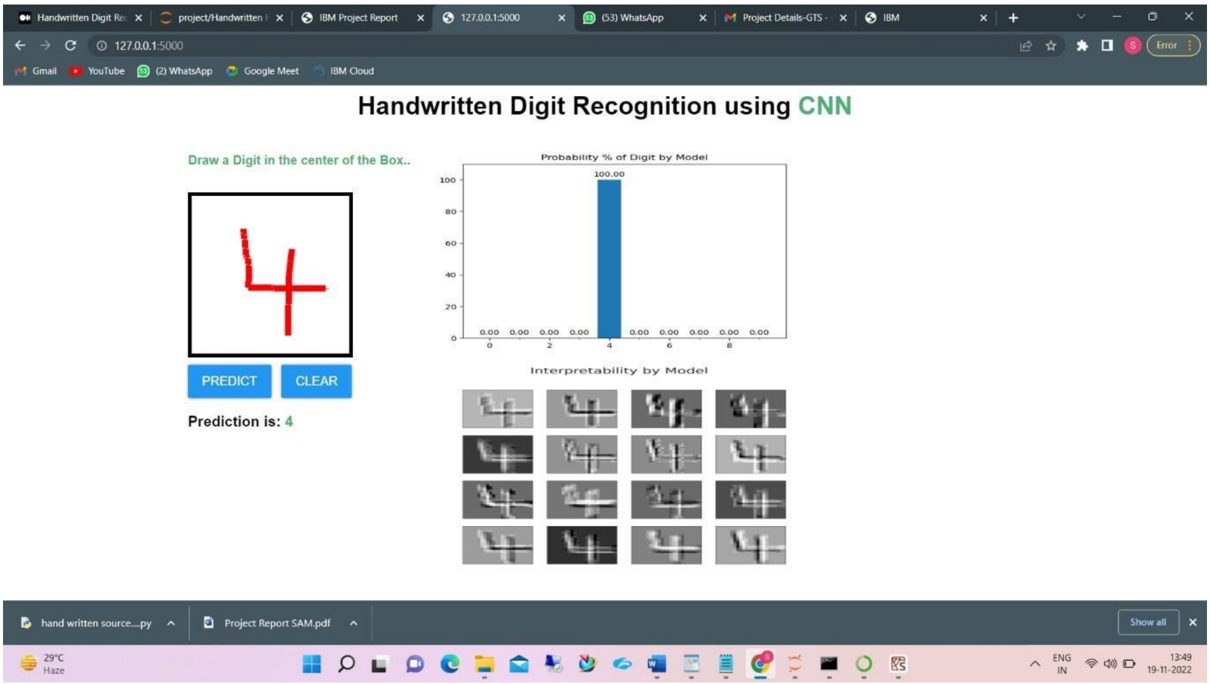

test_the_model Last Checkpoint: 16 hours ago (autosaved)
In [1]: import tensorflow as tf
from tensorflow import keras
import pandas as pd
import cv2
import numpy as np
from skimage.transform import resize
import matplotlib.pyplot as plt
In [2]: new_model = tf.keras.models.load_model('C:/Users/Durga/project/Handwritten Recognition/M2/Model/HDweights.h5')
In [3]: (x_train,y_train) , (x_test,y_test) = keras.datasets.mnist.load_data()
In [4]: x = x_test[35].reshape(1,28,28,1)
In [5]: y = new_model.predict(x)
In [6]: y.argmax()
Out[6]: 2
In [7]: plt.matshow(x_test[35])
Out[7]: <matplotlib.image.AxesImage at 0x14c0c05c250>



SOLUTION (OUTPUT) :




Handwritten Digit Recognition using CNN
Draw a Digit in the center of the Box..
PREDICT
CLEAR
Prediction is: 3
Probability % of Digit by Model
99.95
Interpretability by Model


Handwritten Digit Recognition using CNN
Draw a Digit in the center of the Box..
PREDICT
CLEAR
Prediction is: 4
Probability % of Digit by Model
100.00
Interpretability by Model

CHAPTER 8

8.TESTING

8.1 TEST CASES

Test case ID	Feature Type	Component	Test Scenario	Expected Result	Actual Result	Status
HP_TC_001	UI	Home Page	Verify UI elements in the Home Page	The Home page must be displayed properly	Working as expected	FAIL
HP_TC_002	UI	Home Page	Check if the UI elements are displayed properly in different screen sizes	The Home page must be displayed properly in all sizes	The UI is not displayed properly in screen size 2560 x 1801 and 768 x 630	FAIL
HP_TC_003	Functional	Home Page	Check if user can upload their file	The input image should be uploaded to the application successfully	Working as expected	PASS
HP_TC_004	Functional	Home Page	Check if user cannot upload unsupported files	The application should not allow user to select a non image file	User is able to upload any file	FAIL

HP_TC_005	Functional	Home Page	Check if the page redirects to the result page once the input is given	The page should redirect to the results page	Working as expected	PASS

M_TC_001	Functional	Model	Check if the model can handle various image sizes	The model should rescale the image and predict the results	Working as expected	PASS
M_TC_002	Functional	Model	Check if the model predicts the digit	The model should predict the number	Working as expected	PASS
M_TC_003	Functional	Model	Check if the model can handle complex input image	The model should predict the number in the complex image	The model fails to identify the digit since the model is not built to handle such data	FAIL
RP_TC_001	UI	Result Page	Verify UI elements in the Result Page	The Result page must be displayed properly	Working as expected	PASS

RP_TC_002	UI	Result Page	Check if the input image is displayed properly	The input image should be displayed properly	The size of the input image exceeds the display container	FAIL
RP_TC_003	UI	Result Page	Check if the result is displayed properly	The result should be displayed properly	Working as expected	PASS
RP_TC_004	UI	Result Page	Check if the other predictions are displayed properly	The other predictions should be displayed properly	Working as expected	PASS

8.2 USER ACCEPTANCE TESTING

User acceptance of the system is key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system and user at the time of developing and making changes whenever required. This is done in regarding to the following points.

- Input screen design.
- Output screen design.

CHAPTER 9

9.RESULTS

9.1 PERFORMANCE METRICS

Locust Test Report

During: 11/15/2022, 9:50:40 AM - 11/15/2022, 10:01:59 AM

Target Host: http://127.0.0.1:5000/

Script: locust.py

Request Statistics

Method	Name	# Requests	# Fails	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	RPS	Failures/s
GET	//	1043	0	13	4	290	1079	1.9	0.0
GET	//predict	1005	0	39648	385	59814	2670	1.8	0.0
	Aggregated	**2048**	**0**	**19462**	**4**	**59814**	**1859**	**3.7**	**0.0**

Response Time Statistics

Method	Name	50%ile (ms)	60%ile (ms)	70%ile (ms)	80%ile (ms)	90%ile (ms)	95%ile (ms)	99%ile (ms)	100%ile (ms)
GET	//	10	11	13	15	19	22	62	290
GET	//predict	44000	46000	47000	48000	50000	52000	55000	60000
	Aggregated	**36**	**36000**	**43000**	**45000**	**48000**	**50000**	**54000**	**60000**


Charts
Total Requests per Second
RPS
Failures/s
6
5
4
3
2
1
0
9:50:40 AM
9:51:33 AM
9:52:38 AM
9:53:27 AM
9:54:22 AM
9:55:18 AM
9:58:27 AM
9:59:22 AM
10:00:18 AM
10:01:05 AM
Response Times (ms)
Median Response Time
95% percentile
60,000
50,000
40,000
30,000
20,000
10,000
0
9:50:40 AM
9:51:33 AM
9:52:38 AM
9:53:27 AM
9:54:22 AM
9:55:18 AM
9:58:27 AM
9:59:22 AM
10:00:18 AM
10:01:05 AM
Number of Users
Users
120
100
80
60
40
20
0
9:50:40 AM
9:51:33 AM
9:52:38 AM
9:53:27 AM
9:54:22 AM
9:55:18 AM
9:58:27 AM
9:59:22 AM
10:00:18 AM
10:01:05 AM

CHAPTER10

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.
- Neural Network is used to train and identify written digits for greater efficiency.
- The accuracy rate is very high.
- Speed of data entry.
- It is much easier to dictate the machine than to write.
- Easier data retrieval.

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.
- There is a wide range of handwriting – good and bad.
- It is tricky for programmers to provide enough examples of how every character might look.
- Customers must try with clear image and neat handwriting to get accuracy in digits.
- Unclear image will not give accurate results.

CHAPTER 11

11.CONCLUSION

Convolutional Neural Network (CNN) adds its significant improvement to the Manuscript Document Recognition System. This paper tells us the effectiveness of CNN-based classification of data and pre-processing methods. Our model clearly sees handwriting and achieves outgoing predictions of up to 82.16% and accurate predictions of up to 69.16%. However the model can be continuously developed using multiple training samples. This will help the model to learn as well as the generalize better. There are many images in the training set that are completely invisible to the human eye.

This project demonstrated a web application that uses machine learning to recognize 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.

Through extensive evaluation using a MNIST dataset, the present work suggests the role of various hyper-parameters. Fine tuning of hyper-parameters is essential in improving the performance of CNN architecture. We achieved a recognition rate of 99.89% with the Adam optimizer for the MNIST database, which is better than all previously reported results. The effect of increasing the number of convolutional layers in CNN architecture on the performance of handwritten digit recognition is clearly presented through the experiments.

CHAPTER 12

12.FUTURE SCOPE

This project can be enhanced with a great field of machine learning and artificial intelligence. The world can think of a software which can recognize the text from a picture and can show it to the others, for example a shop name detector. Or this project can be extended to a greater concept of all the character sets in the world. This project has not gone for the total English alphabet because there will be more and many more training sets and testing values that the neural network model will not be enough to detect. Think of a AI modeled car sensor going with a direction modeling in the roadside, user shall give only the destination.

All of these enhancement is an application of the texture analysis where advanced image processing, Neural network model for training and advanced AI concepts will come. These applications can be modeled further .As this project is fully done by free and available resources and packages this can be also a limitation of the project. The fund is very important because all machine learning libraries and advanced packages are not available for free. Unless of those the most of the visualizing platforms like on which developers are doing some works like Watson Studio or Aws. These all are mainly paid platforms where a lot of ML projects are going on.

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.

CHAPTER 13

13. APPENDIX (SOURCE CODE)

```
import numpy as np import pandas as pd import matplotlib.pyplot
as plt import seaborn as sns import warnings
warnings.filterwarnings('ignore')import os
print(os.listdir('C:/Users/gts/handwritten project/M2/')) data =
pd.read_csv("C:/Users/gts/handwritten project/M2/train.csv") data.shape
data.isnull().sum().sum() data.info() data.head()
X = data.drop("label",axis=1) Y
= data["label"]
plt.figure(figsize=(20,10))
sns.countplot(Y,palette="icefire")
plt.title("Number of digits") plt.show()
Y.value_counts() img = X.iloc[0].to_numpy()
img = img.reshape((28,28))
plt.imshow(img,cmap='gra
y') plt.title(X.iloc[0,0])
plt.axis("off") plt.show() img
= X.iloc[3].to_numpy() img
= img.reshape((28,28))
plt.imshow(img,cmap='gra
y') plt.title(X.iloc[3,0])
plt.axis("off") plt.show()


# Normalize the data X = X
/ 255.0 print("x shape:
",X.shape)
```

```
X = X.to_numpy()

# Reshape
X = X.reshape(-1,28,28,1) print("x shape:
",X.shape)
from keras.utils.np_utils import to_categorical

 # convert to one-hot-encoding
Y_train = to_categorical(Y, num_classes = 10)

# Split the train and the validation set for the fitting from
sklearn.model_selection import train_test_split
X_train, X_val, Y_train, Y_val = train_test_split(X, Y, test_size = 0.1,
random_state=2)
print("x_train shape",X_train.shape) print("x_test
shape",X_val.shape) print("y_train shape",Y_train.shape)
print("y_test shape",Y_val.shape) from sklearn.metrics import
confusion_matrix import tensorflow as tf import itertools from
keras.utils.np_utils import to_categorical # convert to
onehotencoding from keras.models import Sequential  from
keras.layers import Dense, Dropout, Flatten, Conv2D,
MaxPool2D from tensorflow.keras.optimizers import Adam from
keras.preprocessing.image import ImageDataGenerator from
keras.callbacks import ReduceLROnPlateau model = Sequential()
model.add(Conv2D(filters = 8, kernel_size = (5,5),padding =
'Same',activation ='relu', input_shape = (28,28,1)))
model.add(MaxPool2D(pool_size=(2,2)))
```

```
model.add(Dropout(0.25)) model.add(Conv2D(filters = 16,
kernel_size = (3,3),padding = 'Same', activation ='relu'))
model.add(MaxPool2D(pool_size=(2,2), strides=(2,2)))
model.add(Dropout(0.25))

# fully connected model.add(Flatten())
model.add(Dense(256, activation = "relu"))
model.add(Dropout(0.5))
model.add(Dense(10, activation = "softmax"))
optimizer = Adam(lr=0.001, beta_1=0.9,
beta_2=0.999)

model.compile(optimizer = optimizer , loss =
"sparse_categorical_crossentropy", metrics=["accuracy"]) epochs
= 10   batch_size = 250

  ImageDataGenerator(         featurewise_center=False,  # set input mean
to 0 over the dataset         samplewise_center=False,  # set each sample
mean to 0         featurewise_std_normalization=False,  # divide inputs by
std of the dataset         samplewise_std_normalization=False,  # divide
each input by its std         zca_whitening=False,  # dimesion reduction
rotation_range=5,  # randomly rotate images in the range 5 degrees
        zoom_range = 0.1, # Randomly zoom image 10%
width_shift_range=0.1,  # randomly shift images horizontally 10%
height_shift_range=0.1,  # randomly shift images vertically 10%
horizontal_flip=False,  # randomly flip images         vertical_flip=False)  #
randomly flip images datagen.fit(X_train) checkpoint_path =
"training_1/cp.ckpt" checkpoint_dir = os.path.dirname(checkpoint_path)
```

```
cp_callback =
tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_path,
save_weights_only=True, verbose=1)

# Fit the model history =
model.fit_generator(datagen.flow(X_train,Y_train, batch_size=batch_size),
epochs = epochs, validation_data = (X_val,Y_val),
steps_per_epoch=X_train.shape[0] //
batch_size,callbacks=[cp_callback]) model_json =
model.to_json() with open(r"./Model/lung_model.json", "w")
as json_file:
    json_file.write(model_json) model.save("./Model/lung_weights.h5")

#html code
<html>
<script type="text/javascript" src="{{url_for('static', filename='jquery.min.js')
}}"></script>
<link rel="stylesheet" type="text/css" href="{{url_for('static',
filename='style.css') }}"> <script type="text/javascript">     var
canvas, ctx, flag = false,         prevX = 0,        currX = 0,
prevY = 0,        currY = 0,        dot_flag = false;
 var  x  =  "red",  y  =  8;    function  init()  {      canvas  =
document.getElementById('can');
document.getElementById("probs").style.display  =  "none";
document.getElementById("interpret").style.display = "none";
ctx  =  canvas.getContext("2d");    w  =  canvas.width;     h  =
canvas.height;          canvas.addEventListener("mousemove",
function (e) {
```

```
            findxy('move', e)
        }, false);
 canvas.addEventListener("mousedown", function (e) {
            findxy('down', e)
        }, false);
 canvas.addEventListener("mouseup", function (e) {
            findxy('up', e)
        }, false);
 canvas.addEventListener("mouseout", function (e) {
findxy('out', e)
        }, false);
    }    function
draw() {
ctx.beginPath();
ctx.moveTo(prevX,   prevY);
ctx.lineTo(currX,       currY);
ctx.strokeStyle        =        x;
ctx.lineWidth         =         y;
ctx.stroke();  ctx.closePath();
    }
function erase() {         ctx.clearRect(0, 0, w, h);
document.getElementById("canvasimg").style.display = "none";
document.getElementById("prediction").style.display = "none";
document.getElementById("probs").style.display = "none";
document.getElementById("interpret").style.display = "none";
  b = document.getElementsByTagName("body")[0];
b.querySelectorAll('a').forEach(n => n.remove());
```

```
    }   function
save() {
document.getElementById("prediction").style.display = "block";
document.getElementById("probs").style.display = "block";
document.getElementById("interpret").style.display = "block";
var final_image = canvas.toDataURL();        var a =
document.createElement('a');        a.href = final_image;
a.download = 'process.png';        document.body.appendChild(a);
       // a.click();        $.ajax({
url: "{{ url_for('process') }}",
type: 'POST',
 data: final_image,
           success: function (response) {                endresult =
JSON.parse(JSON.stringify(response))  console.log(endresult
 $('#prediction').html('Prediction is: <span id="text">'   endresult.data
+ '</span>')
              $('#probs').prop('src', 'data:image/png;base64,' +
endresult.probencoded)
              $('#interpret').prop('src', 'data:image/png;base64,' +
endresult.interpretencoded)
           }
       });     }     function findxy(res, e) {
if (res == 'down') {            prevX = currX;
prevY = currY;           currX = e.clientX -
canvas.offsetLeft;           currY = e.clientY -
canvas.offsetTop;           flag = true;
dot_flag = true;           if (dot_flag) {
ctx.beginPath();               ctx.fillStyle = x;
```

```
ctx.fillRect(currX, currY, 2, 2);
ctx.closePath();                dot_flag = false;
}
        }          if (res == 'up' || res ==
"out") {
flag = false;
        }         if (res == 'move') {            if (flag) {
prevX = currX;                prevY = currY;
currX = e.clientX - canvas.offsetLeft;
currY = e.clientY - canvas.offsetTop;
draw();
            }
        }
    }
</script>
<body onload="init()">
    <center>
        <h1> Handwritten Digit Recognition using <span
id="text">CNN</span></h1>
    </center>
    <div id="side">
        <h4 id='text'> Draw a Digit in the center of the Box.. </h4>
        <canvas id="can" width="200px" height="200px"></canvas>
        <img id="canvasimg">
        <div style="margin-top: 10;">
            <button class="ripple" id="btn" onclick="save()"> predict  </button>
             
```

```
            <button id="clr" onclick="erase()"> clear </button>
            <h3 id="prediction"></h3>
        </div>
    </div>
    <div>
        <img id="probs" src="" alt="" height="45%" width="35%">
        <img id="interpret" src="" alt="" height="45%" width="35%">
    </div>
</body>
</html>
```

```
import torch import base64
import config import
matplotlib import numpy as np
from PIL import Image from
io import BytesIO from train
import MnistModel import
matplotlib.pyplot as plt
from flask import Flask, request, render_template, jsonify matplotlib.use('Agg')
MODEL = None
DEVICE = torch.device('cuda:0' if torch.cuda.is_available() else 'cpu')
app = Flask(__name__) class SaveOutput:     def __init__(self):
self.outputs = []      def __call__(self, module, module_in,
module_out):
        self.outputs.append(module_out) def
clear(self):
        self.outputs = [] def register_hook():     save_output =
```

```
SaveOutput()     hook_handles = []     for layer in MODEL.modules():
if isinstance(layer, torch.nn.modules.conv.Conv2d):             handle =
layer.register_forward_hook(save_output)
hook_handles.append(handle)     return save_output def
module_output_to_numpy(tensor):     return
tensor.detach().to('cpu').numpy() def autolabel(rects, ax):
    """Attach a text label above each bar in *rects*, displaying its height."""
for rect in rects:
        height = rect.get_height()
ax.annotate('{0:.2f}'.format(height),
xy=(rect.get_x() + rect.get_width() / 2, height),
xytext=(0, 3),  # 3 points vertical offset
textcoords="offset points",                     ha='center', va='bottom')
def prob_img(probs):     fig, ax = plt.subplots()     rects =
ax.bar(range(len(probs)), probs)
ax.set_xticks(range(len(probs)), (0, 1, 2, 3, 4, 5, 6, 7, 8, 9))
ax.set_ylim(0, 110)     ax.set_title('Probability % of Digit by
Model')     autolabel(rects, ax)     probimg = BytesIO()
fig.savefig(probimg, format='png')     probencoded =
base64.b64encode(probimg.getvalue()).decode('utf-
8')     return probencoded def interpretability_img(save_output):
    images = module_output_to_numpy(save_output.outputs[0])
with plt.style.context("seaborn-white"):         fig, _ =
plt.subplots(figsize=(20, 20))         plt.suptitle("Interpretability by
Model", fontsize=50)         for idx in range(16):
            plt.subplot(4, 4, idx+1)
plt.imshow(images[0,                                        idx])
plt.setp(plt.gcf().get_axes(),    xticks=[],    yticks=[])
```

```
interpretimg = BytesIO()     fig.savefig(interpretimg,
format='png')                          interpretencoded     =
base64.b64encode(
interpretimg.getvalue()).decode('utf-8')           return
interpretencoded       def       mnist_prediction(img):
save_output    =    register_hook()                img    =
img.to(DEVICE,  dtype=torch.float)         outputs  =
MODEL(x=img)                                   probs       =
torch.exp(outputs.data)[0] * 100       probencoded =
prob_img(probs)                       interpretencoded     =
interpretability_img(save_output)

    _, output = torch.max(outputs.data, 1)     pred =
module_output_to_numpy(output)     return pred[0], probencoded,
interpretencoded
@app.route("/process", methods=["GET", "POST"]) def
process():
    data_url = str(request.get_data())     offset = data_url.index(',')+1
img_bytes = base64.b64decode(data_url[offset:])     img =
Image.open(BytesIO(img_bytes))     img = img.convert('L')
img = img.resize((28, 28))     # img.save(r'templates\image.png')
img = np.array(img)     img = img.reshape((1, 28, 28))     img =
torch.tensor(img, dtype=torch.float).unsqueeze(0)     data,
probencoded, interpretencoded = mnist_prediction(img)
response = {
        'data': str(data),
        'probencoded': str(probencoded),
        'interpretencoded': str(interpretencoded),
```

```
    }
    return jsonify(response)
@app.route("/", methods=["GET", "POST"]) def start():
    return render_template("default.html")
if __name__ == "__main__":     MODEL
= MnistModel(classes=10)
    MODEL.load_state_dict(torch.load(
        'checkpoint/mnist.pt', map_location=DEVICE))      MODEL.to(DEVICE)
MODEL.eval()     app.run(host=config.HOST, port=config.PORT,
debug=config.DEBUG_MODE)
```

GitHub link : **https://github.com/IBM-EPBL/IBM-Project-75831658892843**

Project demo link : **https://drive.google.com/file/d/1W6ACJtCf175FrnjUh82GQ2afFFhiP_G1/view?usp=share_link**