PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION

submitted by

PNT2022TMID18443

Nithiyashree M G

Rizwana S

Shaik Sharmila

Vijayadharshini

TABLE OF CONTENTS

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

    7.1 Feature 1

    7.2 Feature 2

    7.3 Database Schema (if Applicable)

8. TESTING

    8.1 Test Cases

    8.2 User Acceptance Testing

9. RESULTS

    9.1 Performance Metrics

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

    Source Code, GitHub & Project Demo Link

1.INTRODUCTION

1.1 PROJECT OVERVIEW

Machine learning and deep learning play an important role in computer technology and artificial intelligence. With the use of deep learning and machine learning, human effort can be reduced in recognizing, learning, predictions and in many more areas.

Handwritten Digit Recognition is the ability of computer systems to recognise handwritten digits from various sources, such as images, documents, and so on. This project aims to let users take advantage of machine learning to reduce manual tasks in recognizing digits.

1.2 PURPOSE

Digit recognition systems are capable of recognizing the digits from different sources like emails, bank cheque, papers, images, etc. and in different real-world scenarios for online handwriting recognition on computer tablets or system, recognize number plates of vehicles, processing bank cheque amounts, numeric entries in forms filled up by hand (tax forms) and so on.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

The fundamental problem with handwritten digit recognition is that handwritten digits do not always have the same size, width, orientation, and margins since they vary from person to person. Additionally, there would be issues with identifying the numbers because of similarities between numerals like 1 and 7, 5 and6, 3 and 8, 2 and 5, 2 and 7, etc. Finally, the individuality and variation of each individual's handwriting influence the structure and appearance of the digits.

2.2 REFERENCES

Pre-processing techniques involved in the character recognition(2013)

K. Gaurav, Bhatia P. K

This paper deals with the Various pre-processing techniques involved in the character Recognition with different kind of images ranges from a Simple handwritten form based documents and documents Containing colored and complex background and varied Intensities. In this, different preprocessing techniques like Skew detection and correction, image enhancement techniques of contrast stretching, binarization, noise removal techniques, Normalization and segmentation, morphological processing Techniques are discussed. It was concluded that using a single Technique for preprocessing, we can't completely process the Image. However, even after applying all the said techniques Might not possible to achieve the full accuracy in a Preprocessing system

Improving Offline Handwritten Text Recognition With Hybrid HMM/ANN Model(2011)

Salvador Espana-Boquera

In this paper hybrid Hidden Markov Model (HMM) model is proposed for

recognizing unconstrained offline handwritten texts. In this, the structural part of the optical model has been modelled with Markov chains, and a Multilayer Perceptron is used to estimate the emission probabilities. In this paper, different techniques are applied to remove slope and slant from handwritten text and to normalize the size of text images with supervised learning methods. The key features of this recognition system were to develop a system having high accuracyin preprocessing and recognition, which are both based on anns.

Optimizing Feature Selection For Recognizing Handwritten Arabic Characters(2005)

Mohammed Z. Khedher, Gheith A. Abandah, and Ahmed M. Al Khawaldeh

This paper describes that Recognition of characters greatly depends upon the features used. Several features of the handwritten Arabic characters are selectedand discussed. An off-line recognition system based on the selected features wasbuilt. The system was trained and tested with realistic samples of handwritten Arabic characters. Evaluation of the importance and accuracy of the selected features is made. The recognition based on the selected features give average accuracies of 88% and 70% for the numbers and letters, respectively. Further improvements are achieved by using feature weights based on insights gained from the accuracies of individual features.

Fuzzy-Zoning-Based Classification For Handwritten Characters(2011)

G. Pirlo and D. Impedovo

Presented a new class of membership functions, which are called Fuzzymembership functions (fmfs), for zoning-based classification. These fmfs canbe easily adapted to the specific characteristics of a classification problem in order to maximize classification performance. In this research, a realcoded

genetic algorithm is presented to find, in a single optimization procedure, the optimal FMF, together with the optimal zoning described by Voronoi tessellation. The experimental results, which are carried out in the field of

handwritten digit and character recognition, indicate that optimal FMF performs better than other membership functions based on abstract level, ranked-level, and measurement- level weighting models, which can be found in the literature.

Feature Selection Using Genetic Algorithm(2018)

Yoshimasa Kimura

Presented a work on how to select features for Character Recognition Using Genetic Algorithm. The author proposes a novel method of feature selection for character recognition using genetic algorithms (GA). The proposed method selectsonly the genes for which the recognition rate of training samples exceeds than the predetermined threshold as a candidate of the parent gene and adopts a reduction ratio in the number of features used for recognition as the fitness value.

Improved Handwritten Digit Recognition Using Convolutional Neural Networks(CNN) (2020)

Ahlawat, Savita and Choudhary, Amit and Nayyar, Anand and Singh, Saurabh and Yoon, Byungun

This paper's primary goal was to enhance handwritten digit recognition ability. To avoid difficult pre-processing, expensive feature extraction, and a complex ensemble (classifier combination) method of a standard recognition system, they examined different convolutional neural network variations. Their current work makes suggestions on the function of several hyper-parameters through thorough evaluation utilizing an MNIST dataset. They also confirmed that optimizinghyper-parameters is crucial for enhancing CNN architecture performance. With the Adam optimizer for the MNIST database, they were able to surpass many previouslypublished results with a recognition rate of 99.89%. Through the trials, it is made abundantly evident how the performance of handwritten digit recognition is affected by the number of convolutional layers in CNN architecture. According to the paper, evolutionary algorithms can be explored for optimizing convolutional filterkernel sizes, CNN learning parameters, and the quantity of layers and learning rates.

An Efficient And Improved Scheme For Handwritten Digit Recognition Based OnConvolutional Neural Network (2019)

Ali, Saqib and Shaukat, Zeeshan and Azeem, Muhammad and Sakhawat, Zareen and Mahmood, Tariq and others

This study uses rectified linear units (ReLU) activation and a convolutional neural network (CNN) that incorporates the Deeplearning4j (DL4J) architecture to recognize handwritten digits. The proposed CNN framework has all the necessary parameters for a high level of MNIST digit classification accuracy. The system's training takes into account the time factor as well. The system is also tested by altering the number of CNN layers for additional accuracy verification. It is importantto note that the CNN architecture consists of two convolutional layers, the first with 32 filters and a 5x5 window size and the second with 64 filters and a 7x7 window size. In comparison to earlier proposed systems, the experimental findings show thatthe proposed CNN architecture for the MNIST dataset demonstrates great performance in terms of time and accuracy. As a result, handwritten numbers are detected with a recognition rate of 99.89% and high precision (99.21%) in a short amount of time.

Improved Handwritten Digit Recognition Using Quantum K-Nearest NeighborAlgorithm (2019)

Wang, Yuxiang and Wang, Ruijin and Li, Dongfen and Adu-Gyamfi, Daniel and Tian, Kaibin and Zhu, Yixin

The KNN classical machine learning technique is used in this research to enable quantum parallel computing and superposition. They used the KNN algorithm with quantum acceleration to enhance handwritten digit recognition. When dealing with more complicated and sizable handwritten digital data sets, their suggested methodconsiderably lowered the computational time complexity of the traditional KNN algorithm. The paper offered a theoretical investigation of how quantum concepts can be applied to machine learning. Finally, they established a fundamental operational concept and procedure for machine learning with quantum acceleration.

Handwritten Digit Recognition Using Machine And Deep Learning Algorithm(2021)
Pashine, Samay and Dixit, Ritik and Kushwah, Rishika

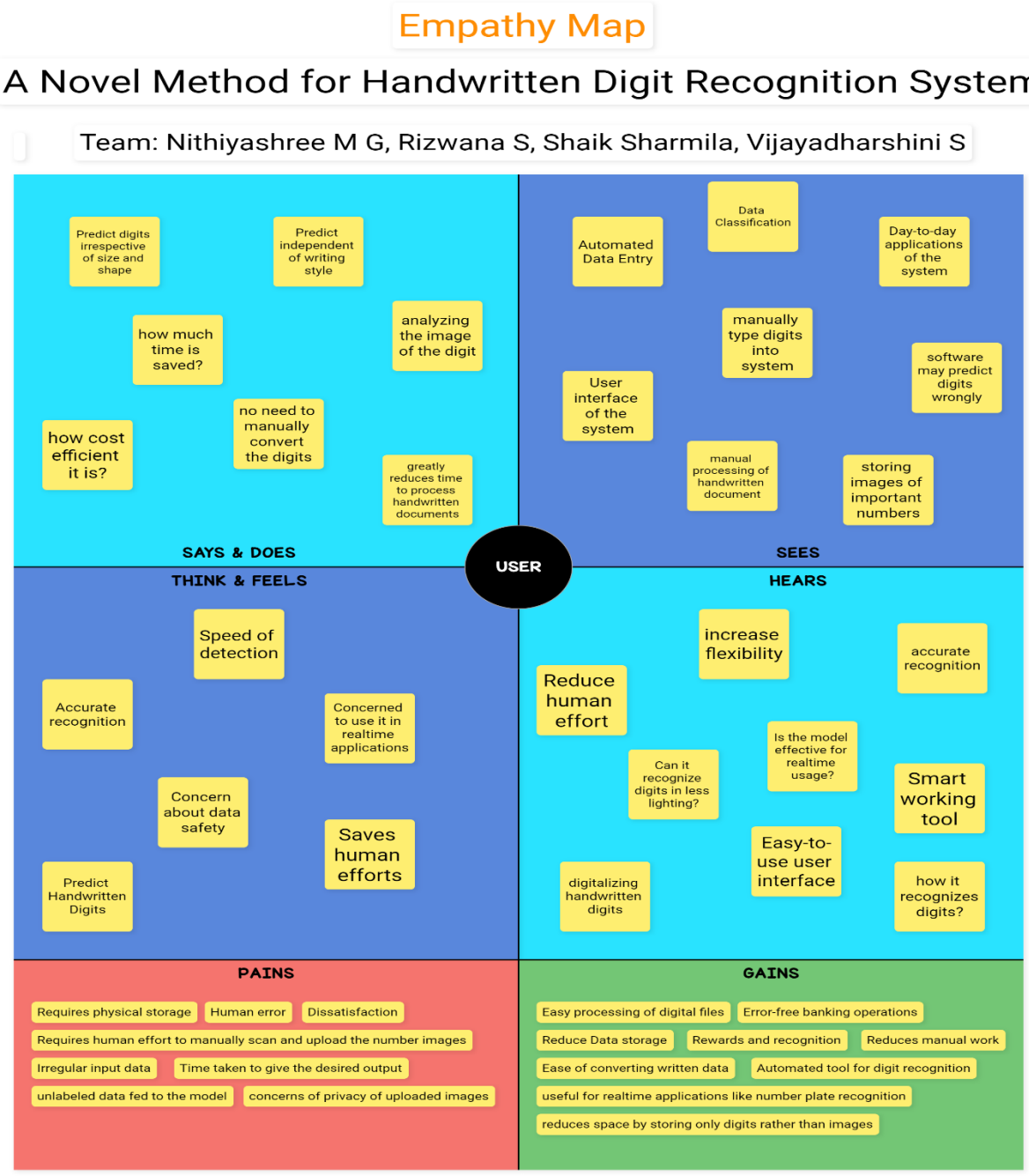
Support vector machines are among the simplest classifiers, making them faster than other algorithms and providing the highest training accuracy rate in this situation. However, due to their simplicity, SVMs cannot categorize complicated andambiguous images as accurately as MLP and CNN algorithms can. In their research,they discovered that CNN produced the most precise outcomes for handwritten digit recognition. This led them to the conclusion that CNN is the most effective solution for all types of prediction issues, including those using picture data. Next, by comparing the execution times of the algorithms, they determined that increasing the number of epochs without changing the configuration of the algorithm is pointless due to the limitation of a certain model, and they discoveredthat beyond a certain number of epochs, the model begins over-fitting the datasetand provides biased predictions.

2.1 PROBLEM STATEMENT DEFINITION

Handwriting recognition is one of the compelling research works going on because every individual inthis 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.

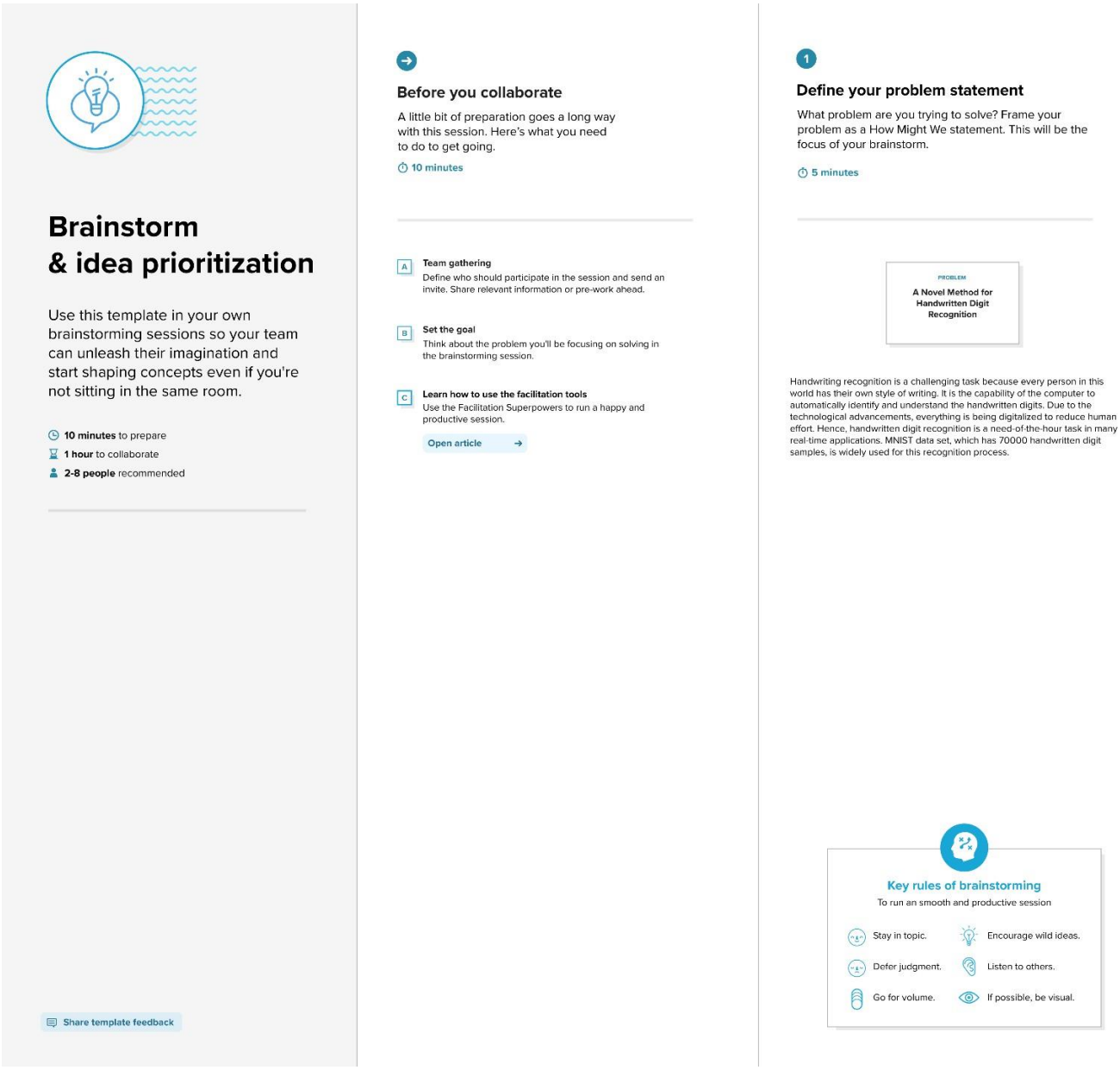
3.IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

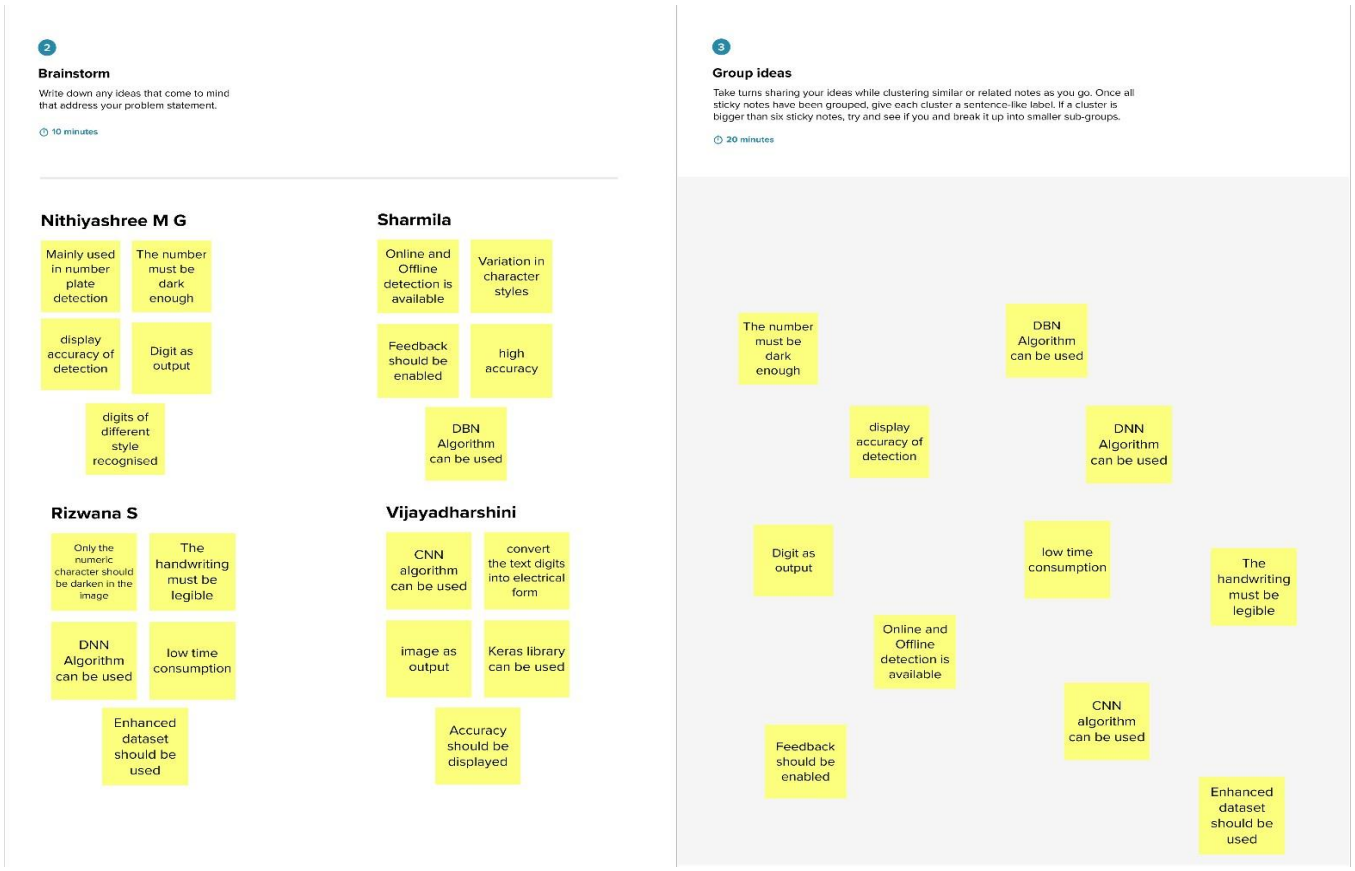


3.2 Ideation& Brainstorming

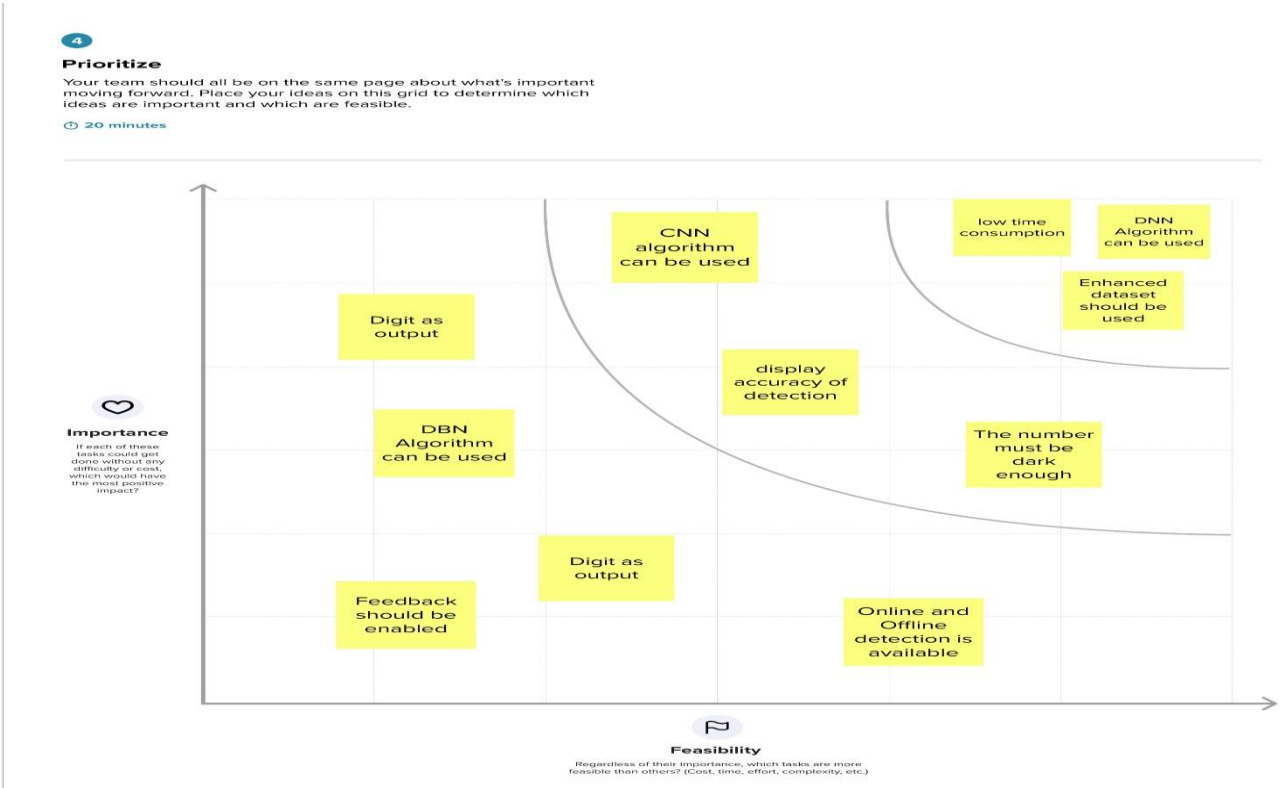
Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization



3.3PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	• It is easy for the human to perform task accurately by practicing it repeatedly and memorizing it for the next time. Human brain can process and analyse images easily. Also, recognize the different element present in the images. • The **handwritten digit recognition** is the capability of computer applications to **recognize** the human **handwritten digits**.
2.	Idea / Solution description	• The algorithm used is Convolution Neural Network(CNN). This will prepare the trained model which will be used to classify the digits present in the test data. Thus, we can classify the digits present in the images as: Class 0,1,2,3,4,5,6,7,8,9. • MNIST is a dataset which is widely used for handwritten digit recognition. The dataset consist of 60,000 training images and 10,000 test images.
3.	Novelty / Uniqueness	• This project introduces an operative strategy for dealing with novelty in the handwritten visual recognition domain. A perfect transcription agent would be able to distinguish known and unknown characters in a picture, as well as determine any aesthetic variations that may

		occur inside or between texts. The existence of novelty has shown to be a major stumbling block for even the most robust machine learning-based algorithms for these activities.
4.	Social Impact / Customer Satisfaction	• There are many benefits associated with the handwriting recognition system. In addition to reading postal addresses and bank check amounts, it is also useful for reading forms. Furthermore, it's used in fraud detection because it makes it easy to compare two texts and determine which one is a copy.
5.	Business Model (Revenue Model)	• The applications where these handwritten digit recognition can be used are Banking sector where it can be used to maintain the security pin numbers, it can be also used for blind peoples by using sound output. • Some of the research areas include signature verification, bank check processing, postal address interpretation from envelopes etc.
6.	Scalability of the Solution	• One of the approaches to make the handwritten digit recognition system scalable is to make use of cloud-native methods. For example, one of the cloud solutions for making AI scalable is IBM Cloud. IBM Cloud Build helps run and manage AI models, optimize decisions at scale across any cloud.

3.4 PROBLEM SOLUTION FIT

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

- Person who are at industry side for recognizing various handwriting digits.
- People working in bank, post offices

6. CUSTOMER CONSTRAINTS CC

- Time
- Accuracy
- Ease to access
- Imperfect findings

5. AVAILABLE SOLUTIONS AS

- In past they get trouble in finding handwritten digits
- Using this system, they can resolve this type of problems
- Pros of this system is quick recognition and
- Accurate prediction
- Cons are network connection is mandatory for using this system
- For using this system Knowledge about the system is required

Explore AS, differentiate

Focus on J&P, tap into BE, understand

Identify strong TR & EM

2. JOBS-TO-BE-DONE / PROBLEMS J&P

- There are different types of handwriting are in world.
- Each and every handwriting has its own characteristics and uniqueness. Its difficult to understand the different people's handwriting digit.

9. PROBLEM ROOT CAUSE RC

- Not everyone can understand everyone's handwriting
- The handwriting is differed from person to person
- So, it is difficult to recognize the digits
- To solve this problem this system has developed

7. BEHAVIOUR BE

To address the problem, they can take a snap of the handwritten digit and upload it in the software

Focus on J&P, tap into BE, understand

Identify strong TR & EM

3. TRIGGERS

- By word of mouth
- Good user experience

4. EMOTIONS: BEFORE / AFTER

- It is a quite irritating and frustrating whilemanually convert the handwrittendigits
- • By using our system, user can save thetimeand reduce the error occur on recognition

10. YOUR SOLUTION

- A novel method for handwritten digit recognition system helps in recognizing the handwritten digitsthatuses MNIST dataset for training the model.
- The model gets the image of the handwritten digits and recognizesthe handwritten digits.
- CNN algorithm is used over the MNIST dataset to recognize the handwritten digits.

8. CHANNELS of BEHAVIOUR

8.1 ONLINE

In online they can upload the handwrittenpicture and yield output

8.2 OFFLINE

In offline they can ask their neighborsto scribble the digits to find them

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	Getting the handwritten digit input	The handwritten digit is obtained as input from the user as an image uploading or writing on the canvas.
FR-2	Data preprocessing	Upgrades the image to make it ready for segmentation, by performing some tasks on the input image.
FR-3	Segmentation & Feature Extraction	Segment the MNIST dataset images using edge detection technique and remove redundancy from the data
FR-4	Classification and Recognition	Passing the feature vectors as individual input to the classifiers or neural networks such as CNN.
FR-5	Prediction	The deep learning model is trained and tested using the MNIST dataset, with accuracy $>$ 90%
FR-6	Evaluation	Ensure that the digit is correctly recognised by the model and produces accurate output.

4.2 Non-functional Requirements:

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

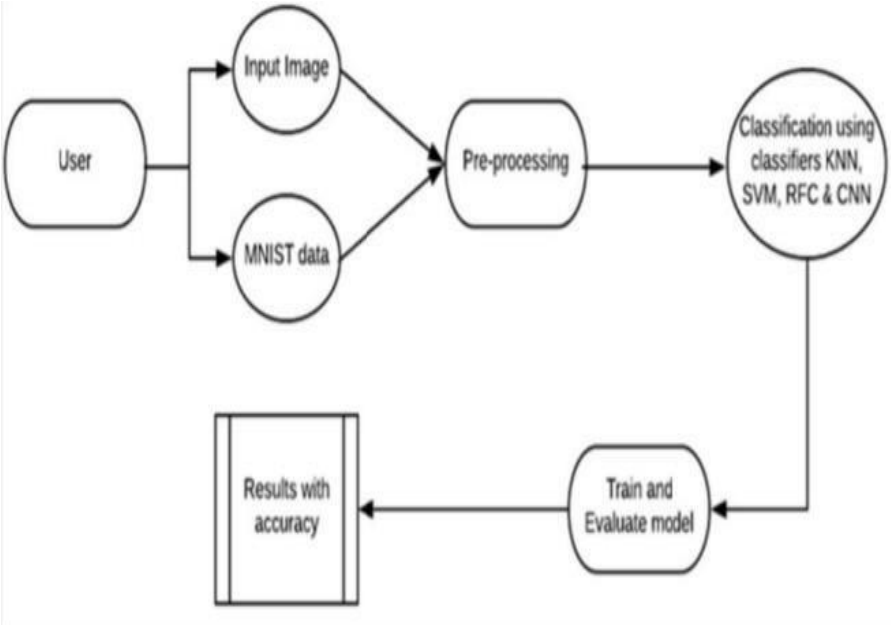
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	To identify and understand handwritten digits automatically, with high accuracy.
NFR-2	Security	Ensures security, since uploaded images are not stored in any database
NFR-3	Reliability	User-friendly web interface for the system. Process confidential information without data leakage.
NFR-4	Performance	High, since artificial neural networks are used to train the images and build deep learning model. Fast prediction using CNN algorithm.
NFR-5	Availability	Using web application, anyone can easily access the system, making it highly available for web and mobile browsers.
NFR-6	Scalability	Performs well even if the count of input handwriting increased, since MNIST dataset is used for recognition process. Low time consumption.

5.PROJECT DESIGN

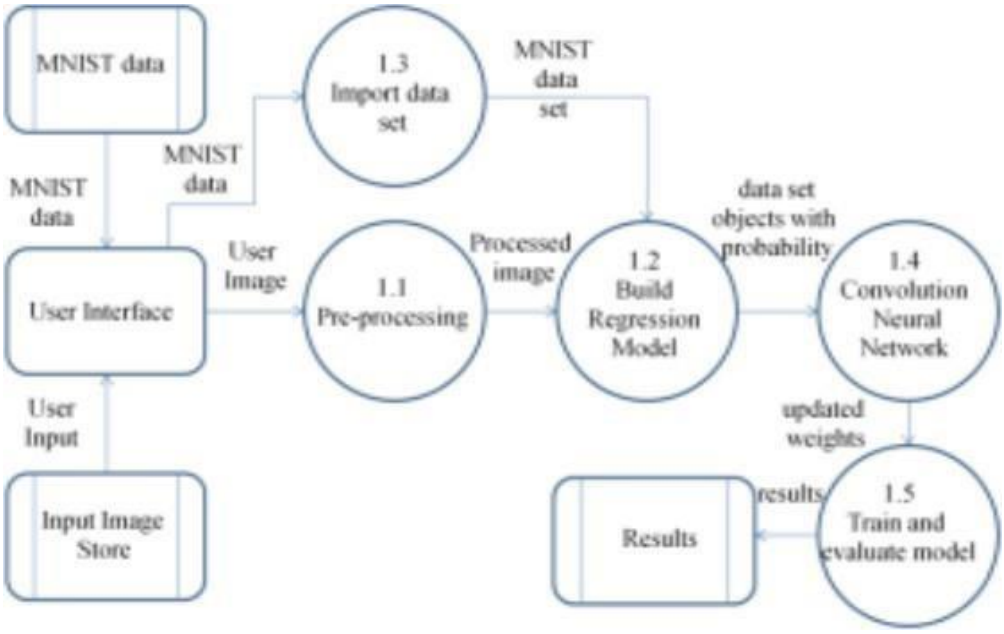
5.1 DATA FLOW DIAGRAM

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

Example:(Simplified)FLOW



Example: DFD Level 0 (Industry Standard)



5.2 SOLUTION & TECHNICAL ARCHITECTURE

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered

Solution Architecture Diagram:

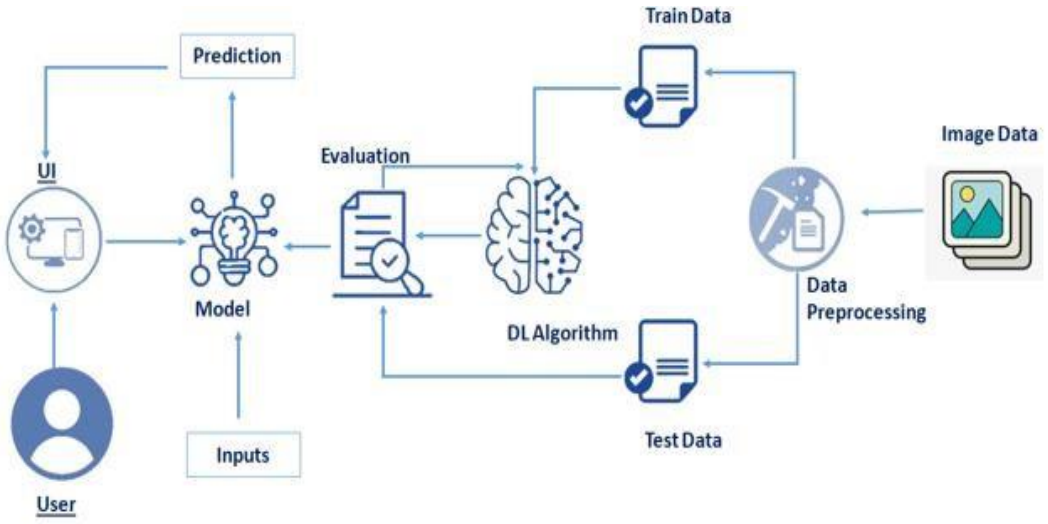


Figure 1: Architecture of the Handwritten digit recognition system

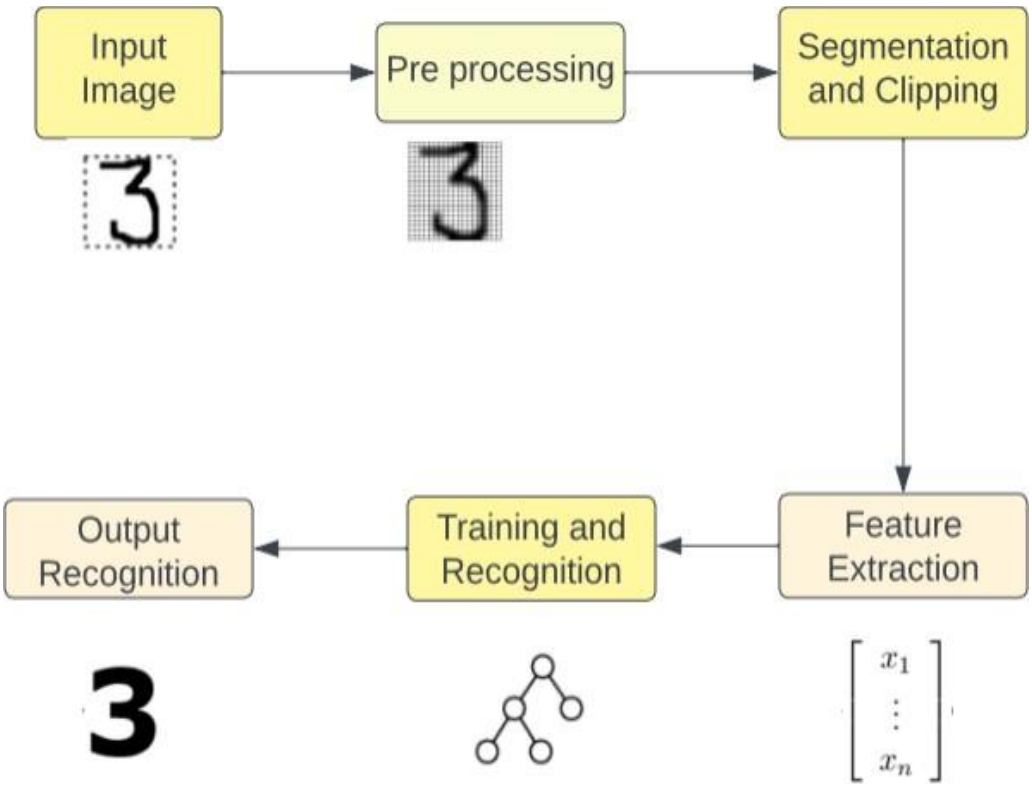


Figure 2: Model Architecture of the Deep Learning model used for Handwritten digit recognition.

5.3USER STORIES

Functional Requirement (Epic)	User Story Number	User Story / Task
Data Pre-processing	USN-1	Collect and load the dataset, scaling and wrangling the data and split into train and test data.
Model Building	USN-2	Build a Deep Learning Model with CNN to recognize the handwritten digit with higher accuracy.
	USN-3	Training and testing the model with the split image dataset and refine the accuracy.
	USN-4	Model saved and used for further integration with the Flask web user interface.
UI Application	USN-5	Building a flask web application interface to upload the handwritten image by clicking the upload button
	USN-6	Integrating the flask web application with the saved deep learning model
	USN-7	As a user, I can see the predicted digits with the accuracy, in the web application
Train the model on IBM	USN-8	Training the model on IBM Cloud and deploy the Flask web application with scoring end point.

6.PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Team Members
Sprint-1	Data Pre-processing	USN-1	Collect and load the dataset, scaling and wrangling the data and split into train and test data.	Nithiyashree M G, Rizwana S
Sprint-2	Model Building	USN-2	Build a Deep Learning Model with CNN to recognize the handwritten digit with higher accuracy.	Nithiyashree M G, Sharmila S
Sprint-2		USN-3	Training and testing the model with the split image dataset and refine the accuracy.	Rizwana S, Vijayadharshini S
Sprint-2		USN-4	Model saved and used for further integration with the Flask web user interface.	Sharmila S, Vijayadharshini S
Sprint-3	UI Application	USN-5	Building a flask web application interface to upload the handwritten image by clicking the upload button	Rizwana S, Vijayadharshini S
Sprint-3		USN-6	Integrating the flask web application with the saved deep learning model	Nithiyashree M G, Sharmila S
Sprint-3		USN-7	As a user, I can see the predicted digits with the accuracy, in the web application	Sharmila S, Vijayadharshini S
Sprint-4	Train the model on IBM	USN-8	Training the model on IBM Cloud and deploy the Flask web application with scoring end point.	Nithiyashree M G, Rizwana S

6.2 SPRINT DELIVERY SCHEDULE

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

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).

$$\text{Average velocity} = \frac{sprint\ duration}{velocity} = \frac{20}{6} = 3.3$$

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 suchas Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

https://www.visual-paradigm.com/scrum/scrum-burndown-chart/

7.CODING & SOLUTIONING

```
import os

import numpy as np

from flask import Flask, render_template, request, send_from_directory,
url_for

#from gevent.pywsgi import WSGIServer

from keras.models import load_model

from keras.preprocessing import image

from PIL import Image

from werkzeug.utils import redirect, secure_filename


UPLOAD_FOLDER = 'D:/NalaiyaThiran/projFiles/data'


app = Flask(__name__)

app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER


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


@app.route('/')

def index():

    return render_template('index.html')
```

```
@app.route('/web', methods=['GET', 'POST'])
def web():
    if request.method == "POST":
        f = request.files["image"]
        basepath = os.path.dirname(__file__)
        filepath = os.path.join(basepath, 'data', f.filename)
        f.save(filepath)
        # img = image.load_img(filepath, target_size=(64, 64))
        # x = image.img_to_array(img)
        # x = np.expand_dims(x, axis=0)

        # 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(filepath).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('web.html', num=str(num[0]))
    return render_template('web.html')


if __name__ == '__main__':
    app.run(debug=True, threaded=False)
```

8. TESTING

8.1 TEST CASE

Test case ID	Feature Type	Component	Test Scenario
Homepage_TC_OO1	Functional	Home Page	Verify user is able to access the homepage
HomePage_TC_OO2	UI	home Page	Verify the UI elements in home page
PredictPage_TC_OO3	Functional	predict page	Verify user is able to upload the image
PredictPage_TC_OO4	Functional	predict page	Verify user is able to preview the uploaded image
PredictPage_TC_OO5	Functional	predict page	Verify user is able to view the predicted result

Pre-Requisite	Steps To Execute	Test Data
	1.Enter URL and click go 2. Verify homepage displayed or not	http://127.0.0.1:5000/
	1.Enter URL and click go 2. Verify home page with below UIelements: a.home link in navigation b.recognize link in navigation	
	1.Enter URL and click go 2.Click on Recognize button 3.Click the choose button 4.Choose an image file with valid format 5.Click on predict button	2.png
	1.Enter URL and click go 2.Click on Recognize button 3.Click the choose button 4.Choose an image file with valid format 5.Click on predict button	2.png
	1.Enter URL and click go 2.Click on Recognize button 3.Click the choose button 4.Choose an image file with valid format 5.Click on predict button	2.png

Expected Result	Actual Result	Status	TC for Automation(Y/N)	BUG ID	Executed By
homepage should be displayed	Working as expected	Pass	N		Vijayadharshini S, Rizwana S
Application should show below UI elements: a.home link in navigation b.recognize link in navigation	Working as expected	Pass	N		Rizwana S, Sharmila S
choose file popup should be displayed	Working as expected	Pass	N		Sharmila S, Nithiyashree M G
image should be displayed on the preview frame	Working as expected	Pass	N		Nithiyashree M G, Vijayadharshini S
User should be able to view the predicted result	Working asexpected	Pass	N		Rizwana S, Sharmila S

8.2 USER ACCEPTANCE TESTING

1. Defect Analysis

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

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

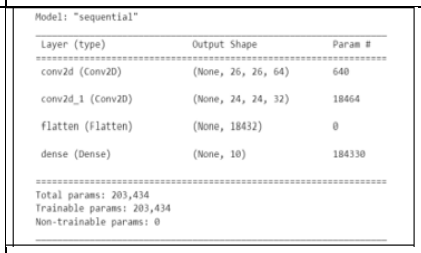
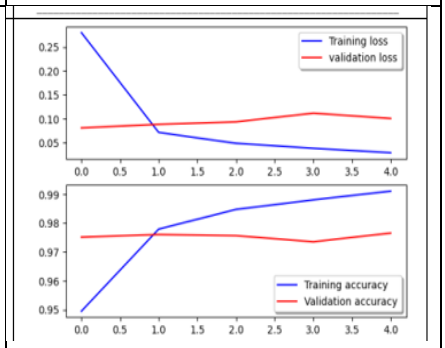
2. Test Case Analysis

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

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

9. RESULTS

9.1 PERFORMANCE METRICS

S.No.	Parameter	Values	Screenshot
1.	Model Summary	–	
2.	Accuracy	Training Accuracy – 99% Validation Accuracy – 97%	

10. ADVANTAGES & DISADVANTAGES

- ADVANTAGE

  - Information of images can be readable with high degree of accuracy. Flatbed scanners are very accurate and may produce reasonably top quality results.
  - Processing of DL information is fast. Large quantities of text are often input quickly.
  - It takes less time to convert within the Digital form Advanced version can even Recreate.

❖ DISADVANTAGE

- Not 100% accurate, there are likely to be some mistakes made during the method.

11. CONCLUSION

- The proposed handwritten recognition system recognizes the letters with acceptable accuracy. Extraction of small and capital letters is done separately.
- In future the small letters and capital letters can be integrated. There is a Scope for Recognition of digits and Special symbols. The information of the document can be edited more conveniently and can reuse the edited information as and when required.
- The grid infrastructure used in the implementation of Optical Character Recognition system can be efficiently used to speed up the translation of image based documents into structure documents that are currently easy to discover, search and process.
- There is no out of box software for this kind of documents which can read handwriting automatically without human validation and training.

12. FUTURE SCOPE

- Font Independent OCR system could be developed by considering the multiple font style in use. Our approach is very much useful for the font independent case. Because, for font or character size, it finds the string and the strings are parsed to recognize the character. Once character is identified, the corresponding character could

be ejected through an efficient editor. Efforts have been taken

- To develop a compatible editor for Tamil and English. There is heavy demand for an OCR system which recognizes cursive scripts and manuscripts like Palm Leaves. This actually avoids keyboard typing.
- The most required application today is Speech recognition. The recognized Printed or Handwritten character could be recorded and through a voice synthesizer speech output could be generated. This would help the blind to send and receive information.

13. APPENDIX

SOURCE CODE

Index.html

```
<html>


<head>
  <title>Handwritten Digit Recognizer</title>


  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
   initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
  <link rel="stylesheet" type="text/css"
    href="/static/css/style.css">


</head>


<body>
  <div class="main-container">
    <div class="header">
      <h1 class="welcome">Handwritten Digit
    Recognizer</h1>
      <div class="navbar">
        <ul>
          <li><a href="{{ url_for('index') }}">Home</a></li>
          <!-- <li><a href="#about">About</a></li> -->
          <li><a href="{{ url_for('web') }}">Recognize</a></li>
        </ul>
      </div>
    </div>
    <div class="body-container">
      <div class="background-image">
        <img class="image" src="/static/images/bgbg.jfif">
```

```
        <div class="content">
          <h1 class="title">About the Project</h1>
          <div class="details">
            <p>
              Handwritten Digit Recognition is the need-of-the-hour technology in the current scenario. This Digit Recognition System is used to identify the digits from different sources like invoices, posts, email and so on. Since people relied on entering the digit from the sources manually, this technology was required to be properly implemented. This could help in reducing the manual errors and the difficulty in efficiently storing and accessing physical data. This project performs the recognition of handwritten digits [0-9] from the MNIST dataset, using a Convolutional Neural Network.
            </p>
          </div>
        </div>
      </div>
    </div>
  </body>


</html>
```

Web.html:

```
<head>
  <title>Handwritten Digit Recognizer</title>
<meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">

  <link rel="stylesheet" type="text/css" href="/static/css/style.css">

</head>

<body>
  <div class="main-container">
    <div class="header">
      <h1 class="welcome">Handwritten Digit Recognizer</h1>
      <div class="navbar">
        <ul>
          <li><a href="{{ url_for('index') }}">Home</a></li>
          <!-- <li><a href="#about">About</a></li> -->
```

```
            <li><a href="{{ url_for('web') }}">Recognize</a></li>
        </ul>
    </div>
</div>
<div class="body-container">
    <div class="background-image">
        <img class="image" src="/static/images/bgbg.jfif">
        <div class="content">
            <form action="/web" method="POST"
enctype="multipart/form-data">
                <div class="imgfile">
                    <label>Select an Image:</label>
                    <input id="image" type="file" name="image"
accept="image/png, image/jpeg"
                        onchange="preview()">
                </div>
                <div class="image-frame">
                    <img id="frame" class="img" width="100px"
height="100px" />
                </div>
                <div class="buttons">
```

```
                    <button type="submit" id="predict_button"
class="btn">Predict</button>
                    <button type="button" id="clear_button"
class="btn" onclick="clearImage()">  Clear
                         </button>
                </div>
                <div>
                    <h2>The number is: {{ num }} </h2>
                </div>
            </form>
        </div>
    </div>
  </div>
</body>

<script>
  function preview() {

document.getElementById("frame").src=URL.createObjectURL(event.tar
get.files[0]);
  }
```

```
  function clearImage() {
      document.getElementById("frame").src = "";
      document.getElementById("image").value = "";}
</script>
</html>
```

Css:

```
* {
  border: 0;
 margin: 0;
}
.main-container {
  width: 100%;
  float: center;
}
.header {
  width: 100%;
  height: 80px;
  float: left;
  background-color: burlywood;
```

```
  border-bottom: 5px solid grey;
}
.welcome {
  padding: 30px;
  float: left;
  color: black;
  font-size: 18px;
  font-family: Impact, Haettenschweiler, 'Arial Narrow Bold',
    sans-serif;
}
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
}
.navbar {
  padding: 30px;
  float: right;
}
li {
  display: inline;
```

```
    padding: 20px;
    margin: 0px 10px;
    color: brown;
  }
  li a {
    display: inline;
    text-align: center;
    font-size: 18px;
    padding: 14px 16px;
    text-decoration: none;
    color: aquamarine;
  }
  li a:hover {
    color: rebeccapurple;
  }
  .body-container {
    width: 100%;
    float: center;
 }
.details {
```

```
  width: 100%;
  float: center;
  height: 100px;
  text-align: justify;
}

.background-image {
  width: 100%;
  float: left;
  background-repeat: no-repeat;
  background-size: cover;
}
.image {
  width: 100%;
  height: 650px;
  float: left;
}
.content {
  position: absolute;
  padding: 20px;
```

```
  width: 100%;
  height: 100px;
  text-align: justify;
  color: darkred;
}

.title {
  text-align: center;
  padding: 10px;
}
.imgfile {
  margin: 30px;
}
.image-frame {
  border: 3px solid rgb(255, 255, 255);
  width: 200px;
  height: 200px;
  margin: auto;
}
.img::after {
```

```
  width: 200px;
  height: 200px;
}
.buttons {
  margin: 20px;
}

.btn {
  padding: 15px;
  margin: 20px;
  border: 1px solid black;
  border-radius: 3px;
  background-color: whitesmoke;
  cursor: pointer;
}
.submit-btn {
  text-align: center;
}
#clear_button{
  margin-left: 15px;
```

```
  font-weight: bold;
  color: rgb(0, 174, 255);
}
#predict_button{
  margin-right: 15px;
  color: rgb(0, 255, 72);
  font-weight: bold;
}
```

App.py:

```
import os


import numpy as np
from flask import Flask, render_template, request,
    send_from_directory, url_for
#from gevent.pywsgi import WSGIServer
from keras.models import load_model
from keras.preprocessing import image
from PIL import Image
from werkzeug.utils import redirect, secure_filename


UPLOAD_FOLDER = 'D:/NalaiyaThiran/projFiles/data'
```

```
app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

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

@app.route('/')
def index():
    return render_template('index.html')
@app.route('/web', methods=['GET', 'POST'])
def web():
    if request.method == "POST":
        f = request.files["image"]
        basepath = os.path.dirname(__file__)
        filepath = os.path.join(basepath, 'data', f.filename)
        f.save(filepath)
        # img = image.load_img(filepath, target_size=(64, 64))
        # x = image.img_to_array(img)
        # x = np.expand_dims(x, axis=0)
  # 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(filepath).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('web.html', num=str(num[0]))

    return render_template('web.html')

if __name__ == '__main__':

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

GITHUB

https://github.com/IBM-EPBL/IBM-Project-24481-1659943480

PROJECT DEMO LINK

https://github.com/IBM-EPBL/IBM-Project-24481-1659943480/blob/main/Final%20Deliverables/Project_Demo/Demo-Video.webm