A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM

TABLE OF CONTENTS

1. INTRODUCTION

1.1. PROJECT OVERVIEW

Traditional methods of recognising handwriting rely heavily on a lot of prior knowledge like Optical Character Recognition (OCR). Since the style of handwriting changes with every individual, it is a challenging task in identifying the characters correctly. The thickness of stroke, style carries uniqueness with different person depending on them. The rapid growth in the need for digitizing handwritten data and the availability of massive processing power demands improvement in recognition accuracy. Hence a highly proficient algorithm is required when dealing with handwriting recognition. Handwritten digit recognition can be done using deep learning methods effectively. The Convolutional Neural Networks (CNN) is a deep learning algorithm that is highly suitable for image recognition and those tasks involving processing of pixel data. MNIST data set is widely used for this recognition process and it has 70000 handwritten digits. Those images are split as train set and test set images. Artificial neural networks is used to train these images and build a deep learning model. Web application is created where the user can upload an image of a handwritten digit. this image is analysed by the model and the detected result is returned on to UI.

1.2. PURPOSE

Each individual has a unique handwriting style which makes it a bit complex to identify the digits. If the handwritten digit recognition becomes an efficient practice, this will help digitize number processing. Huge amounts of data can be processed by machine which will save loads of time. In today’s world, technology plays a major role in handling data, therefore it is important to bring this system in managing data. Workers at the postal office sorting throughs mails using the postal code can be helped using this. This also comes handy while arranging records and huge amounts of information. Manual labour is eased and it saves up a lot of time. It can be used in programming checks and in case of tax documentation. The labour cost will also be reduced with the help of machines. There are also the activities of processing bank checks and tax documentations. Large piles of records and archives can be arranged and sorted well easing the stress and work load from manual labourers.

2. LITERATURE SURVEY

2.1. EXISTING PROBLEM

Because of the progress in science and technology everything is being digitalised to reduce human effort. It takes a lot of time and effort on the side of manual workers when sorting through mails by postal codes. It is not an easy task to handle data by human worker. There is also the possibility of human error while processing huge amount of data. Therefore, digitizing these will help reduce time and labour. The labour cost will also be reduced with the help of machines. There are also the activities of processing bank checks and tax documentations. Large piles of records and archives can be arranged and sorted well easing the stress and work load from manual labourers. The problem with handwriting is that every individual has different style of writing. There is a differing thickness of stroke, style and general uniqueness that just brings a level of hardness in identifying the handwriting. The machine must be capable of picking up the digits correctly with a good accuracy rate. Hence a highly proficient algorithm is required when dealing with handwriting recognition. Handwritten digit recognition can be done using deep learning methods effectively.

2.2. REFERNCES

[1] Handwritten Digit Recognition using Machine and Deep Learning Algorithms by Ritik Dixit of Computer Science and Engineering, Rishika Kushwah of Computer Science and Engineering, Samay Pashine of Computer Science and Engineering, Acropolis Institute of Technology & Research, Indore, India, 23 June 2021.

[2] Recognition of Handwritten Digit using Convolutional Neural Network (CNN), By Md. Anwar Hossain & Md. Mohon Ali, Pabna University of Science & Technology, 2019.

[3] On the Brittleness of Handwritten Digit Recognition Models, Alexander K. Seewald, 30 Nov 2011.

[4] A Novel Handwritten Digit Classification System Based on Convolution al Neural Network Approach, Ali Abdullah Yahya, 18 September 2021.

[5] Handwritten Digit Recognition Using Machine Learning Algorithms, S. M. Shamim, Md Badrul Alam Miah, Angona Sarker, Masud Rana, Abdullah Al Jobair, March 2018.

[6]. Handwritten digit recognition using deep learning, International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), vol. 6, no. 7, A. Dutta and A. Dutta, July 2017.

[7]. Automatic prediction of age, gender, and nationality in offline handwriting. EURASIP Journal on Image and Video Processing, no. 1, Al Maadeed, Somaya, and Abdelaali Hassaine, 2014.

[8]. Handwritten DigitsRecognition, Project Report, Gaurav Jain, Jason Ko, University of Toronto, 11/21/2008.

[9]. Handwritten recognition using SVM, KNN an neural network, arXiv preprint arXiv:1702.00723. Hamid, Norhidayu Abdul, and NilamNur Amir Sjarif, 2017.

[10]. Digit Classification using the MNIST Dataset, M. Wu and Z. Zhang, Handwritten, 2010.

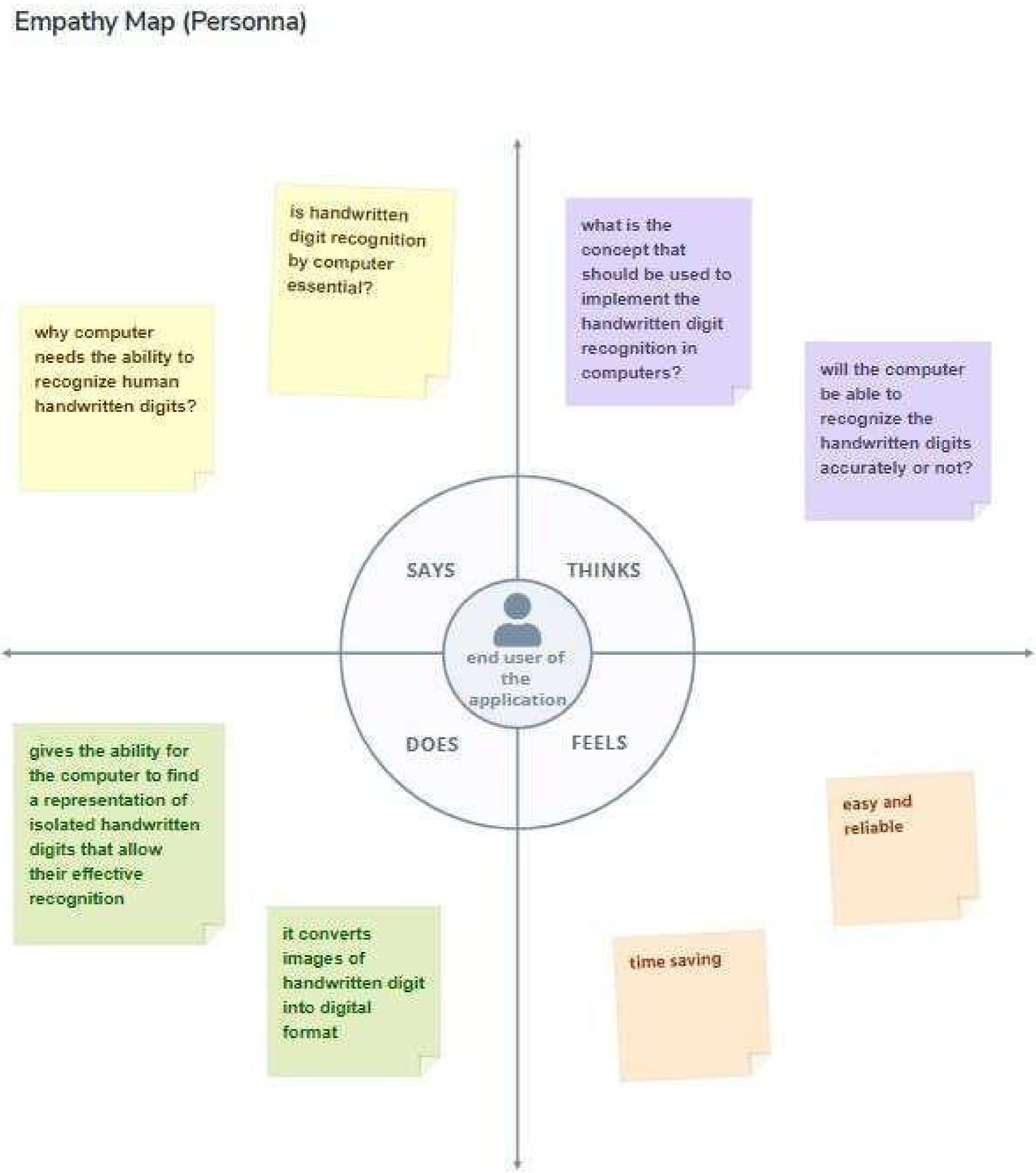
[11]. Handwritten digit classification using support vector machines, R.G.Mihalyi, 2011.

2.3. PROBLEM STATEMENT DEFINITION

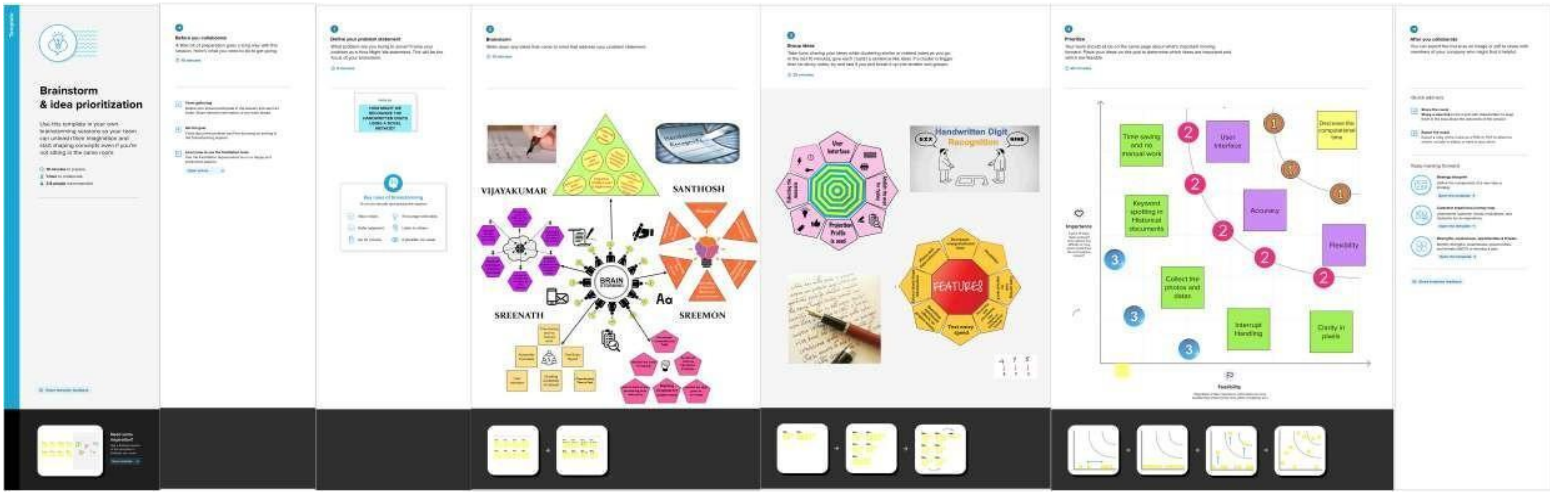
Handwriting recognition is one of the compelling research works going on because every individual in this world has their own style of writing. Since the style of handwriting changes with every individual, it is a challenging task in identifying the characters correctly. The thickness of stroke, style carries uniqueness with different person depending on them. It is the capability of the computer to identify and understand handwritten digits or characters automatically. Because of the progress in the field of science and technology, everything is being digitalized to reduce human effort. Hence, there comes a need for handwritten digit recognition in many real-time applications. MNIST data set is widely used for this recognition process and it has 70000 handwritten digits. Artificial neural network is used to train these images and build a deep learning model. The Convolutional Neural Networks (CNN) is a deep learning algorithm that is highly suitable for image recognition and those tasks involving processing of pixel data. Convolutional neural networks (CNNs) are very effective in perceiving the structure of handwritten characters/words in ways that help in automatic extraction of distinct features and make CNN the most suitable approach for solving handwriting recognition problems. Our aim in the proposed work is to deploy the CNN model effectively and produce a good result with better accuracy. The main objective was to actualize a pattern characterization method to perceive the handwritten digits provided in the MINIST data set of images of handwritten digits (0‑9). Web application is created where the user can upload an image of a handwritten digit. This image is analysed by the model and the detected result is returned on to UI.

3. IDEATION AND PROPOSED SOLUTION

3.1. EMPATHY MAP CANVAS



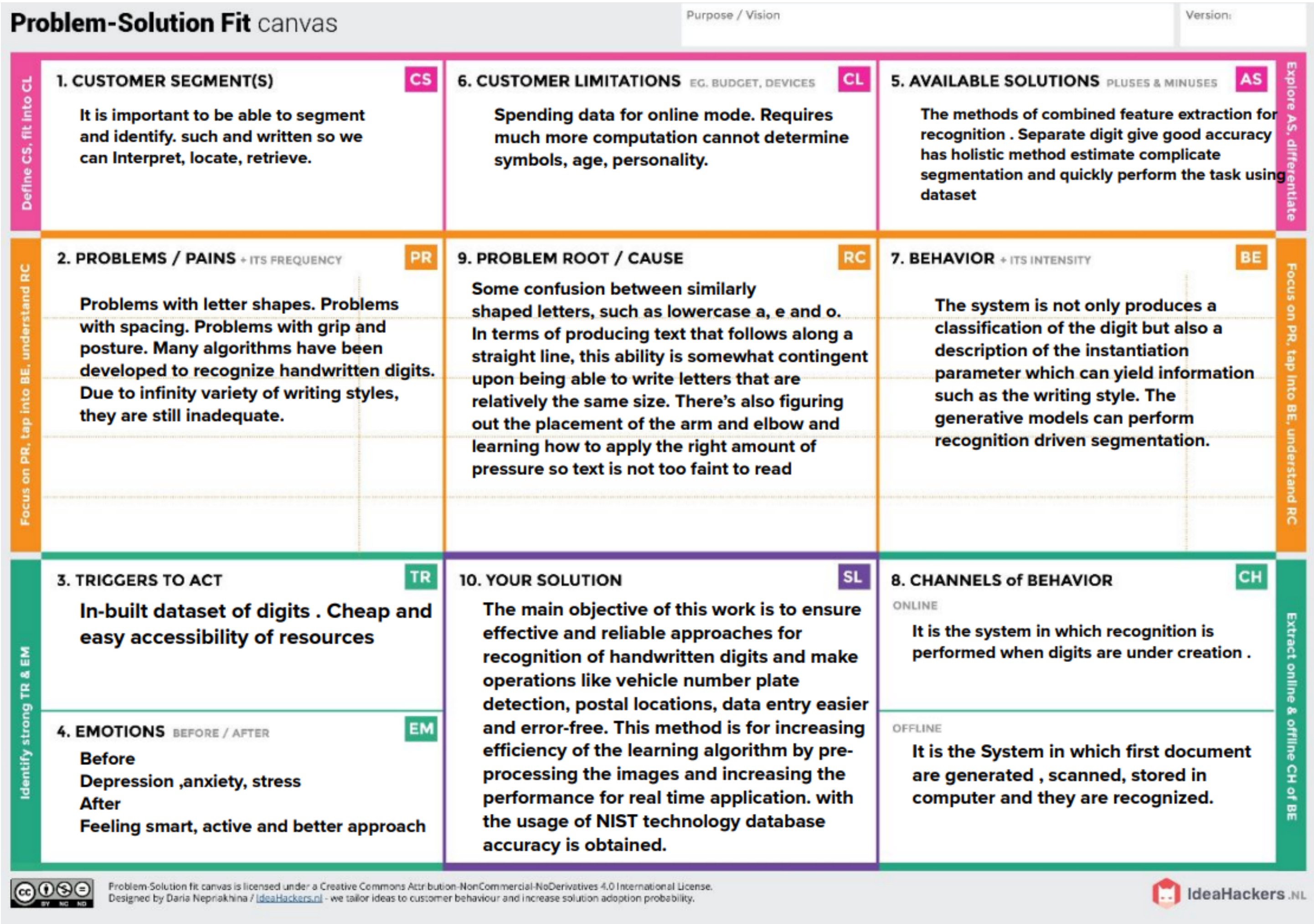
3.2. IDEATION AND BRAINSTORMING



3.3. PROPOSED SOLUTION

S No.	Parameter	Description
1.	Problem statement (problem to be solved)	The problem is machines cannot recognize handwritten digits because handwritten digits can be made with different shapes, size and forms. Handwritten digit recognition system can tackle this problem.
2.	Idea / Solution Description	Using MNIST dataset over the neural network algorithms, it is possible to recognize the digits which is useful for banks sectors, data entry etc.
3.	Novelty / Uniqueness	Using Convolutional Neural Network(CNN) gives greater accuracy and it can detect automatically without any human supervision.
4.	Social impact / Customer Satisfaction	Greater satisfaction by matching the percentage of digits and accuracy recognition without delay , as artificial intelligence reduce the human effort.
5.	Business Model (financial benefit)	Collaboration with bank sectors, government sectors like RTO which is used for detecting cheque, vehicle number detection etc.
6.	Scalability of Solution	The handwriting will be detected by any of the formats such as image, documents , etc.. so that it can be user friendly and flexible where there will be a growth for the users.

3.4. PROBLEM SOLUTION FIT



Problem-Solution Fit canvas

Purpose / Vision

Version:

Define CS, fit into CL

1. CUSTOMER SEGMENT(S) CS

It is important to be able to segment and identify. such and written so we can Interpret, locate, retrieve.

6. CUSTOMER LIMITATIONS EG. BUDGET, DEVICES CL

Spending data for online mode. Requires much more computation cannot determine symbols, age, personality.

5. AVAILABLE SOLUTIONS PLUSES & MINUSES AS

The methods of combined feature extraction for recognition . Separate digit give good accuracy has holistic method estimate complicate segmentation and quickly perform the task using dataset

Explore AS, differentiate

Focus on PR, tap into BE, understand RC

2. PROBLEMS / PAINS + ITS FREQUENCY PR

Problems with letter shapes. Problems with spacing. Problems with grip and posture. Many algorithms have been developed to recognize handwritten digits. Due to infinity variety of writing styles, they are still inadequate.

9. PROBLEM ROOT / CAUSE RC

Some confusion between similarly shaped letters, such as lowercase a, e and o. In terms of producing text that follows along a straight line, this ability is somewhat contingent upon being able to write letters that are relatively the same size. There's also figuring out the placement of the arm and elbow and learning how to apply the right amount of pressure so text is not too faint to read

7. BEHAVIOR + ITS INTENSITY BE

The system is not only produces a classification of the digit but also a description of the instantiation parameter which can yield information such as the writing style. The generative models can perform recognition driven segmentation.

Focus on PR, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TO ACT TR

In-built dataset of digits . Cheap and easy accessibility of resources

4. EMOTIONS BEFORE / AFTER EM

Before
Depression ,anxiety, stress
After
Feeling smart, active and better approach

10. YOUR SOLUTION SL

The main objective of this work is to ensure effective and reliable approaches for recognition of handwritten digits and make operations like vehicle number plate detection, postal locations, data entry easier and error-free. This method is for increasing efficiency of the learning algorithm by pre-processing the images and increasing the performance for real time application. with the usage of NIST technology database accuracy is obtained.

8. CHANNELS of BEHAVIOR CH

ONLINE

It is the system in which recognition is performed when digits are under creation .

OFFLINE

It is the System in which first document are generated , scanned, stored in computer and they are recognized.

Extract online & offline CH of BE

4. REQUIREMENT ANALYSIS

4.1. FUNCTIONAL REQUIREMENTS

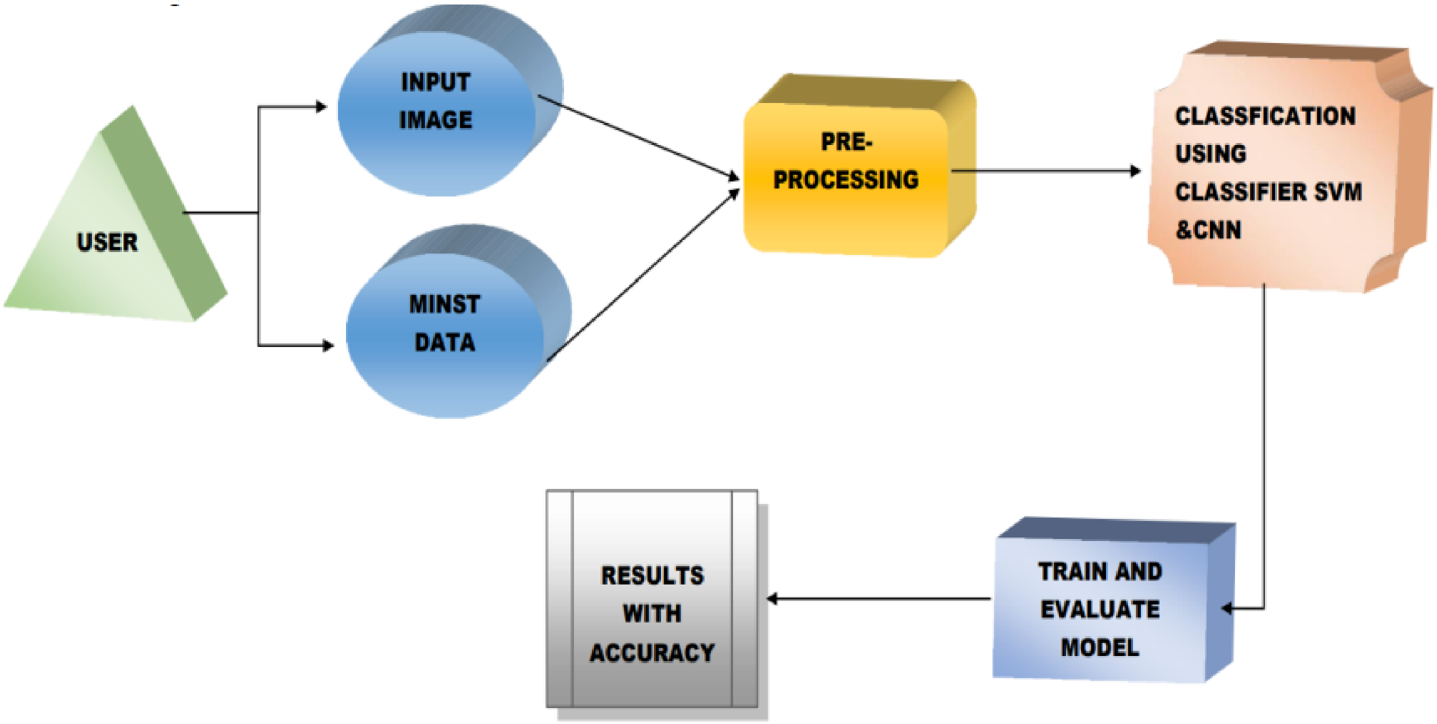
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Implementation	Importing the libraries Loading the datasets
FR-2	Hidden procedure	Pre-processing data, creating and training model
FR-3	User input/user output	GUI interface for drawing the digits
FR-4	User confirmation	Accuracy percentage with help of sample datasets

4.2. NON-FUNCTIONAL REQUIREMENTS

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Bank check processing, postal mail sorting, form data entry, vehicle number detection
NFR-2	**Security**	False alarms and missed events
NFR-3	**Reliability**	Rich in accepting all type of inputs
NFR-4	**Performance**	Accuracy and effective recognition
NFR-5	**Availability**	GUI interface for input , datasets with training and testing images
NFR-6	**Scalability**	High speed, robustness , user friendly and flexible as it accepts all type of formats such as image, documents etc

5. PROJECT DESIGN

5.1. DATA FLOW DIAGRAM



5.2. SOLUTION AND TECHNICAL ARCHITECTURE

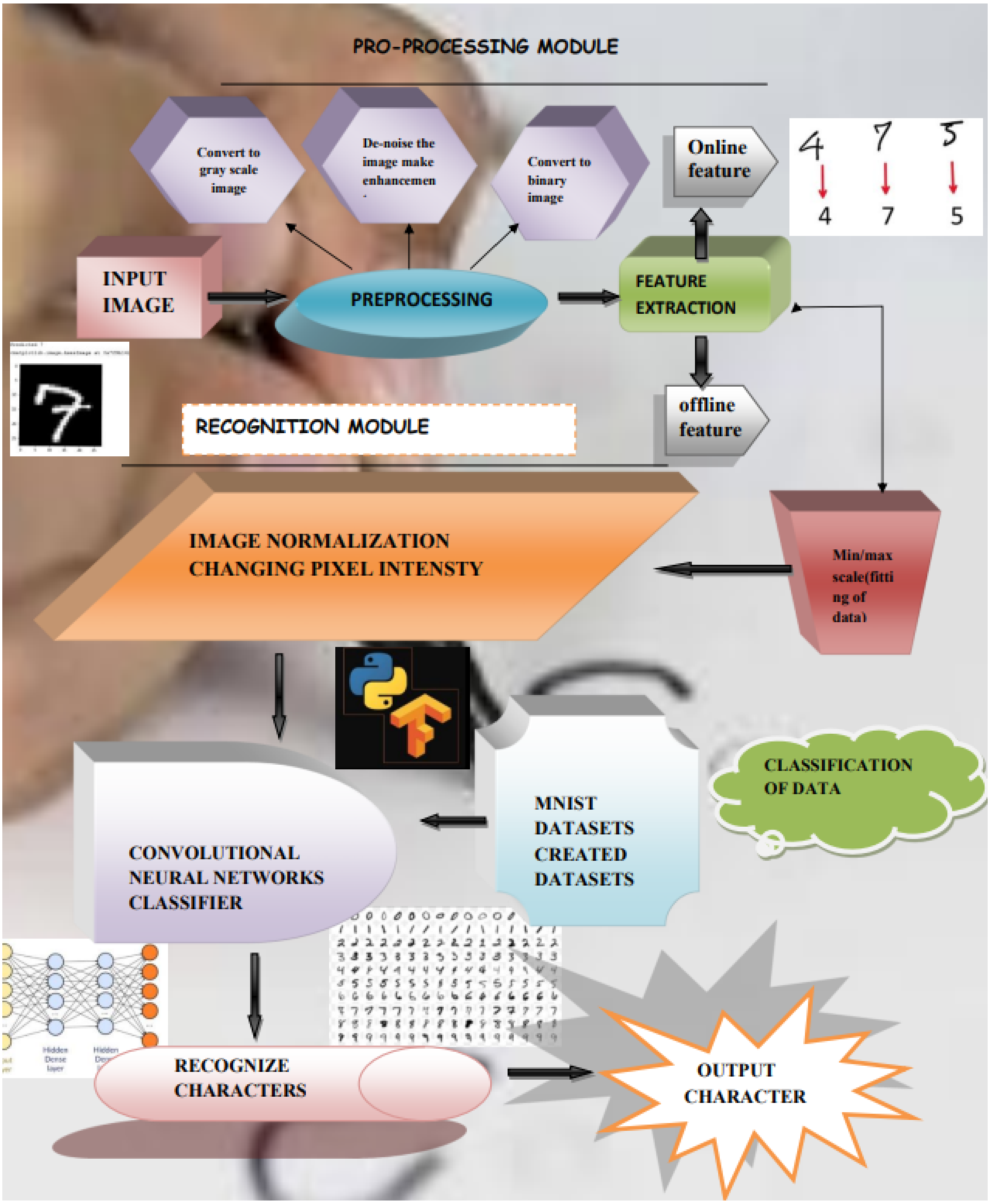
Solution Architecture:

Handwritten Digit Recognition can be done with the help of the deep learning algorithm, Convolutional Neural Network (CNN) which works similar to that of the neurons in human brain. The MNIST dataset containing 70,000 images of handwritten digits is loaded and pre-processed. The dataset is split as training and testing set and then the CNN model is created and saved. The model is used for identifying the handwritten digit from the user.

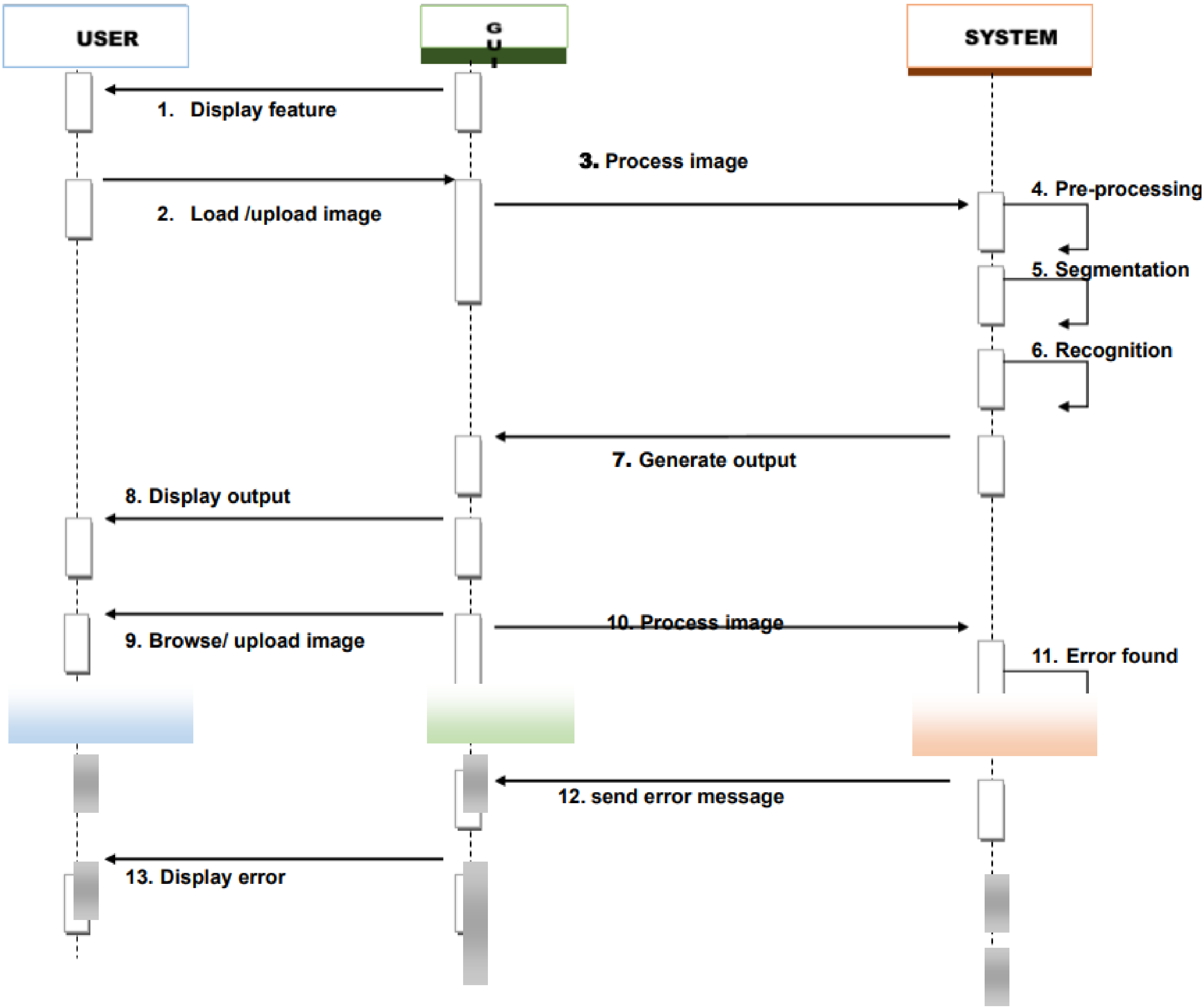
The major steps involved in this:

1. Load the dataset
2. Splitting into training and testing
3. CNN modelling
    - 3.1. Convolution
    - 3.2. Pooling
    - 3.3. Fully connected
4. Output prediction

Architecture Diagram:



Technical Architecture:



Components & Technologies:

S No.	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc	HTML, CSS, JavaScript / Angular Js / React Js etc
2.	Application Logic-1	Logic for a process in the	Python

		application	
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 Cloud	IBM DB2, IBM Cloudant etc
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
9.	External API-2	Purpose of External API used in the application	Aadhar API etc
10.	Machine Learning Model	Purpose of Machine Learning Model	Hand written Recognition Model etc
11.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration Cloud Server Configuration	Local, Cloud Foundry, Kubernetes etc

Application Characteristics:

S No.	Characteristics	Description	Technology
1.	Open-Source Frameworks	Open-source frameworks used	Technology of Opensource framework
2.	Security Implementations	False alarms and missed events	SHA-256, Encryptions, IAM Controls, OWASP etc
3.	Scalable Architecture	High speed, robustness , user friendly and flexible as it accepts all type of formats such as image, documents etc	Technology used
4.	Availability	Justify the availability of application (Example: Use of load balancers, distributed servers etc)	Technology used
5.	Performance	Accuracy and effective recognition	Technology used

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	Sprint-1
		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	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard					
Customer (Web user)						
Customer Care Executive						
Administrator						

6. PROJECT PLANNING & SCHEDULING

6.1. SPRINT PLANNING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	1	High	SREENATH
Sprint-1	Login	USN-2	As a user, I can login into the application by entering email and password	3	High	SREEMON
Sprint-2	Prediction	USN-3	As a user, I can predict the word	5	Low	VIJAYA KUMAR
Sprint-2	Uploading	USN-4	As a user, I can input the image of digital documents to the application	3	Medium	SANTHOSH
Sprint-3	Upload Image of Handwritten document	USN-5	As a user, I can able to input the images of the handwritten documents or images to the application	1	High	SREENATH
Sprint-3	Recognize text	USN-6	As a user, I can able to choose the font of the text to be displayed	1	Medium	SREEMON
Sprint-4	Recognize digit	USN-7	As a user I can able to get the recognised digit as output from the images of digital documents or images	5	Low	VIJAYA KUMAR
Sprint-4	Recognize digit	USN-8	As a user I can able to get the recognised digit as output from the images of handwritten documents or images	3	Medium	SANTHOSH

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	2	6 Days	24 Oct 2022	29 Oct 2022	2	29 Oct 2022
Sprint-2	2	6 Days	31 Oct 2022	05 Nov 2022	1	05 Nov 2022
Sprint-3	2	6 Days	07 Nov 2022	12 Nov 2022	1	12 Nov 2022
Sprint-4	2	6 Days	14 Nov 2022	19 Nov 2022	1	19 Nov 2022

6.3. REPORTS FROM JIRA

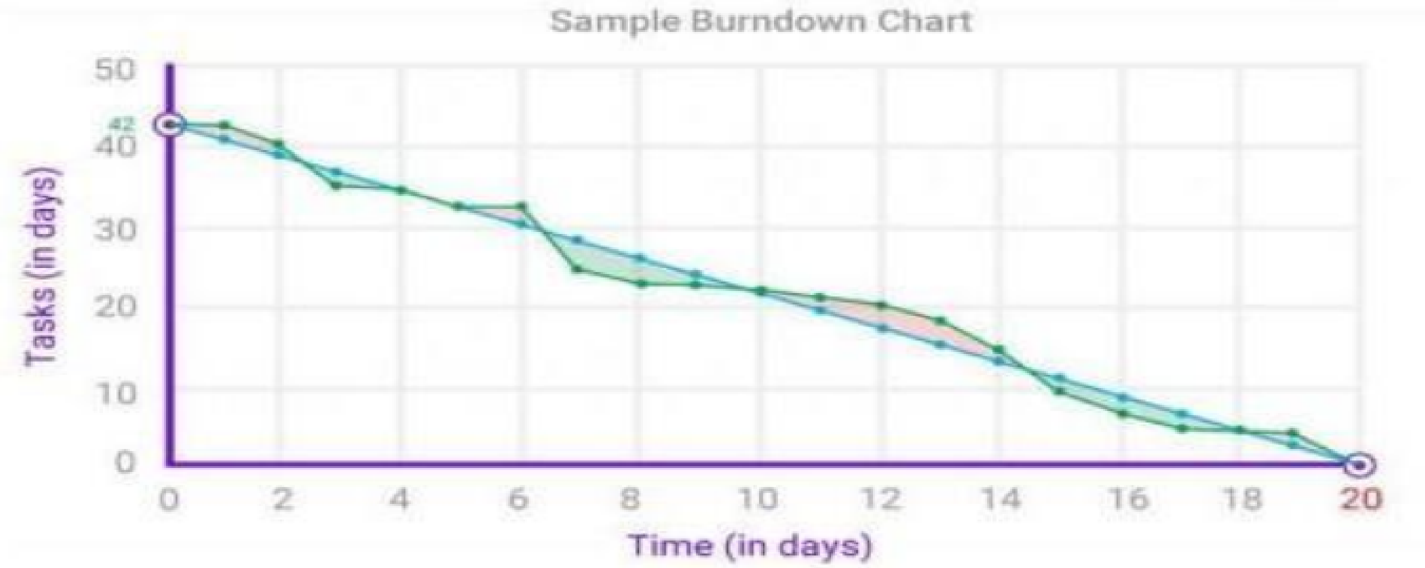
Velocity:

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).

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

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.



Sample Burndown Chart
50
42
40
30
20
10
0
Tasks (in days)
0
2
4
6
8
10
12
14
16
18
20
Time (in days)

7. CODING AND SOLUTIONING

7.1. FEATURE-1 MODEL BUILDING

ML depends heavily on data, without data, it is impossible for a machine to learn. It is the most crucial aspect that makes algorithm training possible. In Machine Learning projects, we need a training data set. It is the actual data set used to train the model for performing various actions. TensorFlow already has MNIST Data set so there is no need to explicitly download or create Dataset. The MNSIT dataset contains ten classes: Digits from 0-9. Each digit is taken as a class. The required libraries are imported which are required for the model to run. The dataset for this model is imported from the Keras module. The data is split into train and test. Using the training dataset, the model is trained and the testing dataset is used to predict the results. Basically, the pixel values range from 0-255. The value of each image is stored is y_train. The model is built with convolutional, pooling and dense layers. The created model is then compiled and saved.

7.2. FEATURE-2 WEB APP

HTML, CSS and JavaScript are used to create the web pages for the front end. An html page that takes in image files as input using form and submits to back end is created. A flask app is created using python flask, where it receives the image files from the templates, html pages and the prediction operation is done over this image. Later the predicted output is sent to the result page.

8. TESTING

8.1. TEST CASES

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)
HomePage_TC_001	Functional	Home Page	Verify user is able to see the navigation bar on top		1.Enter URL and click go 2.Click on the Recognise button on navigation bar	Move to recognise page	Working as expected	Pass		Y
RecognizePage_TC_002	Functional	RecognizePage	Verify user is able to move to recognise page		1.Enter URL and click go 2.Click on the Recognise button on navigation bar 3.Click on select file button on the view page	1.user should be navigate to our computer image folder.	Working as expected	pass		Y
RecognizePage_TC_003	Functional	Recognizepage	Verify user is moved to predict page.		1.Enter URL and click go 2.Click on the Recognise button on navigation bar 3.Click on select file button on the view page. 4.click on the recognise button.	1.move to predict page.	Working as expected	Pass		N
PredictPage_TC_004	Non-Functional	Predictpage	Verify whether digit is predicted correctly.		1.Enter URL and click go 2.Click on the Recognise button on the navigaor bar 3.click on select file button on the view page 4.Choose on image as you want to predict. 5.Click on recognise button	1.User expects the image to be predicted correctly.	There are incorrect predictions at times	Fail	The accuracy of the system affects the results	N
BackPage_TC_005	Functional	BackPage	In case of incorrect prediction or user wants another image predicted, then user clicks on back button.		1.Enter URL and click go 2.Click on the Recognise button on the navigaor bar 3.click on selct file button on the view page 4.Choose on image as you want to predict. 5.click on recognise Button. 6.click on Back Button.	1.user is moved back to recognise page	Working as expected	Pass		Y

8.2. USER ACCEPTANCE TESTING

Defect Analysis:

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved.

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	13	2	1	2	18
Duplicate	4	0	2	0	6
External	3	2	1	0	6
Fixed	12	3	2	17	34
Not Reproduced	0	2	0	0	2
Skipped	0	0	2	1	3
Won't Fix	0	3	4	1	8
Totals	32	12	13	21	77

Test Case Analysis:

This report shows the number of test cases that have passed, failed, and untested.

Section	Total Cases	Not Tested	Fail	Pass
Client Application	37	0	0	37
Image	14	0	0	14
Prediction	5	0	2	3
Section	**Total Cases**	**Not Tested**	**Fail**	**Pass**
Exception Reporting	7	0	0	7
Final Report Output	4	0	0	4
Version Control	2	0	0	2

9. RESULTS

9.1. PERFORMANCE METRICS

S.No.	Parameter	Values	Screenshot
1.	Model Summary		Model: "sequential_1" Layer (type) Output Shape Param # conv2d_2 (Conv2D) (None, 26, 26, 64) 640 conv2d_3 (Conv2D) (None, 24, 24, 64) 36928 max_pooling2d_1 (MaxPooling2D) (None, 12, 12, 64) 0 dropout_2 (Dropout) (None, 12, 12, 64) 0 flatten_1 (Flatten) (None, 9216) 0 dense_2 (Dense) (None, 256) 2359552 dropout_3 (Dropout) (None, 256) 0 dense_3 (Dense) (None, 10) 2570 Total params: 2,399,690 Trainable params: 2,399,690 Non-trainable params: 0
2.	Accuracy	Training Accuracy – 74.94 Validation Accuracy - 87.23000	Epoch 4/10 3000/3000 [==============================] - 184s 61ms/step - loss: 1.4170 - accuracy: 0.5937 - val_loss: 1.0779 - val_accuracy: 0.8083 Epoch 5/10 3000/3000 [==============================] - 184s 61ms/step - loss: 1.2016 - accuracy: 0.6396 - val_loss: 0.8568 - val_accuracy: 0.8286 Epoch 6/10 3000/3000 [==============================] - 183s 61ms/step - loss: 1.0582 - accuracy: 0.6772 - val_loss: 0.7206 - val_accuracy: 0.8431 Epoch 7/10 3000/3000 [==============================] - 181s 60ms/step - loss: 0.9642 - accuracy: 0.6992 - val_loss: 0.6349 - val_accuracy: 0.8517 Epoch 8/10 3000/3000 [==============================] - 184s 61ms/step - loss: 0.8933 - accuracy: 0.7219 - val_loss: 0.5756 - val_accuracy: 0.8601 Epoch 9/10 3000/3000 [==============================] - 183s 61ms/step - loss: 0.8457 - accuracy: 0.7350 - val_loss: 0.5334 - val_accuracy: 0.8670 Epoch 10/10 3000/3000 [==============================] - 182s 61ms/step - loss: 0.8030 - accuracy: 0.7494 - val_loss: 0.4991 - val_accuracy: 0.8736

10. ADVANTAGES & DISADVANTEGES

ADVANTAGES:

- It saves times for arranging and sorting huge amount of data.
- Only requires far less physical space than the storage of the physical copies.
- Recognising multiple digits on a single frame using sequential model in Keras.
- Data storage, for an example, there are many files, contracts and some personal records that contains some handwritten digits.
- It reduces human effort and labour cost.
- This can be used for sorting through mail by postal code.

DISADVANTAGES:

- The system build is complex and holds difficulty.
- The handwriting of every individual varies which proves to be a challenge for the system to predict.
- Possible unemployment of labour that is typical of technology growth.
- The accuracy is not guarantees and there are risk of errors.

11. CONCLUSION

Handwritten digit recognition has immense applications in the field of medical, banking, student management, and taxation process etc. Many classifiers like KNN, SVM, and CNN are used to identify the digit from the handwritten image. Here we've used CNN for implementation. Convolutional Neural Network gets trained from the real-time data and makes the model very simple by reducing the number of variables and gives relevant accuracy. MNIST dataset consist of handwritten numbers from 0-9 and it is a standard dataset used to find performance of classifiers.

Results of HDR is improved a lot by using CNN classifier but it can be improved further in terms of complexity, duration of execution and accuracy of results by making combination of classifiers or using some additional algorithm with it. More accurate results can be established with more convolution layers and more number of hidden neurons. It can completely abolish the need for typing. Digit recognition is an excellent prototype problem for learning about neural networks and it gives a great way to develop more advanced techniques of deep learning.

12. FUTURE SCOPE

In future, different architectures of CNN, namely, hybrid CNN, viz., CNN-RNN and CNN-HMM models, and domain-specific recognition systems, can be investigated. Evolutionary algorithms can be explored for optimizing CNN learning parameters, namely, the number of layers, learning rate and kernel sizes of convolutional filters. The future development of the applications based on algorithms of deep and machine learning is practically boundless.

In the future, we can work on a denser or hybrid algorithm than the current set of algorithms with more manifold data to achieve the solutions to many problems. In future, the application of these algorithms lies from the public to high-level authorities, as from the differentiation of the algorithms above and with future development we can attain high-level functioning applications which can be used in the classified or government agencies as well as for the common people. Currently only the digits are recognized. In future the all the characters in all the language can be predicted with high accuracy rate.

13. APPENDIX

Source Code:

The necessary libraries are imported.

```
import keras
import tensorflow
from keras.datasets import mnist
from keras.layers import Dense, Dropout,
Flatten from keras.layers import Conv2D,
MaxPooling2D from keras import backend as K
from tensorflow.keras.utils import to_categorical
import matplotlib.pyplot as plt
from tensorflow.keras.models import
Sequential from tensorflow.keras.layers import
Conv2D
from tensorflow.keras.layers import
MaxPooling2D from tensorflow.keras.layers import
Dense
from tensorflow.keras.layers import Flatten
from tensorflow.keras.optimizers import SGD
```

The MNIST dataset is downloaded from the keras library and the data is analyzed.

```
# the data, split between train and test sets
(x_train,y_train),(x_test,y_test)=mnist.load_data()
print(x_train.shape,y_train.shape)
print(x_test.shape,y_test.shape)
x_train[0]
```

The data is pre-processed and reshaped.

```
#Preprocess the
data num_classes=10
x_train=x_train.reshape(x_train.shape[0],28,28,1)
x_test=x_test.reshape(x_test.shape[0],28,28,1)
input_shape = (28,28,1)
```

Applying one-hot encoding. The class vectors are converted to binary class matrices.

```
#Convert class vectors to binary class matrices
y_train=keras.utils.to_categorical(y_train,num_classes)
y_test=keras.utils.to_categorical(y_test,num_classes)
x_train=x_train.astype('float32')
x_test=x_test.astype('float32')
x_train=x_train/255
x_test=x_test/255
```

```
print('x_train
shape:',x_train.shape)
print(x_train.shape[0],'train
samples')
```

The CNN model is created. The activation function is Rectified linear unit(ReLU). The pooling layers, dense layers are added and flattened.

```
#Create the
Model
batch_size=128
num_classes=10
epochs=20
model =
Sequential()
model.add(Conv2D(32,
kernel_size=(3,3),activation='relu',input_shape=input_shape))
model.add(Conv2D(64,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(64,activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes,activation='softmax'))
```

The model is then compiled.

```
model.compile(loss=keras.losses.categorical_crossentropy,
optimizer=keras.optimizers.
Adadelta(),metrics=['accuracy'])
```

The model is trained.

```
hist = model.fit(x_train,
y_train,batch_size=20,epochs=5,verbose=1,validation_data=(x_test, y_test))
```

Observing the metrics and testing the model.

```
metrics = model.evaluate(x_test, y_test,
verbose=0) print("Metrics(Loss and Accuracy):")
print(metrics)
prediction =
model.predict(x_test[:4])
print(prediction)
```

The model is saved and then tested. A sample image is given in to test the saved model. The image is reshaped and then predicted.

```
model.save('digit_classifier.h5')
from keras.utils.image_utils import
img_to_array from tensorflow.keras.models import
load_model

model =
load_model('/content/digit_classifier.h5') from
PIL import Image
import numpy as np

img =
Image.open('/content/sample.png').convert("L")
img = img.resize((28,28))
im2arr = np.array(img)
im2arr = im2arr.reshape(1,28,28,1)

#display the image
import matplotlib.pyplot as
plt plt.imshow(img)

#predict the image
y_predict = model.predict(im2arr)
print(np.argmax(y_predict[0]))
```

The pages to display the home and recognise page with navigation bar.

HDR front end.html

```html
<html>
    <head>
        <style>
            body {
              background-image: url('https://cdn.pixabay.com/photo/2020/09/23/03/54/background-5594879_1280.jpg');
              margin: 0;
              font-family: 'Times New Roman', Times, serif, Helvetica, sans-serif;
            }

            .topnav {
              overflow: hidden;
              background-color: rgb(255, 255, 255);
            }

            .topnav a {
              float: left;
              color: #480557;
              text-align: center;
              padding: 14px 16px;
              text-decoration: none;
              font-size: 17px;
            }

            .topnav a:hover {
              background-color: rgb(57, 55, 55);
              color: rgb(250, 248, 248);
            }

            .topnav a.active {
              background-color: #f8e406;
              color: rgb(19, 19, 19);
            }
            p{
                text-align: center;
                background-color: rgb(8, 0, 0);
                margin-left: 25%;
                margin-right: 25%;
                margin-top: 5%;
                font-family:'Times New Roman', Times, serif;
                color:aliceblue;
                font-size: large;
            }
            </style>
```

```html
    </head>
    <body>

        <div class="topnav">
            <a class="active" href="#home">Home</a>
            <a href="Recognize.html">Recognise</a>
        </div>
        <p style="font-size:larger">Handwritten Digit Recognition</p>
        <p>Handwriting recognition is one of the compelling research works going on because every
            individual in this world has their own style of writing. It is the capability of the computer to identify and understand handwritten digits or characters
             automatically. Because of the progress in the field of science and technology, everything is being digitalized to reduce human effort. Hence, there comes
             a need for handwritten digit recognition in many real-time applications. MNIST data set is widely used for this recognition process and it has 70000
             handwritten digits. We use Artificial neural networks to train these images and build a deep learning model. Web application is created where the user
             can upload an image of a handwritten digit. this image is analyzed by the model and the detected result is returned on to UI</p>
    </body>
</html>
```

The recognise page where the user can upload the image for prediction

Recognise.html

```html
<html>
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge" />
        <title>Digit Recognition</title>
        <style>
            body {
              background-image: url('https://img.freepik.com/premium-vector/falling-colorful-messy-numbers-math-study-concept-with-flying-digits-fascinating-back-school-mathematics-banner-white-background-falling-numbers-vector-illustration_174187-2929.jpg?w=2000');
              margin: 0;
              font-family: 'Times New Roman', Times, serif, Helvetica, sans-serif;
              height: 100%;
              width: 100%;
            }
            h1 {
                display: block;
                font-size: 3.5em;
                margin-top: 5.4em;
                margin-bottom: 0em;
                margin-left: 50%;
                margin-right: 0;
                font-weight: bold;
            }


            .button {
                border:#e5b9f3;
                color: rgb(56, 1, 69);
                padding: 15px 32px;
                text-align: center;
                text-decoration: none;
                display: inline-block;
                font-size: 16px;
                margin: 4px 2px;
                cursor: pointer;
                }

            .button1 {
                background-color: #b6e6f0;
                margin-top: 5.4em;
                margin-left: 56%;
                margin-right: 0;
                border: black;
            }
            .button2 {
                background-color: #b6e6f0;
                margin-top: 5.5em;
                margin-left: 55%;
                margin-right: 0;
                border: black;
            }
        </style>
        <script>
```

```html
            function view() {
                frame.src=URL.createObjectURL(event.target.files[0]);
            }

            $('#submit').click(function(){
                $.ajax({
                    url: 'app.py',
                    type: 'POST',


                })
            })

            var input = document.getElementById('image');
            var infoArea = document.getElementById('frame');

            input.addEventListener('change', showFileName);

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

        </script>
        <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>
        <script src="https://ajax.googleis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>

    </head>
    <body>
        <h1 style = "color: rgb(2, 60, 0)">Digit Recognition
        <br>
        <br>
        <form action="{{url_for('predict')}}" method="POST" enctype="multipart/form-data">

        <input id="image" type="file" name="image" accept="image/png, image/jpeg" onchange="view()"><br><br>
        <img id="frame" type="submit" src="" width="100px" height="100px"/>

<a href = "Predict.html">Detect</a>
        </h1>
        </form>


    </body>
</html>
```

The page where the predicted output is displayed

Predict.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <title>Prediction</title>
</head>

<style>
    body{
     background-image: url('https://img.freepik.com/premium-vector/falling-colorful-messy-numbers-math-study-concept-with-flying-digits-fascinating-back-school-mathematics-banner-white-background-falling-numbers-vector-illustration_174187-2929.jpg?w=2000');
     background-repeat: no-repeat;
     height: 100%;
     width: 100%
    }
    .button{
        border:#e5b9f3;
        color: rgb(56, 1, 69);
        padding: 15px 32px;
        text-align: center;
        text-decoration: none;
        display: inline-block;
        font-size: 16px;
        margin-left:45%;
        cursor: pointer;

    }


    #pred{
        text-align: center;
        font-family: 'Times New Roman', Times, serif;
        font-size: 40px;
        margin: 0 auto;
        padding: 3% 5%;
        padding-top: 15%;
        color: rgb(0, 10, 80);
    }
</style>

<body>

        <p id="pred">PREDICTION : {{ num }}</p>
        <p>
        <a href="Recognize.html">
            <button data-inline="True" class="button">Back</button>
        </a>



        </p>
</body>
</html>
```

The flask app.py python code to calculate the prediction value from processing the image uploaded by the user.

app.py

```
import os
import numpy as np
from flask import Flask, render_template,
request import tensorflow as tf
from PIL import Image
from werkzeug.utils import secure_filename
UPLOAD_FOLDER =
'C:\Users\AKSHAYA\Pictures\static\images' app = Flask(
name_)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

@app.route('
/') def
index():
        return render_template('recognise.html')

model = tf.keras.models.load_model("digit_classifier.h5")
@app.route('/predict', methods = ['GET','POST'])
def upload_image_file():
            if request.method ==
        'POST': imagefile =
        request.files['image']
                filename = secure_filename(imagefile.filename)
        imagefile.save(os.path.join(app.config['UPLOAD_FOLDER'],
        filename)) path_img = os.path.join(UPLOAD_FOLDER, filename)
                img =
        Image.open(path_img).convert("L")
        img = img.resize((28,28))
                im2arr = np.array(img)
                im2arr =
        im2arr.reshape(1,28,28,1)
        y_pred = model.predict(im2arr)
        return render_template('predict.html', num =

str(y_pred)) if_name_== '_main_' :

        app.run(host='0.0.0.0', port=8000, debug=True)
```

Github:

https://github.com/IBM-EPBL/IBM-Project-17799-1659676458

Project Demo Link:

https://drive.google.com/file/d/1AUrTi02zoFrwP8cP6qWhtFJrAKyaHBah/view?usp=share_link