IBM NALAIYATHIRAN

PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM

SUBMITTED BY:

TEAM ID: PNT2022TMID54113

TEAM LEAD:

1)SOWMIYA A (111919106058)

TEAM MEMBERS:

1)NIVETHA S (111919106045)

2)KOWSHIKA T (111919106032)

3)RAJESWARI C (111919106050)

TABLE OF CONTENTS

1.INTRODUCTION

1.1 Project Overview

Full Text Available Handwritten digit recognition plays a significant role in many user authentication applications in the modern world. As the handwritten digits are not of the same size, thickness, style and orientation. Therefore these challenges are to be faced to resolve this problem in my project. The objective of this project is to build a Graphical User Interface (GUI) in which we can draw the digit and recognize it straight away. I will be using a special type of deep neural network that is Convolutional Neural Network which is applied in analyzing visual imagery where large set of pixel data in images are converted to conserve useful data of images which can be fed as input layer data to Artificial Neural Network for training purpose. After that system will use hidden layers of CNN to develop a model for handwritten digit recognition. Here we will apply a LeNet-5 Convolution Neural Network algorithm on Modified National Institute of Standards and Technology (MNIST) dataset which includes handwritten digits total of 70,000 images. Keras, a Neural Network library written in python will be used. Stochastic gradient and backpropagation algorithm are used for training the network and the forward algorithm is used for testing. Once the model is ready, user can input their image which consist of digit on our GUI and they will get correct prediction of their input.

1.2 Purpose

This project aims to meet the following objectives:

i. To develop handwritten digit recognizing system that enables users to automate the process of digit recognition using this deep learning model.

ii. To test the accuracy of the model 10 .

iii. Efficient model which is less computation intensive.

2.LITERATURE SURVEY

2.1 Existing problem

Handwritten digit recognition finds its application in various fields such as post mail sorting system where scanned images of mail envelopes are made into queue and extract the section describing postcode to be delivered. With the help of digit recognizer, sorting of mails can be done based on these postcodes according to their region. Another application that utilizes this technique is form processing, digits are extracted from certain columns of a form and users put certain filters to get the desired results they want. But there is no interface for a user to get their images scanned and recognized which makes the task complicated to use for a normal user.

2.2 References

1. https://www.ijnrd.org/papers/IJNRD1704024.pdf -PRIYA, RAJENDRA SINGH.

2. https://www.irjet.net/archives/V9/i6/IRJET-V9I6208.pdf -Dhruv Sharma, Ishaan Singh.

3.https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.228.158&rep=rep1&type=pdf -MALOTHU NAGU, N VIJAY SHANKAR, K. ANNAPURNA .

4. http://ijcsit.com/docs/Volume%207/vol7issue1/ijcsit2016070101.pdf -Ayush Purohit, Shardul Singh Chauhan.

5. http://troindia.in/journal/ijcesr/vol6iss6part2/32-36.pdf -Rohini.M , Dr.D.Surendran

2.3 Problem Statement Definition

1. 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.

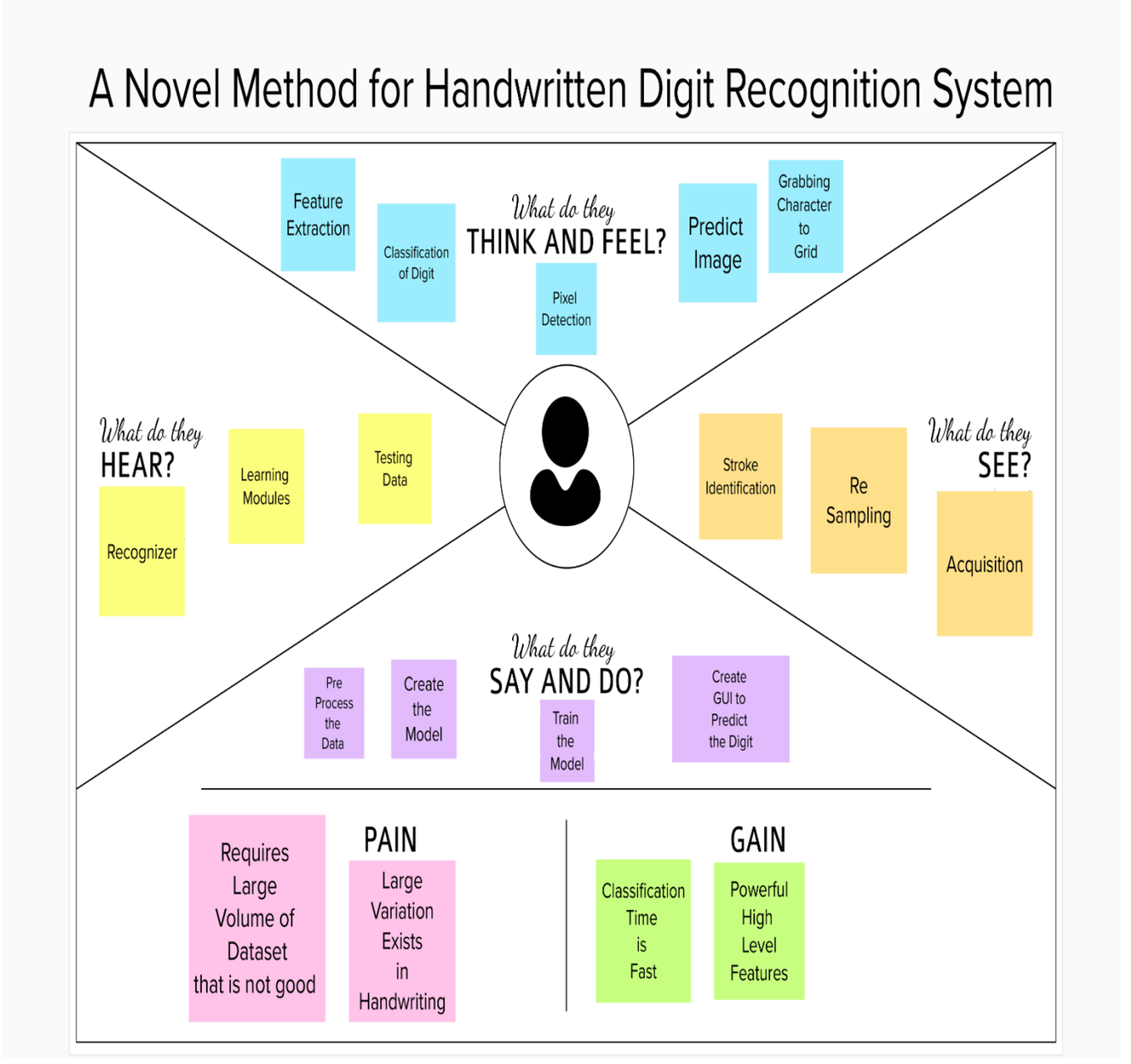
2. The similarity between digits such as 1 and 7, 5 and 6, 3 and 8, 2 and 7 etc. So, classifying between these numbers is also a major problem for computers.

3. The uniqueness and variety in the handwriting of different individuals also influence the formation and appearance of the digits, As the handwritten digits are not of the same size, thickness, style and orientation. Therefore these challenges are to be faced to resolve this problem.

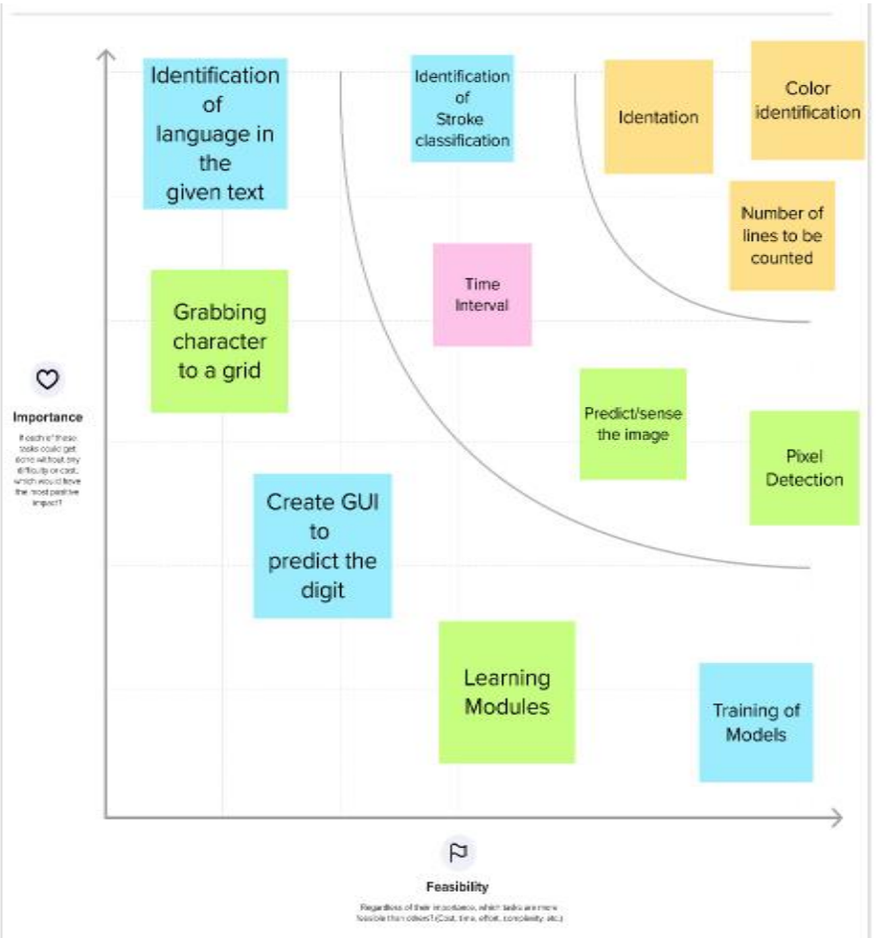
4. There are many types of handwriting ,it is hard to identify , in several application it is more time consuming because The shape of the digits are little bit different, The digits are not written properly.

3. DEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION AND BRAINSTROMING



3.3 PROPOSED SOLUTION

1. Problem Statement (Problem to be solved) - Computer programmes' ability to detect human-written numbers is known as handwritten digit recognition. Because handwritten figures are not always accurate and can take many various forms and sizes, it is a difficult work for the machine.

2. Novelty / Uniqueness - Recognize the digits precisely rather than all the characters like OCR.

3. Idea / Solution description - Using data from various sources, including images, documents, and touch defenses, a computer is able to celebrate the mortal handwritten numbers. It permits users to convert all of their handwritten notes and signatures into text documents in electronic form, using

much less physical space than would be needed to store the physical copies of those documents.

4. Social Impact / Customer Satisfaction - The Handwritten Digit Recognizer software was made using artificial intelligence. It approximates the printed word digitally by identifying letters using sophisticated algorithms before producing a digital approximation.

5. Business Model (Revenue Model) - For efficient traffic control, this technology can be connected with traffic surveillance cameras to read license plates. Pin-code details can be easily identified and recognized by integrating with the postal system.

6. Scalability of the Solution - The capacity to recognize numbers in more distracting circumstances. The maximum number of digits that can be recognized is unlimited.

3.4 PROBLEM SOLUTION FIT

1.CUSTOMER SEGMENTS

- Fintech Industries.
- Supply Chain Management.
- Medical data Transcriptions.
- Scientific and Space Research.

2. CUSTOMER CONSTRAINTS

- Speed and Accuracy of the system.
- Size of the vocabulary.
- Spatial layout.
- Lack of feedback-based system.

3. AVAILABLE SOLUTIONS

- Free OCR API.
- Human centric data feed.

4. JOBS-TO-BE-DONE / PROBLEMS

- To design a system that recognizes a wide range of handwriting scripts.
- ML based approach to identify the character quickly and accurately.
- Adaptive learning module to learn from its own instances and get updated.

5. PROBLEM ROOT CAUSE

- In cases where distinct characters look very similar making it hard for a computer to recognize it accurately.
- Different styles of cursive handwriting is another challenge that requires a support system on vocabulary.

4. REQUIREMENT ANALYSIS

4.1 Functional Requirements

Functional Requirement and description:

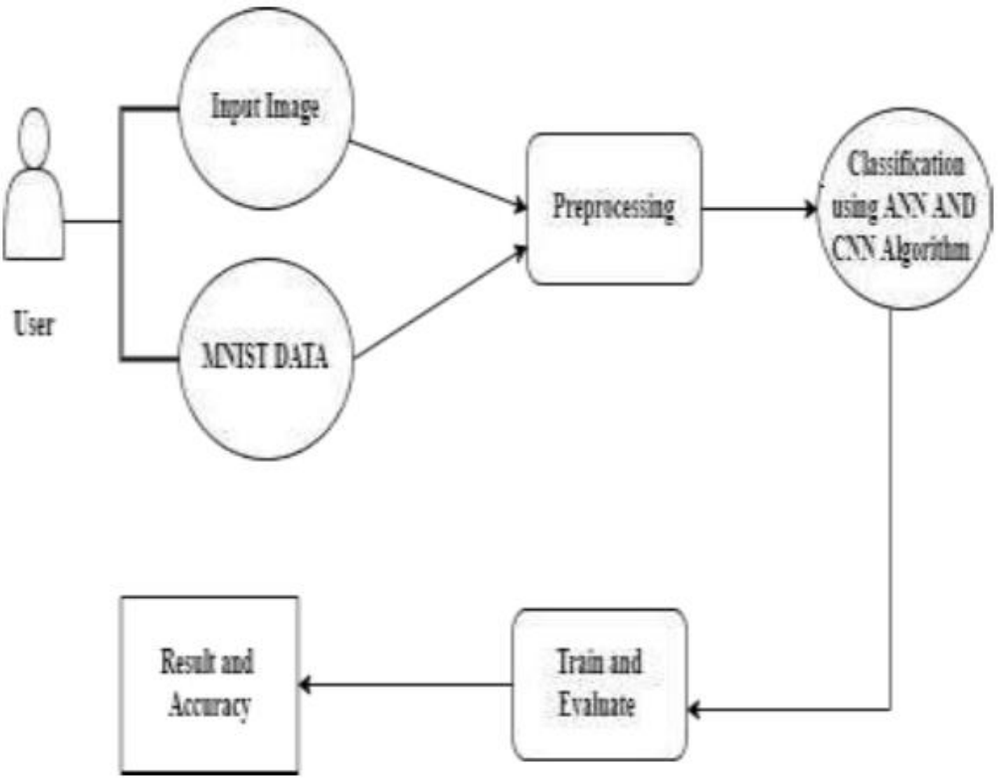
FR-1	**Image Data**: Handwritten digit recognition is the ability of a computer to recognize the human handwritten digits from different sources like images, papers, touch screens, etc., and classify them into 10 predefined classes (0-9). this has been a topic of boundless-research in the field of deep learning.
FR-2	**Website:** Web hosting makes the files that comprise a website (code, images, etc.) available for viewing online. Every website you've ever visited is hosted on a server. The amount of space allocated on a server to a website depends on the type of hosting. the main types of hosting are shared, dedicated, VPS..
FR-3	**Digit_Classifieí_Model:** Use the MNISl' database of handwritten digits to train a convolutional network to predict the digit given an image. First obtain the training and validation data.
FR-4	**MNISl' dataset:** the MNIST dataset is an acronym that stands for the Modified National Institute of Standards and Technology dataset. It is a dataset of 60,000 small square 28×28 pixel grayscale images of handwritten single digits between 0 and 9.
FR-5	**Cloud:** The cloud provides a number of IT services such as servers, databases, software, virtual storage, and networking, among others. In layman's terms, Cloud Computing is defined as a virtual platform that allows you to store and access your data over the internet without any limitations.

4.2 NON-FUNCTIONAL REQUIREMENTS

NÏR No.	Non-Functional Requirement
NFR-1	**Usability:** Handwritten character recognition is one of the practically important issues in pattern recognition applications. the applications of digit recognition include in postal mail sorting, bank check processing, form data entry, etc.
NFR-2	**Reliability:** 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.
NFR-3	**Performance:** the neural network uses the examples to automatically infer rules for recognizing handwritten digits. Furthermore, by increasing the number of training examples, the network can learn more about handwriting, and so improve its accuracy. there are a number of ways and algorithms to recognize handwritten digits, including Deep Learning/CNN, SVM, Gaussian Naive Bayes, KNN, Decision lees, Random Joists, etc.
NFR-4	**Accuracy:** Optical Character Recognition (OCR) technology provides higher than 99% accuracy with typed characters in high- quality images. However, the diversity in human writing types, spacing differences, and inequalities of handwriting causes less accurate character recognition.

5. PROJECT DESIGN

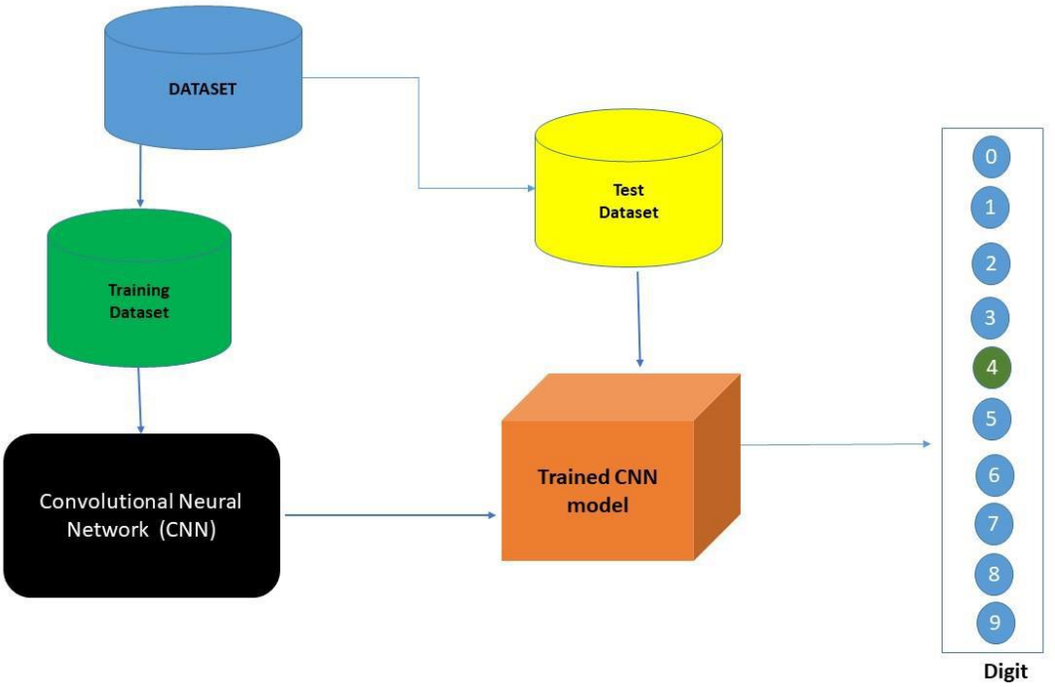
5.1 DATA FLOW DIAGRAM



5.2 SOLUTION & TECHNICAL ARCHITECTURE

Technical Architecture:

The architectural diagram of the model is as below and the Technology used is shown in the below table.



S.No	Components	Description	Technology
1.	User Interface	How user interacts with application e.g., Mobile Application	HTML, CSS, JavaScript / Angular JS / Node Red.
2.	Application Logic-1	Logic for a process in the application	Java / Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on AI	IBM DB2.
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1	Purpose of External API used in the application	IBM Weather API, etc.

5.3 USER STORIES

USER 1 - As a user, I can view the guide and awareness to use this application.

I can view the awareness to use this application and its limitations.

USER 2 - As a user, In this prediction page I get to choose the image.

I can choose the image from our local system and predict the output.

USER 3 - As a user, I'm Allowed to upload and choose the image to be uploaded

I can upload and choose the image from the system storage and also in any virtual storage.

USER 4 - As a user, I will train and test the input to get the maximum accuracy of output.

I can able to train and test the application until it gets maximum accuracy of the result.

USER 5 - As it is a web application, it is installation free

I can use it without the installation of the application or any software.

USER 6 - As a user, I'm allowed to view the guided video to use the interface of this application.

I can gain knowledge to use this application by a practical method.

6 . PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	Data Collection	USN-1	As a user, I can collect the dataset from various resources with different handwritings.	10	Low
Sprint-1	Data Preprocessing	USN-2	As a user, I can load the dataset, handling the missing data, scaling and split data into train and test.	10	Medium
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	High
Sprint-2	Add CNN layers	USN-4	Creating the model and adding the input, hidden, and output layers to it.	5	High
Sprint-2	Compiling the model	USN-5	With both the training data defined and model defined, it's time to configure the learning process.	2	Medium

6.2 SPRINT DELIVERY SCHEDULE

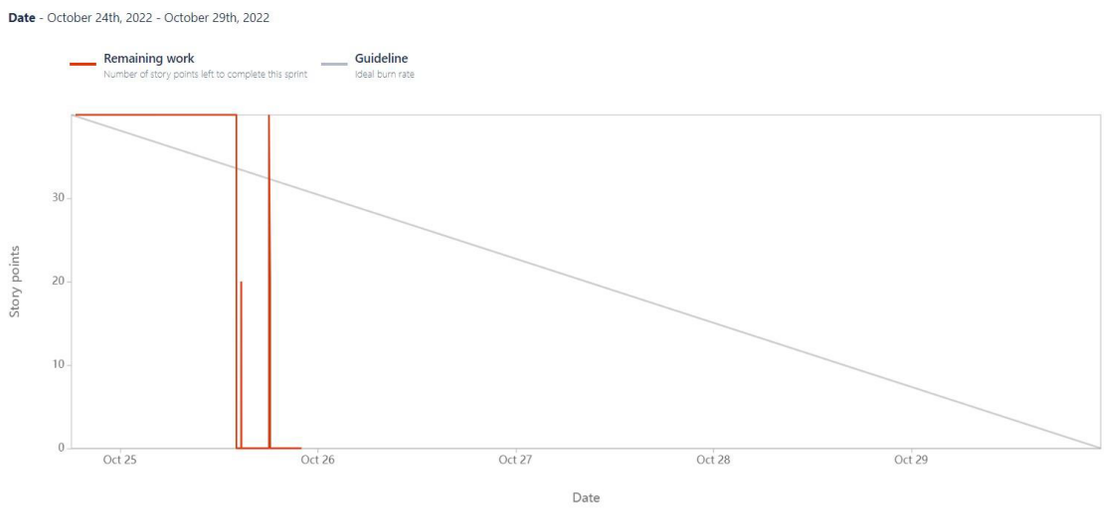
Sprint-2	Train & test the model	USN-6	As a user, let us train our model with our image dataset.	6	Medium
Sprint-2	Save the model	USN-7	As a user, the model is saved & integrated with an android application or web application in order to predict something.	2	Low
Sprint-3	Building UI Application	USN-8	As a user, I will upload the handwritten digit image to the application by clicking a upload button.	5	High
Sprint-3		USN-9	As a user, I can know the details of the fundamental usage of the application.	5	Low
Sprint-3		USN-10	As a user, I can see the predicted / recognized digits in the application.	5	Medium
Sprint-4	Train the model on IBM	USN-11	As a user, I train the model on IBM and integrate flask/Django with scoring end point.	10	High
Sprint-4	Cloud Deployment	USN-12	As a user, I can access the web application and make the use of the product from anywhere.	10	High

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

6.3 REPORTS FROM JIRA

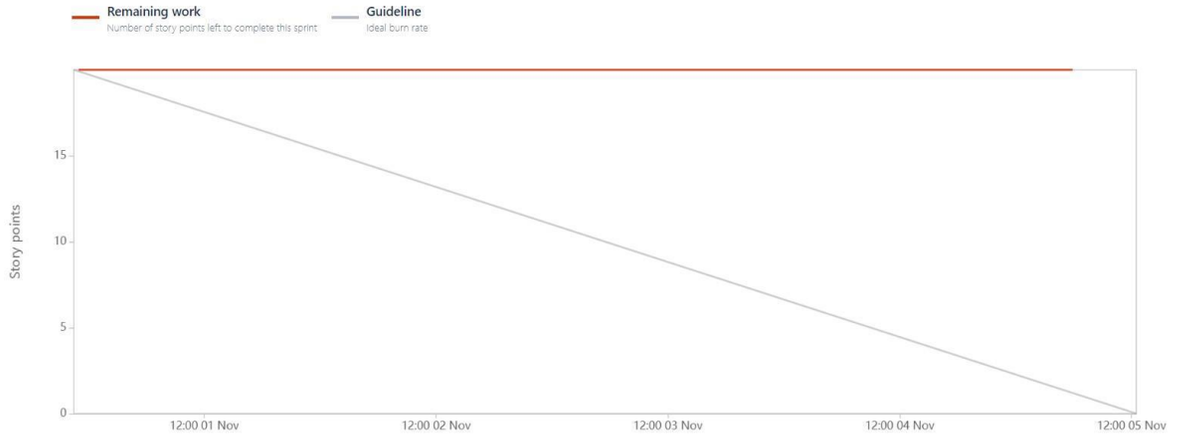
Velocity Report

SPRINT 1



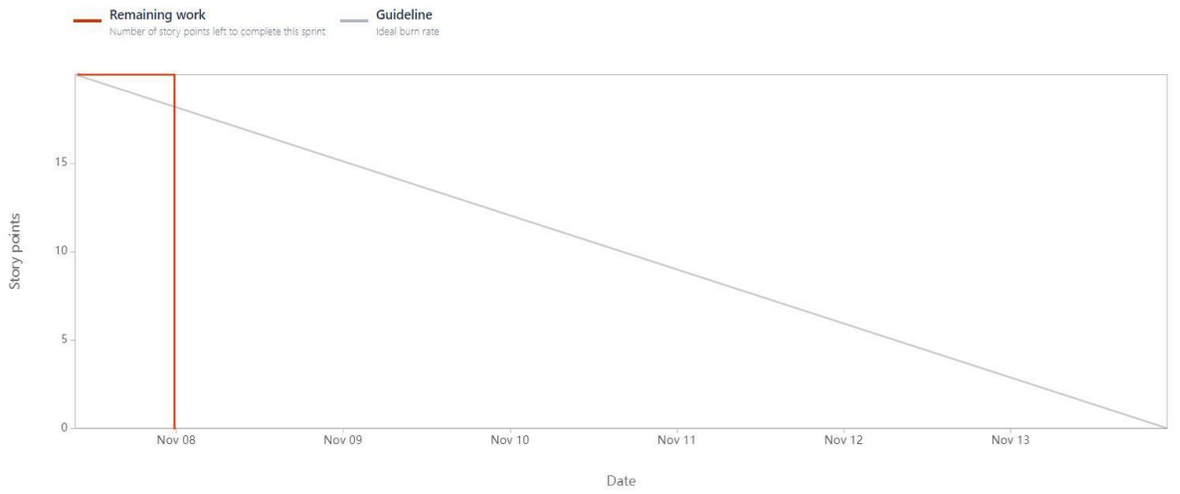
SPRINT 2

Date - October 31st, 2022 - November 5th, 2022



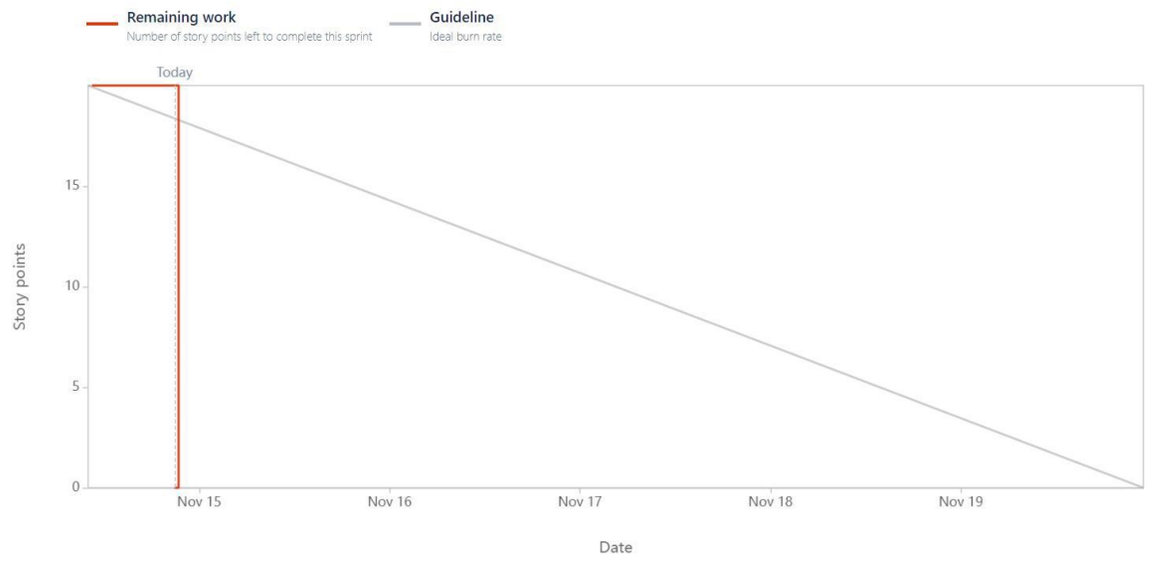
SPRINT 3

Date - November 7th, 2022 - November 13th, 2022



SPRINT 4

Date - November 14th, 2022 - November 19th, 2022



7. CODING & SOLUTIONING

```
pwd
import cv2
import numpy as np
from keras.datasets import mnist
from keras.layers import Dense, Flatten, MaxPooling2D, Dropout
from keras.layers.convolutional import Conv2D
from keras.models import Sequential
from tensorflow.keras.utils import to_categorical
import matplotlib.pyplot as plt
(X_train, y_train), (X_test, y_test) = mnist.load_data()
## Looking at a sample
plt.imshow(X_train[0], cmap="gray")
plt.show()
print (y_train[0])
## Checking out the shapes involved in dataset
print ("Shape of X_train: {}".format(X_train.shape))
print ("Shape of y_train: {}".format(y_train.shape))
print ("Shape of X_test: {}".format(X_test.shape))
print ("Shape of y_test: {}".format(y_test.shape))
# Reshaping so as to convert images for our model
X_train = X_train.reshape(60000, 28, 28, 1)
X_test = X_test.reshape(10000, 28, 28, 1)
print ("Shape of X_train: {}".format(X_train.shape))
```

```
print ("Shape of y_train: {}".format(y_train.shape))
print ("Shape of X_test: {}".format(X_test.shape))
print ("Shape of y_test: {}".format(y_test.shape))
### Lets one hot encode labels
y_train = to_categorical(y_train)
y_test = to_categorical(y_test)
## Declare the model
model = Sequential()

## Declare the layers
layer_1 = Conv2D(64, kernel_size=3, activation='relu',
input_shape=(28, 28, 1))
layer_2 = MaxPooling2D(pool_size=2)
layer_3 = Conv2D(32, kernel_size=3, activation='relu')
layer_4 = MaxPooling2D(pool_size=2)
layer_5 = Dropout(0.5)
layer_6 = Flatten()
layer_7 = Dense(128, activation="relu")
layer_8 = Dropout(0.5)
layer_9 = Dense(10, activation='softmax')

## Add the layers to the model
model.add(layer_1)
model.add(layer_2)
model.add(layer_3)
model.add(layer_4)
```

```
model.add(layer_5)

model.add(layer_6)

model.add(layer_7)

model.add(layer_8)

model.add(layer_9)

model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])

model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=3)

example = X_train[1]

prediction = model.predict(example.reshape(1, 28, 28, 1))

print ("Prediction (Softmax) from the neural network:\n\n
{}".format(prediction))

hard_maxed_prediction = np.zeros(prediction.shape)

hard_maxed_prediction[0][np.argmax(prediction)] = 1

print ("\n\nHard-maxed form of the prediction: \n\n
{}".format(hard_maxed_prediction))

print ("\n\n--------- Prediction --------- \n\n")

plt.imshow(example.reshape(28, 28), cmap="gray")

plt.show()

print("\n\nFinal Output: {}".format(np.argmax(prediction)))

image = cv2.imread('test_image.jpg')

image = np.full((100,80,3), 12, dtype = np.uint8)

grey = cv2.cvtColor(image.copy(), cv2.COLOR_BGR2GRAY)

ret, thresh = cv2.threshold(grey.copy(), 75, 255,
cv2.THRESH_BINARY_INV)

contours,hierarchy = cv2.findContours(thresh.copy(),
cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
```

```
preprocessed_digits = []

for c in contours:

    x,y,w,h = cv2.boundingRect(c)

    # Creating a rectangle around the digit in the original image (for displaying the digits fetched via contours)

    cv2.rectangle(image, (x,y), (x+w, y+h), color=(0, 255, 0), thickness=2)

    # Cropping out the digit from the image corresponding to the current contours in the for loop

    digit = thresh[y:y+h, x:x+w]

    # Resizing that digit to (18, 18)

    resized_digit = cv2.resize(digit, (18,18))

    # Padding the digit with 5 pixels of black color (zeros) in each side to finally produce the image of (28, 28)

    padded_digit = np.pad(resized_digit, ((5,5),(5,5)), "constant", constant_values=0)

    # Adding the preprocessed digit to the list of preprocessed digits

    preprocessed_digits.append(padded_digit)

print("\n\n\n----------------Contoured Image--------------------")

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.

client_34f2913f79dd4b1b9693d877bf1e3b72 = ibm_boto3.client(service_name='s3',

    ibm_api_key_id='LvCXgkvxRMaY4f0OG5GlXpBS-Y6uo7EtRqeewYYE2bWC',

    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')

print=("\n\n\n----------Contoured Image----------")

plt.imshow(image, cmap="gray")

plt.show()

inp = np.array(preprocessed_digits)

for digit in preprocessed_digits:

    prediction = model.predict(digit.reshape(1, 28, 28, 1))

    print ("\n\n-------------------------------------\n\n")

    print ("=========PREDICTION========== \n\n")

    plt.imshow(digit.reshape(28, 28), cmap="gray")

    plt.show()

    print("\n\nFinal Output: {}".format(np.argmax(prediction)))

    print ("\nPrediction (Softmax) from the neural network:\n\n {}".format(prediction))

    hard_maxed_prediction = np.zeros(prediction.shape)

    hard_maxed_prediction[0][np.argmax(prediction)] = 1

    print ("\n\nHard-maxed form of the prediction: \n\n {}".format(hard_maxed_prediction))

    print ("\n\n-------------------------------------\n\n")
```

8 TESTING

8.1 Test Cases

8. TESTING Test case ID	Feature Type	Component	Test Scenario	Expected Result	Actual Result	Status
Homepage_TC_OO1	Functional	Home Page	Verify user is able to see the Homepage when clicked on the link	Home Page should be displayed.	Working as expected	Pass
Homepage_TC_OO2	UI	Home Page	Verify the UI elements in Homepage	Application should show below UI elements: a.choose file button b.predict button c.clear button	Working as expected	Pass
Homepage_TC_OO3	Functional	Home Page	Verify user is able to choose file from the local system and click on predict	Choose file popup screen must be displayed and user should be able to click on predict button	Working as expected	Pass
Homepage_TC_OO4	Functional	Home page	Verify user able to select invalid file format	Application won't allow to attach formats other than ".png, .jiff, .pjp, .jpeg, .jpg, .pjpeg"	Working as expected	Pass

Predict_TC_OO5	Functional	Predict page	Verify user is able to navigate to the predict to and view the predicted result	User must be navigated to the predict page and must view the predicted result	Working as expected	Pass

8.2 User Acceptance Testing

Defect Analysis Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	0	0	0	0	0
Duplicate	0	0	0	0	0
External	0	0	0	0	0
Fixed	0	0	0	0	0
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	0	0	0	0	0

TEST CASES ANALYSIS

Section	Total Cases	Not Tested	Fail	Pass
Client Application	5	0	0	5
Security	5	0	0	5
Final Report Output	5	0	0	5
Performance	5	0	0	5

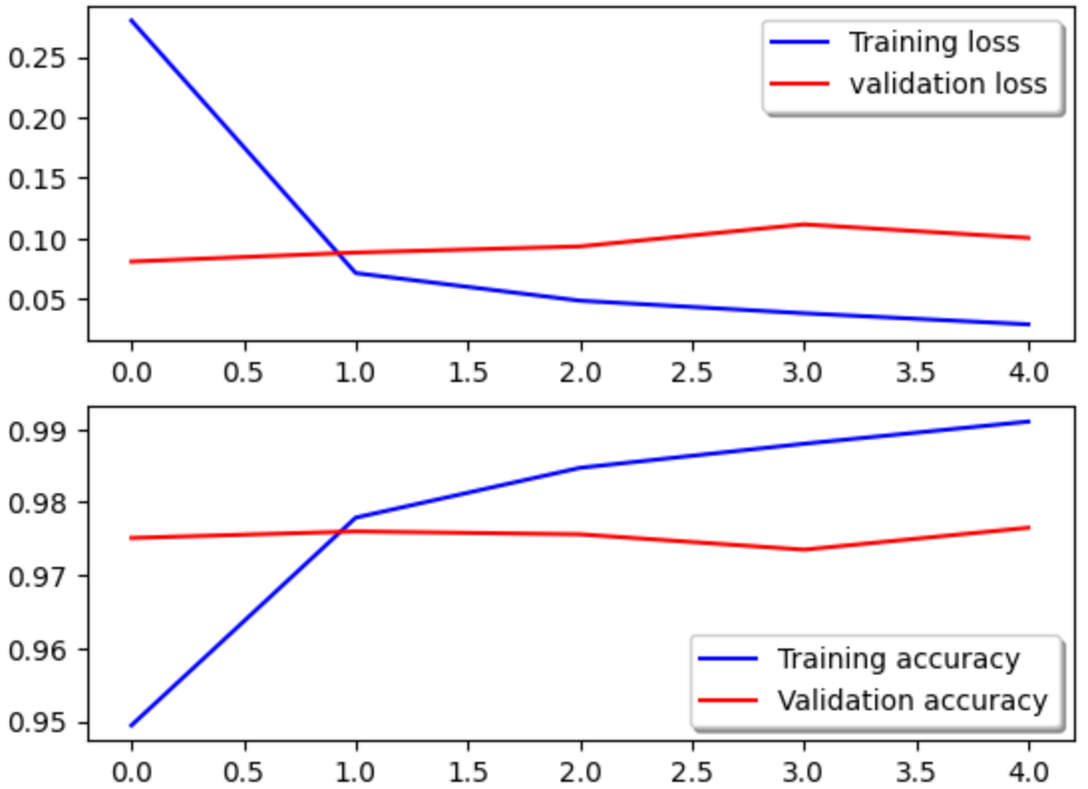
9. RESULTS

9.1 Performance Metrics

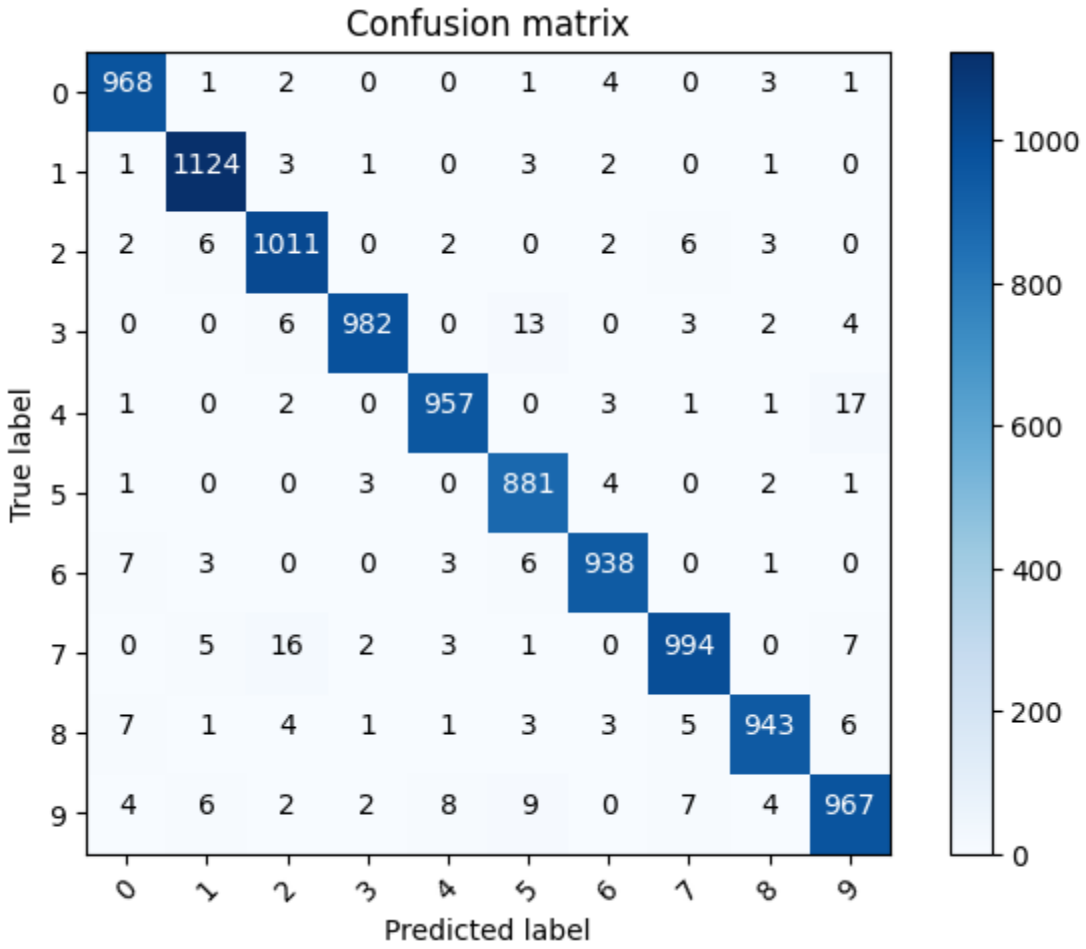
Model Summary:

```
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
_________________________________________________________________
None
```

Accuracy:


Training loss
validation loss
0.25
0.20
0.15
0.10
0.05
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
0.99
0.98
0.97
0.96
0.95
Training accuracy
Validation accuracy
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

Confusion Matrix:



Classification Report:

	precision	recall	f1-score	support
0	0.98	0.99	0.98	980
1	0.98	0.99	0.99	1135
2	0.97	0.98	0.97	1032
3	0.99	0.97	0.98	1010
4	0.98	0.97	0.98	982
5	0.96	0.99	0.97	892
6	0.98	0.98	0.98	958
7	0.98	0.97	0.97	1028
8	0.98	0.97	0.98	974
9	0.96	0.96	0.96	1009
accuracy			0.98	10000
macro avg	0.98	0.98	0.98	10000
weighted avg	0.98	0.98	0.98	10000

Performance Metrics Result:

Locust Test Report

During: 11/15/2022, 10:52:19 PM - 11/15/2022, 10:56:36 PM

Target Host: http://127.0.0.1:5000/

Script: locustfile.py

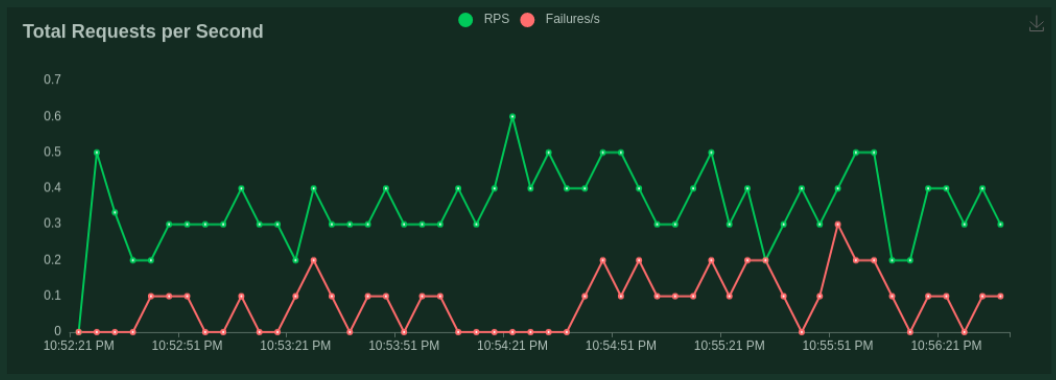
Request Statistics

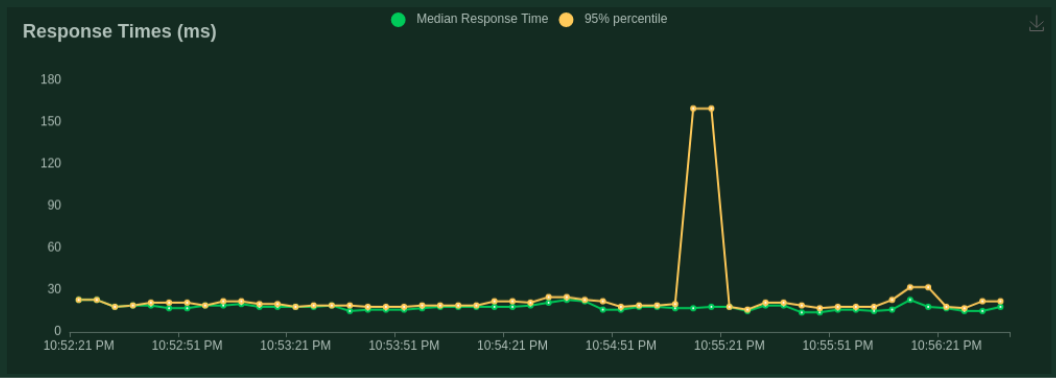
Method	Name	# Requests	# Fails	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	RPS	Failures/s
GET	//	67	0	17	12	24	5875	0.3	0.0
GET	//predict	23	23	21	11	163	265	0.1	0.1
	Aggregated	**90**	**23**	**18**	**11**	**163**	**4441**	**0.4**	**0.1**

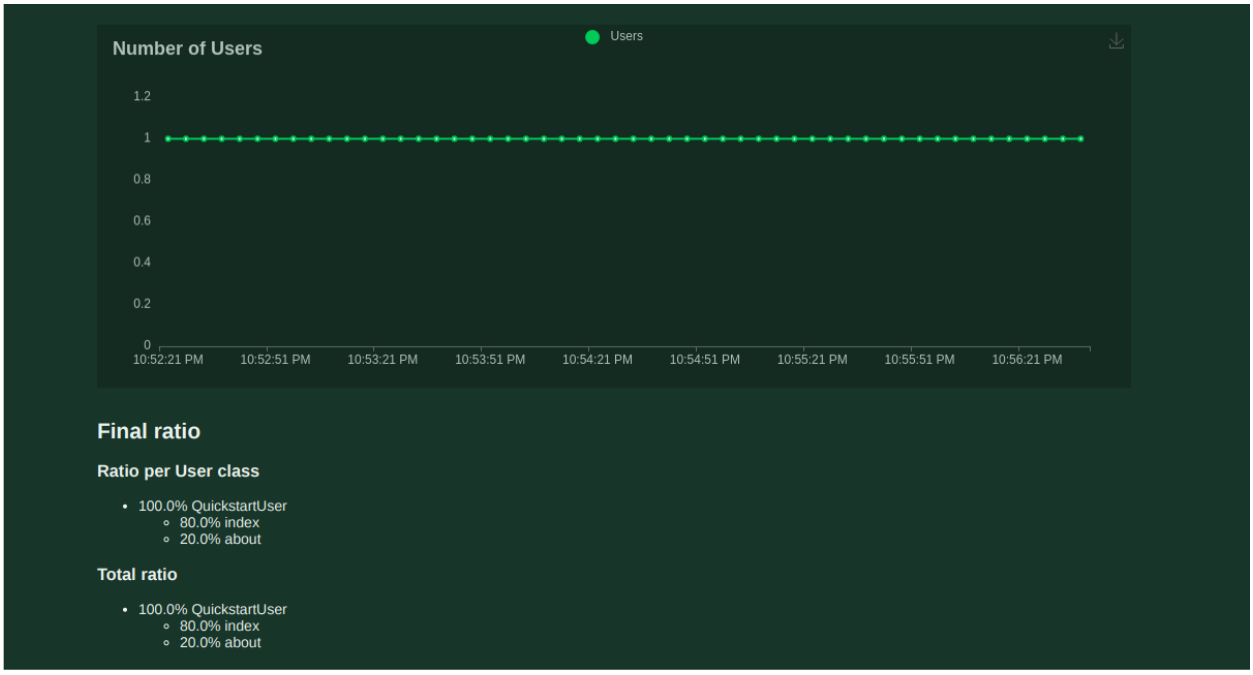
Response Time Statistics

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	//	18	18	19	19	22	23	25	25
GET	//predict	15	15	16	16	17	32	160	160
	Aggregated	**17**	**18**	**18**	**19**	**22**	**23**	**160**	**160**

Charts







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 high performance server for faster predictions.

✓ Prone to occasional errors.

11. CONCLUSION

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. There is so much room for improvement, which can be implemented in subsequent versions.

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.

13. APPENDIX

Source Code

index.html:

```
<html>

<head>
  <title>HDR</title>
```

```
  <meta name="viewport" content="width=device-width">

  <link href="https://fonts.googleapis.com/css2?family=Prompt:wght@600&display=swap"
rel="stylesheet">
  <link href="https://fonts.googleapis.com/css2?family=Varela+Round&display=swap"
rel="stylesheet">
  <link href="https://fonts.googleapis.com/css2?family=Source+Code+Pro:wght@500&display=swap"
rel="stylesheet">
  <link
href="https://fonts.googleapis.com/css?family=Calistoga|Josefin+Sans:400,700|Pacifico&display
=swap" rel="stylesheet">

  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
  <link rel="stylesheet" type= "text/css" href= "{{
url_for('static',filename='css/style.css') }}">

  <script src="https://kit.fontawesome.com/b3aed9cb07.js" crossorigin="anonymous"></script>

  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous"></script>
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
  <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest"></script>
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css">
  <script src="https://cdn.jsdelivr.net/npm/jquery@3.6.0/dist/jquery.slim.min.js"></script>









  <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></script>
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js"></script>

</head>
<style>
    body{
     background-image: url('static/images/bc1.jpg');
```

```
     background-repeat: no-repeat;
     background-size: cover;
    }
</style>

<script>
  function preview() {
    frame.src=URL.createObjectURL(event.target.files[0]);
}

    $(document).ready(function() {
          $('#clear_button').on('click', function() {
              $('#image').val('');
              $('#frame').attr('src',"");
            });
        });

</script>

<body>
    <h1>HandWritten Digit Recognition System</h1>
        <div class="container p-3 my-3 bg-dark text-white">
            <p>Handwritten Digit 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 email, posts, cheque etc. Before proper implementation of this
technology we have relied on writing text with our own hands which can result in error.It's
difficult to store and access physical data with efficiency.The project presents in
representing the recognization of handwritten digits (0 - 9) from the famous MNIST dataset.
Here we will be using Convolutional Neural Network for the prediction.</p>
        </div>
        <section id="content">

            <div class="leftside">
            <form action="/predict" method="POST" enctype="multipart/form-data">
            <label>Select a image:</label>
            <input id="image" type="file" name="image" accept="image/png, image/jpeg"
onchange="preview()"><br><br>
              <img id="frame" width="100px" height="100px"/>
              <div class="buttons_div">
```

```
                <button type="submit" class="btn btn-light">Predict</button>
                <button type="button" class="btn btn-light">  Clear  </button>
              </div>
            </form>
            </div>
      </section>


</body>

</html>
```

Predict.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Prediction</title>
</head>

<style>
    body{
     background-image: url('static/images/bc1.jpg');
     background-repeat: no-repeat;
     background-size: cover;
    }

    #rectangle{
     width:600px;
     height:150px;
     background-color: #000000;
     border-radius: 25px;
     position:absolute;
     box-shadow: 0px 0px 10px 5px white;
     top:25%;
     left:50%;
     transform:translate(-50%,-50%);
    }

    #num{
  text-align: center;
  font-size: 30px;
  margin: 0 auto;
  padding: 3% 5%;
  padding-top: 8%;
```

```
  color: white;
    }

</style>

<body>

    <div id="rectangle">
        <h1 id="num">Predicted Number is {{num}}</h1>
    </div>

</body>
</html>
```

Style.css

```
  #clear_button{
    margin-left: 15px;
    font-weight: bold;
    color: rgb(0, 174, 255);
  }

  #confidence{
    font-family: 'Josefin Sans', sans-serif;
    margin-top: 7.5%;
  }

  #content{
    margin: 0 auto;
    padding: 2% 15%;
    padding-bottom: 0;
  }

  .welcome{
     text-align: center;
     position:  relative;
     color: rgb(0, 32, 112);
     background-color: skyblue;
     padding-top: 1%;
     padding-bottom: 1%;
     font-weight: bold;
     font-family: 'Bookman', 'URW Bookman L', serif;
  }

  #team_id{
     text-align: right;
```

```
    font-size: 25px;
    padding-right: 3%;
}

#predict_button{
  margin-right: 15px;
  color: rgb(0, 255, 72);
  font-weight: bold;
}

#prediction_heading{
  font-family: 'Josefin Sans', sans-serif;
  margin-top: 7.5%;
}

#result{
  font-size: 5rem;
}

#title{
  padding: 1.5% 15%;
  margin: 0 auto;
  text-align: center;
}

.btn {
    font-size: 15px;
    padding: 10px;
    /* -webkit-appearance: none; */
    background: #eee;
    border: 1px solid #888;
    margin-top: 20px;
    margin-bottom: 20px;
}

.buttons_div{
  margin-bottom: 30px;
  margin-right: 80px;
}

.heading{
  font-family:"American Typewriter", serif;
  font-weight: 700;
  font-size: 2rem;
  display: inline;
}

.leftside{
```

```
    text-align: center;
    margin: 0 auto;
    margin-top: 2%;
    /* padding-left: 10%; */
  }

  #frame{
    margin-right: 10%;
  }

  .predicted_answer{
    text-align: center;
    margin: 0 auto;
    padding: 3% 5%;
    padding-top: 0;
    /* padding-left: 10%; */
  }

  h1{
    text-align: center;
    color: aliceblue;
    padding: 100px 50px 65px 100px;
}

  @media (min-width: 720px) {
    .leftside{
      padding-left: 10%;
    }
  }
```

FLASK:

app.py:

```
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__)
```

```python
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

model = load_model("./DigitRecog_IBM_model/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)
```

MODEL CREATION:

```
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
    (x_train, y_train), (x_test, y_test)=mnist.load_data ()
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  #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)
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'])
x_train = np.asarray(x_train)
y_train = np.asarray(y_train)
history = model.fit(x_train, y_train, validation_data=(x_test, y_test), epochs=5,
batch_size=32)

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

fig, ax = plt.subplots(2,1)
ax[0].plot(history.history['loss'], color='b', label="Training loss")
ax[0].plot(history.history['val_loss'], color='r', label="validation loss",axes =ax[0])
legend = ax[0].legend(loc='best', shadow=True)

ax[1].plot(history.history['accuracy'], color='b', label="Training accuracy")
ax[1].plot(history.history['val_accuracy'], color='r',label="Validation accuracy")
legend = ax[1].legend(loc='best', shadow=True)

def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
```

```
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, cm[i, j],
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

# Predict the values from the validation dataset
Y_pred = model.predict(x_test)
# Convert predictions classes to one hot vectors
Y_pred_classes = np.argmax(Y_pred,axis = 1)
# Convert validation observations to one hot vectors
Y_true = np.argmax(y_test,axis = 1)
# compute the confusion matrix
confusion_mtx = confusion_matrix(Y_true, Y_pred_classes)
# plot the confusion matrix
plot_confusion_matrix(confusion_mtx, classes = range(10))

import sklearn
print(sklearn.metrics.classification_report(Y_true, Y_pred_classes))


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

plt.imshow(x_test[5100])

import numpy as np
print(np.argmax(prediction, axis=1))

np.argmax(y_test[5100:5101]) #printing the actual labels
# Save the model
model.save('models/mnistCNN.h5')
```

GitHub & Project Demo Link

GitHub Link https://github.com/IBM-EPBL/IBM-Project-37435-1660309097

Demo Link https://youtu.be/NpaNCiiX6Y0