Handwritten Digit Recognition - Report

Submitted by:

Charanya . S (312319205025)
Dharshini . N (312319205032)
Jane Ann Elson (312319205053)
Harshini . R (312319205046)

TABLE OF CONTENTS

S.NO	TITLE	PAGE NO
1	INTRODUCTION	
1.1	PROJECT OVERVIEW	4
1.2	PURPOSE	5
2	LITERATURE SURVEY	
2.1	EXISTING PROBLEM	6
2.2	REFERENCES	7
2.3	PROBLEM STATEMENT	15
3	IDEATION & PROPOSED SOLUTION	
3.1	EMPATHY MAP CANVAS	16
3.2	IDEATION & BRAINSTORMING	16
3.3	PROPOSED SOLUTION	21
3.4	PROBLEM - SOLUTION FIT	25
4	REQUIREMENT ANALYSIS	
4.1	FUNCTIONAL REQUIREMENTS	26
4.2	NON FUNCTIONAL REQUIREMENTS	28
5	PROJECT DESIGN	
5.1	DATA FLOW DIAGRAM	29
5.2	SOLUTION & TECHNICAL	31

	ARCHITECTURE	
5.3	USER STORIES	32
6	PROJECT PLANNING & SCHEDULING	
6.1	SPRINT PLANNING & ESTIMATION	33
6.2	SPRINT DELIVERY SCHEDULE	34
6.3	REPORTS FROM JIRA	35
7	CODING & SOLUTIONING	
7.1	FEATURE 1	38
7.2	FEATURE 2	39
8	TESTING	
8.1	TEST CASES	40
8.2	USER ACCEPTANCE TESTING	42
9	RESULTS	
9.1	PERFORMANCE METRICS	43
10	ADVANTAGES & DISADVANTAGES	45
11	CONCLUSION	46
12	FUTURE SCOPE	47
13	APPENDIX	
13.1	SOURCE CODE	48
13.2	GITHUB & DEMO LINK	54

1. INTRODUCTION

1.1 PROJECT OVERVIEW

Hand-written character and digit recognition have been one of the most exigent and engrossing field of pattern recognition and image processing. The task for handwritten digit recognition has been troublesome due to various variations in writing styles. Therefore, we have tried to create a base for future researches in the area so that the researchers can overcome the existing problems. The existing methods and techniques for handwritten digit recognition were reviewed and understood to analyze the most suitable and best method for digit recognition. A number of 60,000 images were used as training sets of images with pixel size of 28×28. The images/training sets were matched with original image. In this recognition task, the numbers are not accurately written or scripted as they differ in shape or size; due to which the feature extraction and segmentation of hand-written numerical script is arduous. The vertical and horizontal projections methods are used for the purpose of segmentation in the proposed work. SVM is applied for recognition and classification, while Convex hull algorithm is applied for feature extraction. The system uses the MINST dataset as a training sample and pre-processes the picture with the Opencv toolkit. Then it uses LeNet-5 in the convolutional neural network to extract the handwritten digit image features, repeatedly convolution pooling, and pull the result into a one-dimensional vector. And finally find the highest probability point to determine the result to achieve handwritten digit recognition with the Softmax regression model. The application of this system can greatly reduce labor costs and improve work efficiency, which is of great significance in many fields.

1.2 PURPOSE

Handwritten character recognition is one of the practically important issues in pattern recognition applications.The heart of the problem lies within the ability to develop an efficient algorithm that can recognize handwritten digits and which is submitted by users by the way of a scanner, tablet, and other digital devices.The applications of digit recognition includes in postal mail sorting, bank check processing, form data entry, etc.The main application of machine learning method over the last decade has determined efficacious in conforming decisive systems which are competing to human performance and which accomplish far improved than manually written classical artificial intelligence systems used in the beginnings of optical character recognition technology.The task of handwritten digit recognition, using a classifier, has great importance and use such as – online handwriting recognition on computer tablets, recognize zip codes on mail for postal mail sorting, processing bank check amounts, numeric entries in forms filled up by hand (for example ‐ tax forms) and so on. There are different challenges faced while attempting to solve this problem. The handwritten digits are not always of the same size, thickness, or orientation and position relative to the margins. Our goal was to implement a pattern classification method to recognize the handwritten digits provided in the MINIST data set of images of handwritten digits (0‐9).CNN is used as the Model for the classification of the image and more specifically Keras Sequential Model is used as a classifier. The image preprocessing is the most important step which has done with the help of OpenCV and Scipy. MNIST is the dataset used for training and testing. Handwritten Digit Recognition has various real-life time uses. It is used in the detection of vehicle number, banks for reading cheques, post offices for arranging letter, and many other tasks.

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

In handwriting recognition (HWR) the device interprets the user's handwritten characters or words into a format that the computer understands (e.g., Unicode text). The input device typically comprises a stylus and a touch-sensitive screen. There are many levels of HWR, starting from the recognition of simplified individual characters to the recognition of whole words and sentences of cursive handwriting. However, the existing system had sophisticated features like the recognition of common shapes and symbols but the recognition accuracy was often not very good. The suboptimal handwriting recognition was unquestionably among the reasons for the system's downfall. 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

Paper 1: Handwritten Digit Recognition of MNIST dataset using Deep Learning state-of-the-art Artificial Neural Network (ANN) and Convolutional Neural Network (CNN)

Drishti Beohar, Akhtar Rasool 2021 International Conference on Emerging Smart Computing and Informatics (ESCI)

Handwritten digit recognition is an intricate assignment that is vital for developing applications, in computer vision digit recognition is one of the major applications. There has been a copious exploration done in the Handwritten Character Recognition utilizing different deep learning models. Deep learning is rapidly increasing in demand due to its resemblance to the human brain. The two major Deep learning algorithms Artificial Neural Network and Convolutional Neural Network which have been compared in this paper considering their feature extraction and classification stages of recognition. The models were trained using categorical cross-entropy loss and ADAM optimizer on the MNIST dataset. Backpropagation along with Gradient Descent is being used to train the networks along with reLU activations in the network which do automatic feature extraction. In neural networks, Convolution Neural Network (ConvNets or Convolutional neural networks) is one of the primary classifiers to do image recognition, image classification tasks in Computer Vision.

Paper 2: Handwritten Digit Recognition Using Machine Learning Algorithms

S M Shamim Global Journal of Computer Science and Technology, 18(1), 17–23, 2018-04-13

Handwritten character recognition is one of the practically important issues in pattern recognition applications. The applications of digit recognition includes in postal mail sorting, bank check processing, form data entry, etc. The heart of the problem lies within the ability to develop an efficient algorithm that can recognize hand written digits and which is submitted by users by the way of a scanner, tablet, and other digital devices. This paper presents an approach to off-line handwritten digit recognition based on different machine learning technique. The main objective of this paper is to ensure effective and reliable approaches for recognition of handwritten digits. Several machines learning algorithm namely, Multilayer Perceptron, Support Vector Machine, NaFDA5; Bayes, Bayes Net, Random Forest, J48 and Random Tree has been used for the recognition of digits using WEKA. The result of this paper shows that highest 90.37% accuracy has been obtained for Multilayer Perceptron

Paper 3: Character Recognition using Artificial Neural Network

Pranjali Pohankar, Namrata Taralkar, Snehalata Karmare, Smita Kulkarni International Journal of Electronics Communication and Computer Engineering 5 (4), 2014

A neural network is a machine designed to model the way in which the brain performs a particular task. Character recognition techniques help in recognizing the characters written on paper documents and converting it in digital form. Handwritten character recognition is a very difficult problem due to great variation of writing style, different size and shape of the character. Neural network is a technique used to improve the accuracy and efficiency of the handwritten character recognition system. The error back propagation algorithm is used to train the MLP networks. The main advantage of the back propagation neural network (BPN) method is that it can fairly approximate a large class of functions. The aim of the paper is to use the improved neural network technique to recognize the offline handwritten characters.

Paper 4: Handwritten Digit Recognition Based on Convolutional Neural Network

Chao Zhang, Zhiyao Zhou, Lan Lin Department of Electronic Science and Technology Tongji University Shanghai

In order to meet the needs of paperless offices and greatly improve work efficiency, it is necessary to research and implement a handwritten digit recognition system. Handwritten digit recognition plays an important role in large-scale data statistics and the financial business, such as industry annual inspection, population census, tax statements and checks, etc. This paper proposes a new type of handwritten digit recognition system based on convolutional neural networks (CNN). In order to improve the recognition performance, the network was trained with a large number of standardized pictures to automatically learn the spatial characteristics of handwritten digits. For model training, according to the loss function, the convolutional neural network continuously updates the network parameters with the data set in MNIST, which contains 60,000 examples. For model tests, the system uses the camera to capture the pictures composed of the images generated by the test data set of MNIST and the samples written by different people, then continuously processes the captured graphics and refreshes the output every 0.5 seconds. With the trained deep learning model, we got a recognition accuracy of 97.3% in the test process. Good performance in this experiment shows that our system can automatically recognize the handwritten digital content appearing in the target area and output the content label in real time.

Paper 5: Offline handwritten digit recognition using neural network

Sumedha B Hallale, Geeta D Salunke International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering 2 (9), 4373-4377, 2013

Optical character recognition is a typical field of application of automatic classification methods. In this paper, we have introduced a whole new idea of recognition of isolated handwritten digits which is known to be a difficult task and still lacks a satisfactory technical solution. The present paper proposes a novel approach for recognition of handwritten digits ie neural network classification. Back propagation neural network is one of the simplest methods for training multilayer neural networks. In this paper, we designed a back propagation neural network and trained it with a set of handwritten digits. The average success rates of recognition of all digits are 91.2%.

Paper 6: Simplified Neural Network Design for Hand Written Digit Recognition

Muhammad Zubair Asghar, Hussain Ahmad, Shakeel Ahmad, Sheikh Muhammad Saqib, Bashir Ahmad, Muhammad Junaid Asghar International Journal of Computer Science and Information Security 9 (6), 319, 2011

Neural Network is an abstraction of the central nervous system and works as a parallel processing system. Optimization, image processing, Diagnosis and many other applications are made very simple through neural networks, which are difficult and time consuming when conventional methods are used for their implementation. Neural Network is the simplified version of the human brain. Like the human brain, neural networks also exhibit efficient performance on perceptive tasks like recognition of visual images of objects and handwritten characters etc: Recognition of handwritten digits is one of the oldest applications of ANN. The recognition of digits written in different handwritings and also from scanned text has remained a trouble thus it has received much attention from researchers in the field of artificial neural networks. In this research work a very simple and flexible neural network scheme is proposed and implemented for handwritten digit recognition, which will assist beginners and AI students who want to understand the perceptive capability of neural networks. In the proposed system, a very simple design of artificial neural networks is implemented.

Paper 7: Persian Handwritten Digit Recognition using Support Vector Machines

Omid Rashnodi,Hedieh Sajedi,Mohammad Saniee Abadeh International Journal of Computer Applications (0975 –8887), Volume 29–No.12, September 2011

In this paper, appropriate features set based on Discrete Fourier Transform coefficients and the box approach have been proposed to achieve higher recognition accuracy, decreasing the features set dimensions and recognition time of Persian numerals. In classification phase, support vector machine (SVM) has been employed as the classifier. Feature sets consists of 154 dimensions, which are the Fourier coefficients in the contour pixels of input image, average angle and distance pixels which are equal to one in each boxthe box approach.The scheme has been evaluated on 80,000 handwritten samples of Persian numerals. Using 60,000 samples for training, scheme was testedon other 20,000 samples and 98.84% correct recognition rate was obtained.

Paper 8: A Survey on using Neural Networkbased Algorithms forHandWritten Digit Recognition

Muhammad Ramzan,Shahid Mehmood Awan,Hikmat Ullah Khan,,Waseem Akhtar,Ammara Zamir,Mahwish Ilyas,Ahsan Mahmood International Journal of Advanced Computer Science and Applications,Vol. 9, No. 9, 2018

The detection and recognition of handwritten content is the process of converting non-intelligent information such as images into machine edit-able text. This research domain has become an active research area due to vast applications in a number of fields such as handwritten filing of forms or documents in banks, exam form filled by students, users' authentication applications. Generally, the handwritten content recognition process consists of four steps: data preprocessing, segmentation, the feature extraction and selection, application of supervised learning algorithms. In this paper, a detailed survey of existing techniques used for Handwritten Digit Recognition(HWDR) is carried out. This review is novel as it is focused on HWDR and also it only discusses the application of Neural Network (NN) and its modified algorithms. We discuss an overview of NN and different algorithms which have been adopted from NN. In addition, this research study presents a detailed survey of the use of NN and its variants for digit recognition. Each existing work, we elaborate its steps, novelty, use of dataset and advantages and limitations as well. Moreover, we present a Scientometricanalysis of HWDR which presents top journals and sources of research content in this research domain. We also present research challenges and potential future work.

Paper 9: Handwritten Digits Recognition with Decision tree Classification : a Machine learning approach

Tsehay Admassu Assegie, Pramod Sekharan Nair International Journal of Electrical and Computer Engineering (IJECE) ,Vol. 9, No. 5, October 2019.

Handwritten digits recognition is an area of machine learning, in which a machine is trained to identify handwritten digits. One method of achieving this is with decision tree classification model. A decision tree classification is a machine learning approach that uses the predefined labels from the past known sets to determine or predict the classes of the future data sets where the class labels are unknown. In this paper we have used the standard kaggle digits dataset for recognition of handwritten digits using a decision tree classification approach. And we have evaluated the accuracy of the model against each digit from 0 to 9.

Paper 10: Handwritten Digit Recognition Using Various Machine Learning Algorithms and Models

Pranit Patil International Journal of Innovative Research in Computer Science & Technology (IJIRCST) ISSN: 2347-5552, Volume- 8, Issue- 4, July- 2020

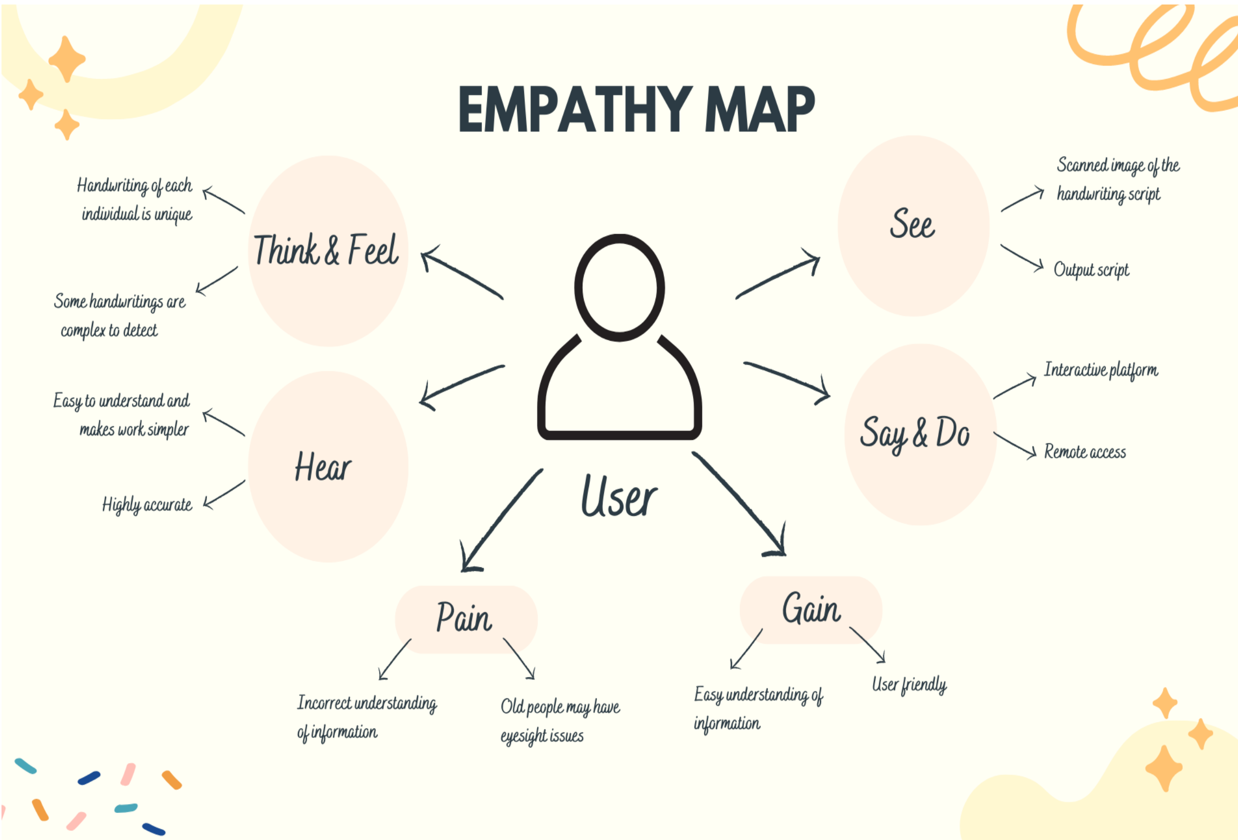
Handwritten digit recognition is a technique or technology for automatically recognizing and detecting handwritten digital data through different Machine Learning models. In this paper we use various Machine Learning algorithms to enhance the productiveness of technique and reduce the complexity using various models. Machine Learning is an application of Artificial Intelligence that learns from previous experience and improves automatically through experience. We illustrate various Machine learning algorithms such as Support Vector Machine, Convolutional Neural Network, Quantum Computing, K-Nearest Neighbor Algorithm, Deep Learning used in Recognition technique.

2.3 PROBLEM STATEMENT

Our system aims to accurately recognize digit given as the input in the form of image or document in order to quickly identify different handwriting styles and to minimize the need for human interpretation. A few applications of the system would be postal mail sorting, bank cheque processing, form data entry, etc. We aim to develop 7-layered CNN model with 5-hidden layer along with Gradient descent and Back propagation model to find and compare accuracy on different Epochs. We achieve this by using MNIST Dataset ,which contains 60,000 handwritten digit images for the classifier training and 10,000 handwritten digit images for the classifier testing. Finally, the recognized digit will be displayed in the user interface. 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. 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. The uniqueness and variety in the handwriting of different individuals also influence the formation and appearance of the digits.

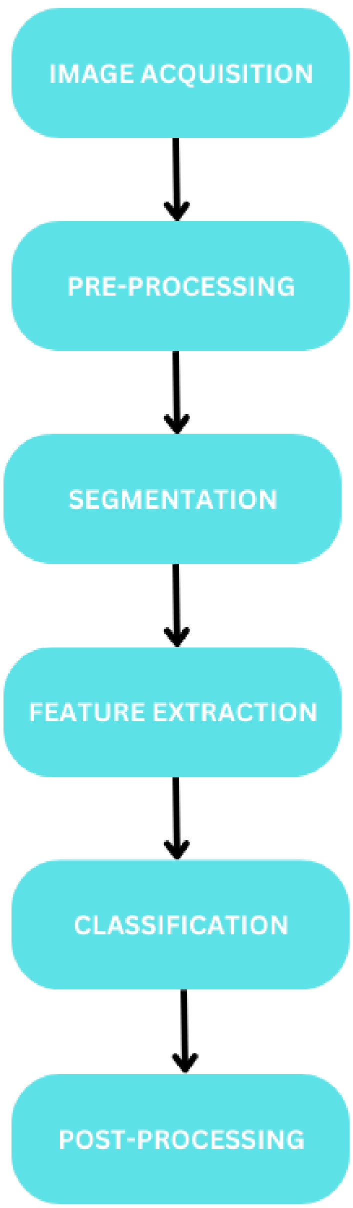
3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

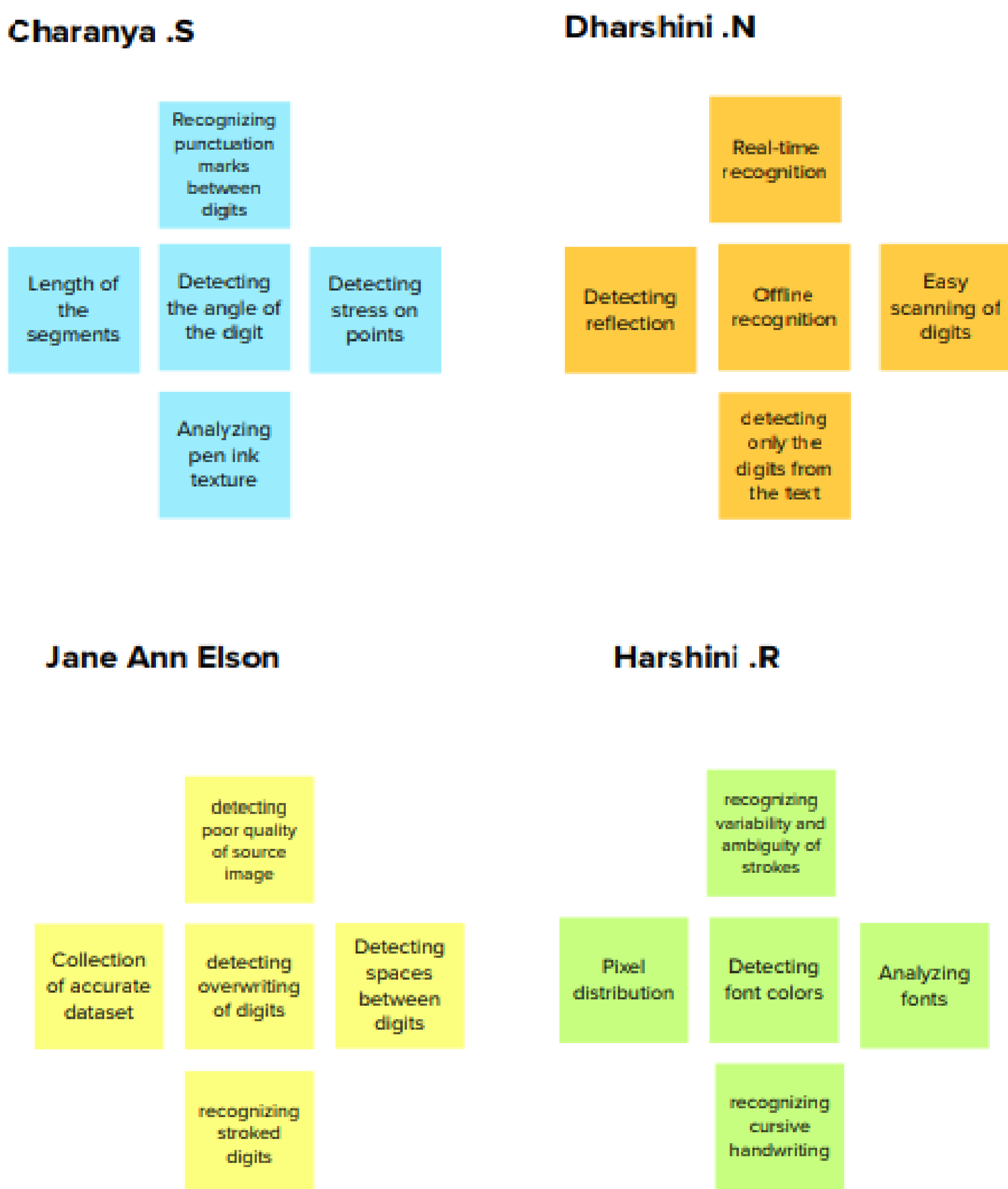


3.2 IDEATION & BRAINSTORMING

STEP 1:



STEP 2:


Charanya .S
Recognizing punctuation marks between digits
Length of the segments
Detecting the angle of the digit
Detecting stress on points
Analyzing pen ink texture
Dharshini .N
Real-time recognition
Detecting reflection
Offline recognition
Easy scanning of digits
detecting only the digits from the text
Jane Ann Elson
detecting poor quality of source image
Collection of accurate dataset
detecting overwriting of digits
Detecting spaces between digits
recognizing stroked digits
Harshini .R
recognizing variability and ambiguity of strokes
Pixel distribution
Detecting font colors
Analyzing fonts
recognizing cursive handwriting

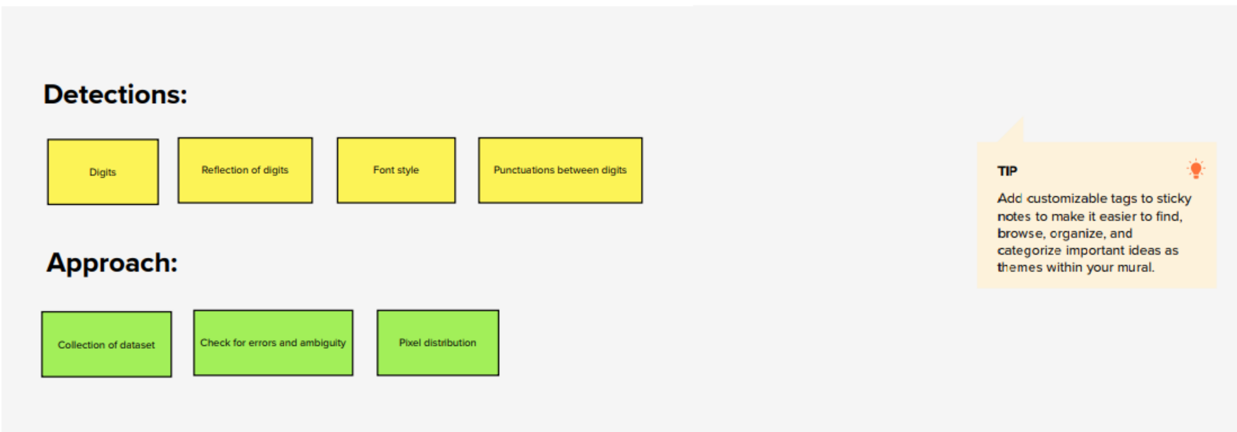
STEP 3:

3

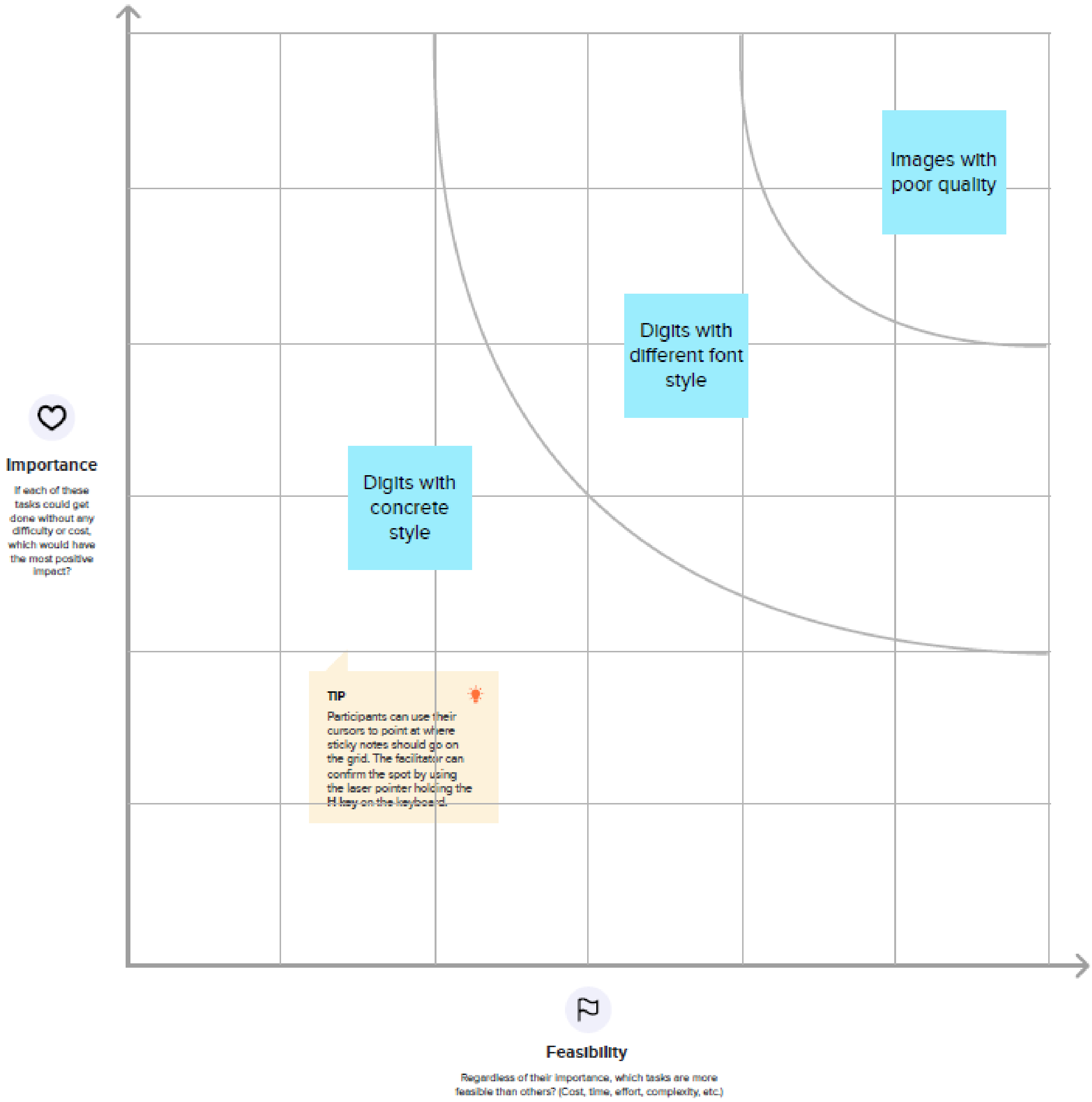
Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes



STEP 4:



Images with poor quality
Digits with different font style
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Digits with concrete style
TIP
Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the H key on the keyboard.
Feasibility
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The handwritten digit recognition is the ability of computers to recognize human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different flavors. The handwritten digit recognition is the solution to this problem which uses the image of a digit and recognizes the digit present in the image. The handwritten digit recognition is the ability of computers to recognize human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different flavors. The handwritten digit recognition is the solution to this problem which uses the image of a digit and

		recognizes the digit present in the image.
2.	Idea / Solution description	Our system aims to accurately recognize digit given as the input in the form of image or document in order to quickly identify different handwriting styles and to minimize the need for human interpretation. A few applications of the system would be postal mail sorting, bank cheque processing, form data entry, etc. We aim to develop 7-layered CNN model with 5-hidden layer along with Gradient descent and Back propagation model to find and compare accuracy on different Epochs. We achieve this by using MNIST Dataset, which contains 60,000 handwritten digit images for the classifier training and 10,000 handwritten digit images for the classifier testing. Finally, the recognized digit will be displayed in the user interface

3.	Novelty / Uniqueness	This work aims to identify digits of varied languages. It is not suppressed to standard numericals but is extended to digits of all global languages.
4.	Social Impact / Customer Satisfaction	This approach recognizes number plates of vehicles, numeric entries in forms filled up by hand (say — tax forms), postal mail sorting, bank check processing, form data entry and makes understanding better for the customers.
5.	Business Model (Revenue Model)	We propose a two-tier system namely a ‘free’ and a ‘premium’ tier. The free tier would include ads that display below when classifying a website. The premium tier is a recurring subscription either monthly or annually that removes all advertisements, increasing speed and efficiency of

		the classifying models.
6.	Scalability of the Solution	This solution can be used by people working in different fields, in different environments ,at ease. It can be accessed and used by everyone across the world. It prevents irregularities in handwritten papers and provides high accuracy.

3.4 PROBLEM - SOLUTION FIT

Problem-Solution fit

Purpose / Vision

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?
i.e. working parents of 0-5 y.o. kids

The customer base is anyone who want to identify or recognize handwritten digits.

5. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.

The main constraints would include,

1. Not educated about this concept.
2. Not knowing how to use application.

6. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons dothese solutions have? i.e. pen and paper is an alternative to digital notetaking

1.Our system aims to accurately recognize digit given as the input in the form of image or document in order to quickly identify different handwriting styles and to minimize the need for human interpretation.
.2 We achieve this by using MNIST Dataset, which contains 60,000 handwritten digit images for the classifier training and 10,000 handwritten digit images for the classifier testing. Finally, the recognized digit will be displayed in the user interface

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

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

Which jobs-to-be-done (or problems) do you address for your customers?There could be more than one; explore different sides.

1. Assist users and help them identify handwritten digits digitally.
2. Educate the users about how to use the application.

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists? What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

1. Manual written are difficult to recognize because there various handwriting .
2. It is difficult when processing bank checks.
3. It is useful for recognizing Vehicle number plates.

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

Since manual work can cause inaccuracy,uncertainity.This solution provides accurate and certain results and it is also more efficient and effective.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

Manual work done by customers can causes uncertainty. But this approach recognizes number plates of vehicles, numeric entries in forms filled up by hand (say — tax forms), postal mail sorting, bank check processing, form data entry and makes understanding better for the customers.

4. EMOTIONS: BEFORE / AFTER EM

How do customers feel when they face a problem or a job and afterwards?
i.e. lost, insecure > confident, in control - use it in your communication strategy & design.

Before: Inaccuracy, uncertainty ,difficult to analyze.
After: Certainty, Accuracy, efficient and effective.

10. YOUR SOLUTION SL

Our system aims to accurately recognize digit given as the input in the form of image or document in order to quickly identify different handwriting styles and to minimize the need for human interpretation. A few applications of the system would be postal mail sorting, bank cheque processing, form data entry, etc. We aim to develop 7-layered CNN model with 5-hidden layer along with Gradient descent and Back propagation model to find and compare accuracy on different Epochs. We achieve this by using MNIST Dataset, which contains 60,000 handwritten digit images for the classifier training and 10,000 handwritten digit images for the classifier testing. Finally, the recognized digit will be displayed in the user interface

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE
What kind of actions do customers take online? Extract online channels from #7

Promote the browser extension by leaving a review and eductating themselves about web phishing

8.2 OFFLINE
What kind of actions do customers take offline? Extract offline channels from #7and use them for customer development.

Educate more people about the potential dangers of websites stealing the user's data. They could also ask their acquaintances to use the browser extension.

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	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Authentication	Verify the user
FR-4	Provide handwritten digits	Give input to the application to read the handwritten digits.

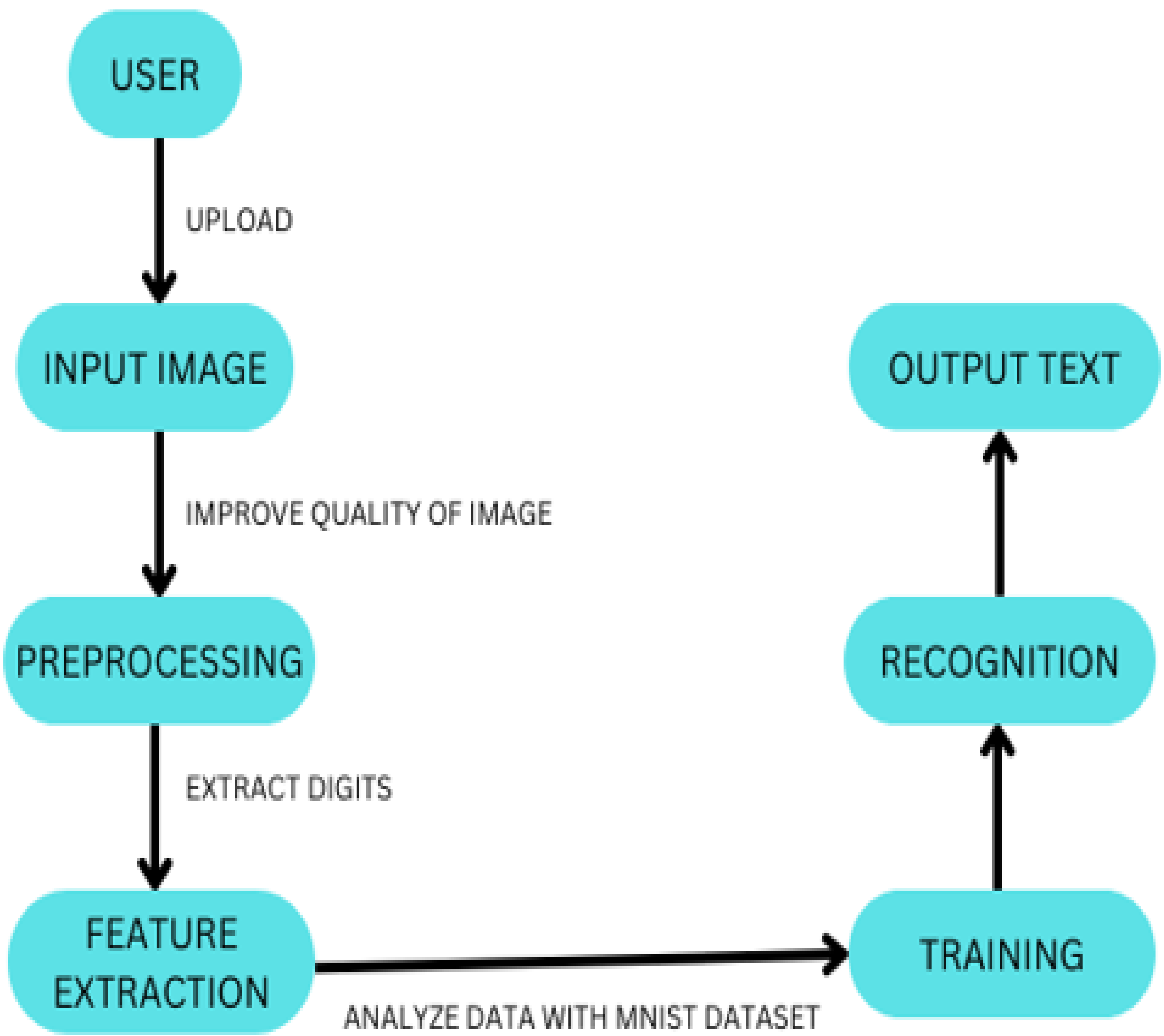
FR-5	Recognition of digits	Recognized digits are shown with higher accuracy.

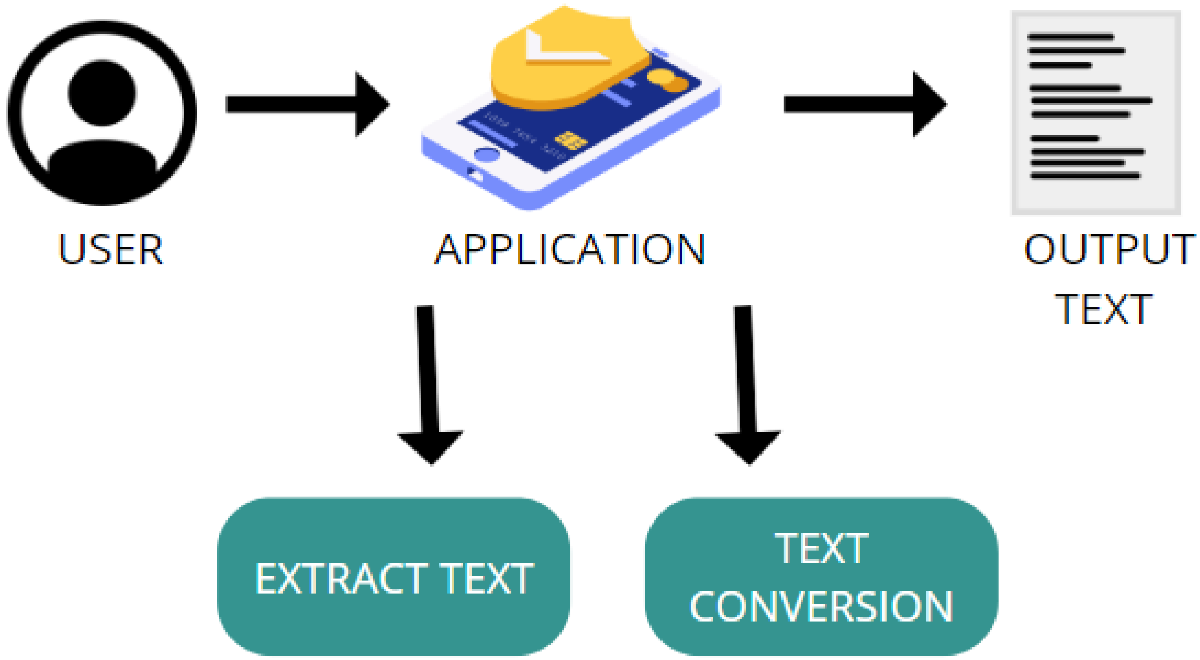
4.2 NON FUNCTIONAL REQUIREMENTS

FR No.	Non-Functional Requirements	Description
NFR-1	Usability	The application is user friendly.
NFR-2	Security	Data is secured.
NFR-3	Reliability	Highly reliable and authentic
NFR-4	Performance	Recognition of handwritten digits is accurate.
NFR-5	Availability	Deployed in cloud so it is accessible.
NFR-6	Scalability	This application can be used by people working in different fields and environment at ease.

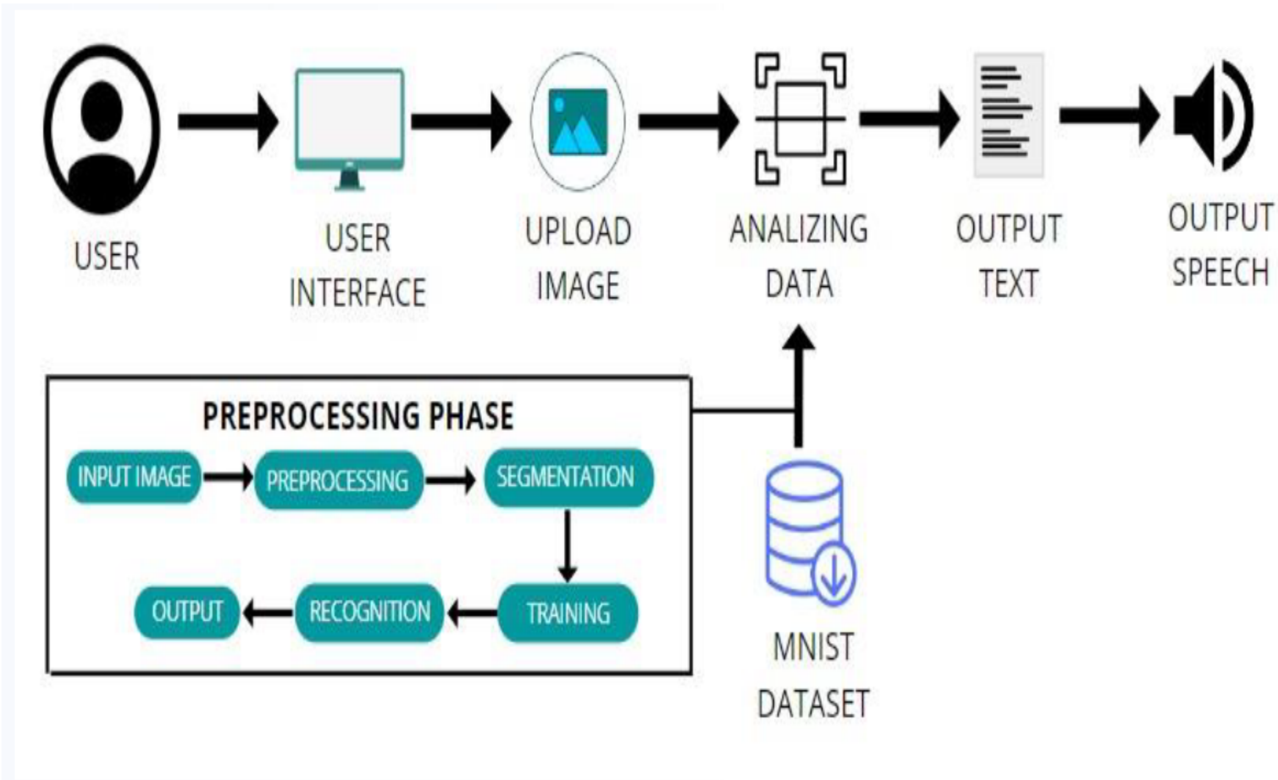
5. PROJECT DESIGN

5.1 DATAFLOW DIAGRAM




USER
APPLICATION
OUTPUT
TEXT
EXTRACT TEXT
TEXT
CONVERSION

5.2 SOLUTION & TECHNICAL ARCHITECTURE



5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	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 Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	Login successful message will be displayed	High	Sprint-1
	Dashboard					
Customer (Web 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 Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	Login successful message will be displayed	High	Sprint-1
	Dashboard		Access all resources		High	Sprint-2

6. PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

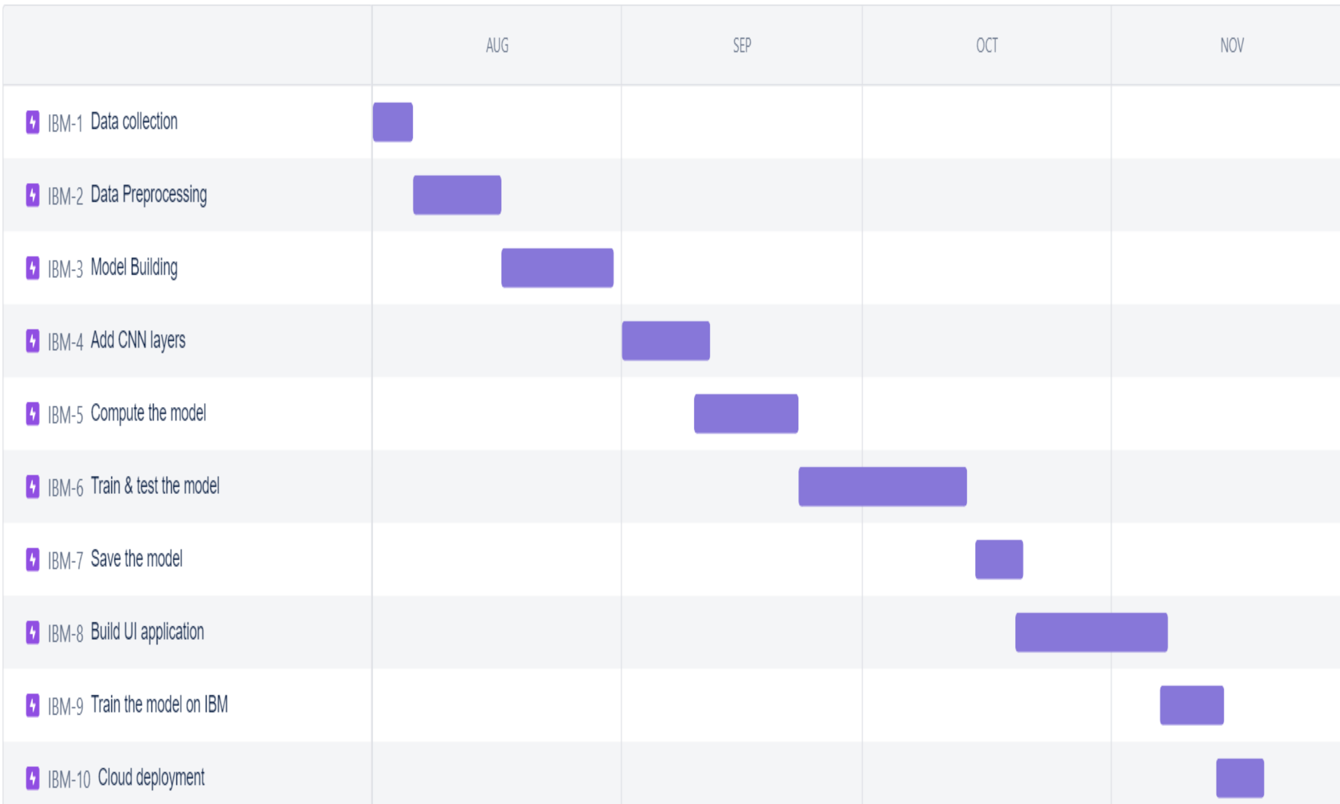
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	As a user, I can collect the dataset from various resources with different handwritings.	10	Low	Jane Ann Elson Harshini R
Sprint-1	Data Preprocessing	USN-2	As a user, I can load the dataset, handling the missing data, scaling and split data into train and test.	10	Medium	Charanya S Dharshini N
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	Jane Ann Elson Harshini R Charanya S Dharshini N
Sprint-2	Add CNN layers	USN-4	Creating the model and adding the input, hidden, and output layers to it.	5	High	Jane Ann Elson Harshini R Charanya S Dharshini N
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	Harshini R
Sprint-2	Train & test the model	USN-6	As a user, let us train our model with our image dataset.	6	Medium	Charanya S Dharshini N
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	Dharshini N Jane Ann Elson
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	Charanya S Dharshini N
Sprint-3		USN-9	As a user, I can know the details of the fundamental usage of the application.	5	Low	Charanya S
Sprint-3		USN-10	As a user, I can see the predicted / recognized digits in the application.	5	Medium	Harshini R Jane Ann Elson
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	Jane Ann Elson Harshini R
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	Jane Ann Elson Harshini R Charanya S Dharshini N

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 2022S	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

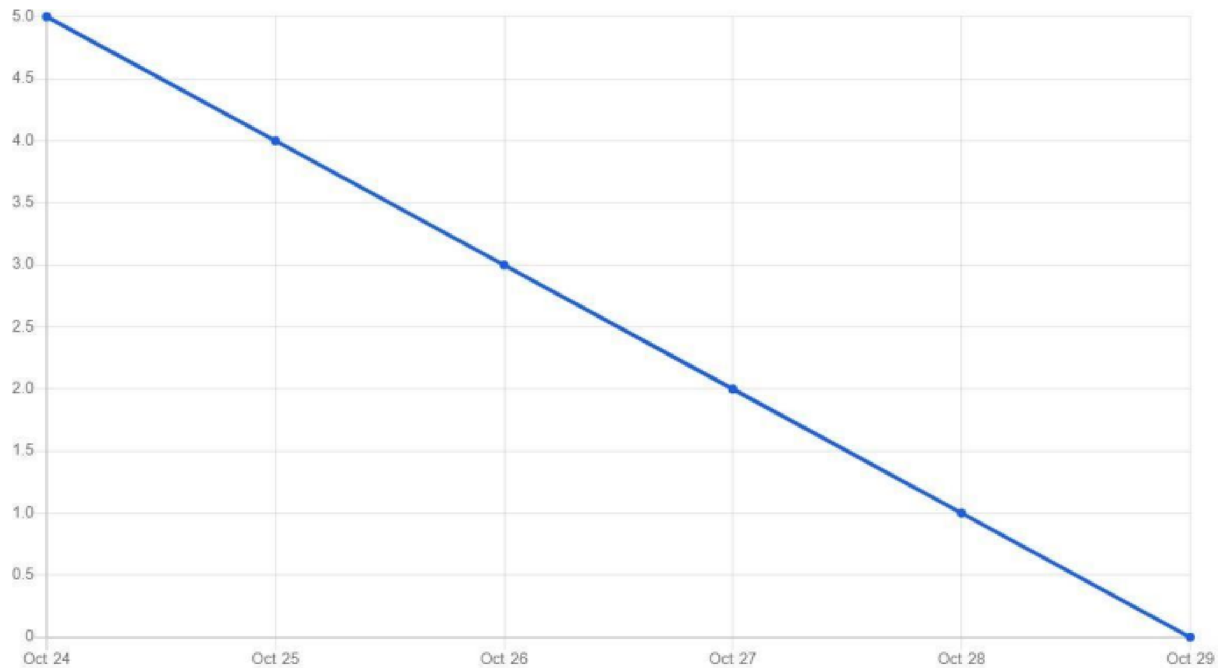
6.3 REPORTS FROM JIRA

Roadmap:



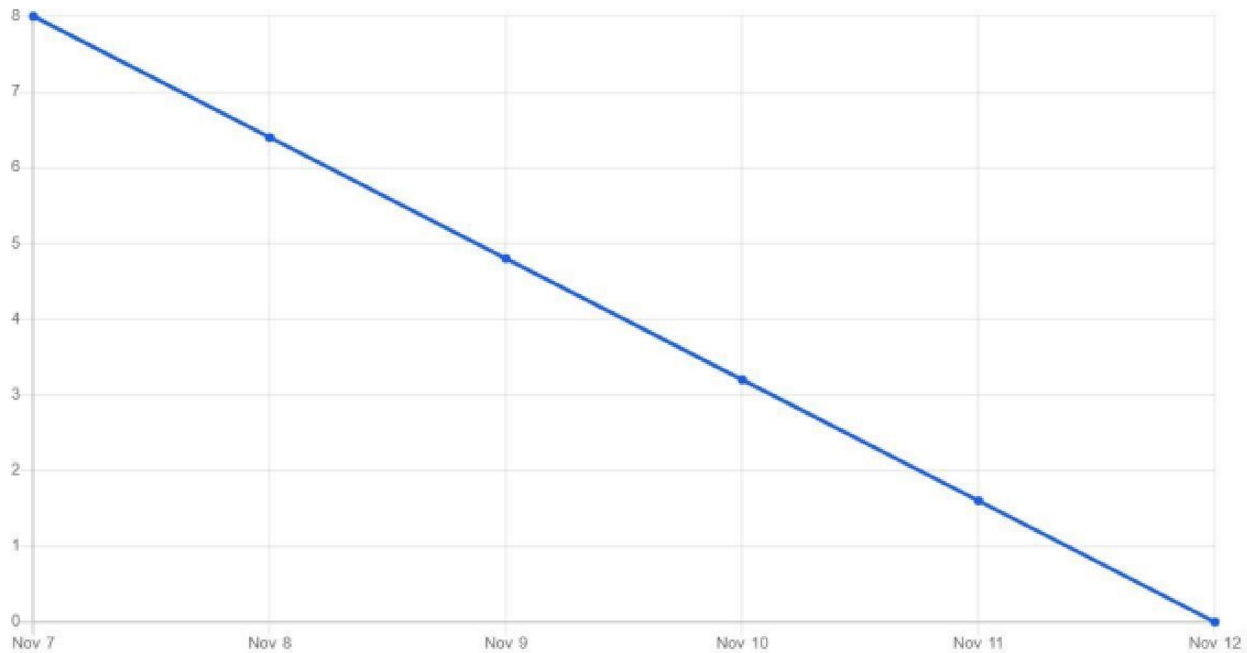
Burndown chart:

Sprint 1:



Sprint 2:

Sprint 3:



Sprint 4:

7. CODING & SOLUTIONING

7.1 FEATURE 1

```
#Add CNN Layers
#create model
model=Sequential ()
#adding modeL Layer
model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation='relu'))
model.add(Conv2D(32, (3, 3), activation = 'relu'))
#flatten the dimension of the image
model.add(Flatten())
#output layer with 10 neurons
model.add(Dense(number_of_classes,activation = 'softmax'))
```

Convolutional layer is a simple application of a filter which acts as an activation function. What this does is takes a feature from a input image, then filter different features from that image and makes a feature map. Some of the features are location, strength etc. the filter is then moved over the whole image and the value of each pixel is calculated. This is the part where the actual classification happens. All the matrix from the pooling layer is stacked up here and put into a single list. The values which are higher are the points of prediction for the given image.

7.2 FEATURE 2

```
        # Grid structure
        self.canvas.grid(row=0, column=0, pady=2, sticky=W, )
        self.label.grid(row=0, column=1,pady=2, padx=2)
        self.classify_btn.grid(row=1, column=1, pady=2, padx=2)
        self.button_clear.grid(row=1, column=0, pady=2)

        #self.canvas.bind("<Motion>", self.start_pos)
        self.canvas.bind("<B1-Motion>", self.draw_lines)

    def clear_all(self):
        self.canvas.delete("all")

    def classify_handwriting(self):
        HWND = self.canvas.winfo_id() # get the handle of the canvas
        rect = win32gui.GetWindowRect(HWND) # get the coordinate of the canvas
        im = ImageGrab.grab(rect)

        digit, acc = predict_digit(im)
        self.label.configure(text= str(digit)+', '+ str(int(acc*100))+'%')

    def draw_lines(self, event):
        self.x = event.x
        self.y = event.y
        r=8
        self.canvas.create_oval(self.x-r, self.y-r, self.x + r, self.y + r, fill='black')
```

For recognizing real-time handwritten digits, the input scanned handwritten image is converted to gray scale. With the help of threshold technique the pixel values are assigned to the input image and then handwritten digits are detected and preprocessed. These preprocessed images are predicted with the proposed model and display a digital output.

8. TESTING

8.1 TEST CASES

Test caseID	Feature Type	Component	Test Scenario	Expected Result	Actual Result	Status
TC_001	UI	Home Page	Verify UI elements in the Home Page	The Home page must be displayed properly	Working as expected	FAIL
TC_002	UI	Home Page	Check if the UI elements are displayed properly in different screen sizes	The Home page must be displayed properly in all sizes	The UI is not displayed properly in screen size 2560 x 1801 and 768 x 630	FAIL
TC_003	Functional	Home Page	Check if user can upload their file	The input image should be uploaded to the application successfully	Working as expected	PASS
TC_004	Functional	Home Page	Check if user cannot upload unsupported files	The application should not allow user to select a non image file	User is able to upload any file	FAIL
TC_005	Functional	Home Page	Check if the page redirects to the result page once the input is given	The page should redirect to the results page	Working as expected	PASS

TC_006	Functional	Backend	Check if all theroutes are working properly	All the routes should properly work	Working as expected	PASS
TC_007	Functional	Model	Check if the model can handle various image sizes	The model should rescale the image and predict the results	Working as expected	PASS
TC_008	Functional	Model	Check if the model predicts the digit	The model should predict the number	Working as expected	PASS
TC_009	Functional	Model	Check if the model can handle complex input image	The model should predict the number in the complex image	The model fails to identify the digit since the model is not built to handle such data	FAIL
TC_010	UI	Result Page	Verify UI elements in the Result Page	The Result page must be displayed properly	Working as expected	PASS

8.2 USER ACCEPTANCE TESTING

Defect Analysis:

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Total
By Design	1	0	1	0	2
Duplicate	0	0	0	0	0
External	0	0	2	0	2
Fixed	4	1	0	1	6
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	1	1
Won't Fix	1	0	1	0	2
Total	6	1	4	3	14

Test case Analysis:

Section	Total Cases	Not Tested	Fail	Pass
Client Application	10	0	3	7
Security	2	0	1	1
Performance	3	0	1	2
Exception Reporting	2	0	0	2

9. RESULTS

9.1 PERFORMANCE METRICS

S.No.	Parameter	Values	Screenshot
1	Model Summary	The model has two main aspects: the feature extraction front end comprised of convolutional and pooling layers, and the classifier backend that will make a prediction.	
2	Accuracy	Training Accuracy – 98.03%	

		Validation Accuracy – 99.24%	
3	Confidence Score (Only Yolo Projects)	Class Detected - NA Confidence Score - NA	

10. ADVANTAGES & DISADVANTAGES

Advantages:

- Reduces manual work
- More accurate than average human
- Capable of handling a lot of data
- Can be used anywhere from any device
- Neural Network is used to train and identify written digits for greater efficiency.
- The accuracy rate is very high.
- Speed of data entry
- It is much easier to dictate the machine than to write
- Easier data retrieval

Disadvantages:

- Cannot handle complex data
- All the data must be in digital format
- Requires a high performance server for faster predictions
- Prone to occasional errors
- There is a wide range of handwriting – good and bad.
- It is tricky for programmers to provide enough examples of how every character might look.
- Customers must try with clear image and neat handwriting to get accuracy in digits.
- Unclear image will not give accurate results.

11. CONCLUSION

In this work, with the point of improving the exhibition of transcribed digit acknowledgment, we assessed variations of a convolutional neural organization to keep away from complex pre-preparing, exorbitant component extraction and a perplexing troupe approach of a conventional acknowledgment framework. Through broad assessment utilizing a MNIST dataset, the current work recommends the job of different hyper-boundaries. We additionally confirmed that tweaking of hyper-boundaries is fundamental in improving the presentation of CNN engineering. We accomplished acknowledgment pace of 99.89% for the MNIST information base, which is superior to all recently revealed outcomes. The impact of expanding the quantity of convolutional layers in CNN design on the presentation of transcribed digit acknowledgment is unmistakably introduced through the tests. The oddity of the current work is that it altogether explores all the boundaries of CNN engineering that convey best acknowledgment precision for a MNIST dataset. Companion scientists couldn't coordinate this precision utilizing an unadulterated CNN model. A few analysts utilized gathering CNN network models for the equivalent dataset to improve their acknowledgment precision at the expense of expanded computational expense and high testing multifaceted nature yet with practically identical exactness as accomplished in the present work.

12. FUTURE WORK

In future, various designs of CNN, in particular, cross breed CNN, viz., CNN-RNN and CNN-HMM models, and space explicit acknowledgment frameworks, can be researched. Developmental calculations can be investigated for streamlining CNN learning boundaries, to be specific, the quantity of layers, learning rate and portion sizes of convolutional channels. This project can be enhanced with a great field of machine learning and artificial intelligence. The world can think of a software which can recognise the text from a picture and can show it to the others, for example a the shop name detector. Or this project can be extended to a greater concept of all the character sets in the world. This project has not gone for the total english alphabet because there will be more and many more training sets and testing values that the neural network model will not be enough to detect. Think of a AI modeled car sensor going with a direction modeling in the roadside, user shall give only the destination. All of these enhancement is an application of the texture analysis where advanced image processing. Neural network model for training and advanced AI concepts will come.These applications can be modeled further. As this project is fully done by free and available resources and packages this can be also a limitation of the project.

13. APPENDIX

13.1 SOURCE CODE

Handwritten.py:

```
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 #convolutional Layer
from keras.utils import np_utils #used for one-hot encoding
import matplotlib.pyplot as plt   #used for data visualization


# In[3]:


(x_train, y_train), (x_test, y_test)=mnist.load_data () #splitting the mnist data into train and test
print (x_train.shape)  #shape is used for give the dimens ion values #60000-rows 28x28-pixels
print (x_test.shape)
x_train[0]
```

```
#Add CNN Layers
#create model
model=Sequential ()
#adding modeL Layer
model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation='relu'))
model.add(Conv2D(32, (3, 3), activation = 'relu'))
#flatten the dimension of the image
model.add(Flatten())
#output layer with 10 neurons
model.add(Dense(number_of_classes,activation = 'softmax'))


# In[8]:


#Compile model
model.compile(loss= 'categorical_crossentropy', optimizer="Adam", metrics=['accuracy'])
x_train = np.asarray(x_train)
y_train = np.asarray(y_train)


# In[9]:


#fit the model
model.fit(x_train, y_train, validation_data=(x_test, y_test), epochs=5, batch_size=32)
```

```python
model.save('mnistCNN.h5')
get_ipython().system('tar -zcvf handwritten-digit-recognition-model_new.tgz mnistCNN.h5')


# In[19]:


get_ipython().system('pip install watson-machine-learning-client --upgrade')


# In[42]:


from ibm_watson_machine_learning import APIClient
credentials ={
    "url":"https://us-south.ml.cloud.ibm.com",
    "apikey":"qMWf3TNlOnECf3QWhXLLXXlSjol3iwYANNKEj1SvR4UQ"
}
client = APIClient(credentials)


# In[43]:


client.spaces.get_details()
```

```python
def guid_from_space_name(client,deploy):
  space = client.spaces.get_details()
  return (next(item for item in space['resources'] if item['entity']['name']==deploy)['metadata']['id'])


# In[44]:


client.spaces.get_details()


# In[54]:


space_uid = guid_from_space_name(client, 'sparks')
print("Space UID = " +space_uid)


# client.set.default_space(space_uid)

# In[55]:


client.set.default_space(space_uid)
```

```
client.software_specifications.list()
software_space_uid = client.software_specifications.get_uid_by_name('tensorflow_rt22.1-py3.9')
software_space_uid
model_details = client.repository.store_model(model='handwritten-digit-recognition-model_new.tgz',meta_props={
    client.repository.ModelMetaNames.NAME:"CNN Digit recognition model",
    client.repository.ModelMetaNames.TYPE:"tensorflow_2.7",
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_space_uid
})
model_details
model_id = client.repository.get_model_id(model_details)
model_id
client.repository.download(model_id,'DigitRecog_IBM_model1.tar.gz')

ls
from tensorflow.keras.models import load_model
from keras.preprocessing import image
from PIL import Image
import numpy as np
model = load_model("mnistCNN.h5")
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.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='is_QZGPyU8oxZr3W-td-LCHXS3QPMaWArILi18FdSyGT',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.ap.cloud-object-storage.appdomain.cloud')

bucket = 'handwrittenimagerecognition-donotdelete-pr-8tlrnykut46vpi'
object_key = 'mnist-dataset-1024x424 (2).png'

streaming_body_1 = cos_client.get_object(Bucket=bucket, Key=object_key)['Body']

# Your data file was loaded into a botocore.response.StreamingBody object.
# Please read the documentation of ibm_boto3 and pandas to learn more about the possibilities to load the data.
# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/
# pandas documentation: http://pandas.pydata.org/
img = Image.open(streaming_body_1).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)
print(pred)


# In[ ]:




# In[66]:


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

Gui.py:

```
from keras.models import load_model
from tkinter import *
import tkinter as tk
import win32gui
from PIL import ImageGrab, Image
import numpy as np

model = load_model('m1.h5')

def predict_digit(img):
    #resize image to 28x28 pixels
    img = img.resize((28,28))
    #convert rgb to grayscale
    img = img.convert('L')
    img = np.array(img)
    #reshaping to support our model input and normalizing
    img = img.reshape(1,28,28,1)
    img = img/255.0
    #predicting the class
    res = model.predict([img])[0]
    return np.argmax(res), max(res)

class App(tk.Tk):
    def __init__(self):
        tk.Tk.__init__(self)

        self.x = self.y = 0

        # Creating elements
        self.canvas = tk.Canvas(self, width=300, height=300, bg = "white", cursor="cross")
```

```python
        # Creating elements
        self.canvas = tk.Canvas(self, width=300, height=300, bg = "white", cursor="cross")
        self.label = tk.Label(self, text="Draw..", font=("Helvetica", 48))
        self.classify_btn = tk.Button(self, text = "Recognise", command =         self.classify_handwriting)
        self.button_clear = tk.Button(self, text = "Clear", command = self.clear_all)

        # Grid structure
        self.canvas.grid(row=0, column=0, pady=2, sticky=W, )
        self.label.grid(row=0, column=1,pady=2, padx=2)
        self.classify_btn.grid(row=1, column=1, pady=2, padx=2)
        self.button_clear.grid(row=1, column=0, pady=2)

        #self.canvas.bind("<Motion>", self.start_pos)
        self.canvas.bind("<B1-Motion>", self.draw_lines)

    def clear_all(self):
        self.canvas.delete("all")

    def classify_handwriting(self):
        HWND = self.canvas.winfo_id() # get the handle of the canvas
        rect = win32gui.GetWindowRect(HWND) # get the coordinate of the canvas
        im = ImageGrab.grab(rect)

        digit, acc = predict_digit(im)
        self.label.configure(text= str(digit)+', '+ str(int(acc*100))+'%')
```

```python
    def draw_lines(self, event):
        self.x = event.x
        self.y = event.y
        r=8
        self.canvas.create_oval(self.x-r, self.y-r, self.x + r, self.y + r, fill='black')

app = App()
mainloop()
```

Test.py:

```
#Importing the Keras libraries and packages
from tensorflow.keras.models import load_model
from PIL import Image #used for manipulating image uploaded by the user.
import numpy as np #used for numerrical analysis
model = load_model('mnistCNN.h5')
img = Image.open('digits/digit9.png').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
y_pred = model.predict(im2arr) #predicting the results
print(y_pred)
import numpy as np
print(np.argmax(y_pred, axis=1)) #printing our Labels from first 4 images
```

13.2 GITHUB & DEMO LINK

https://github.com/IBM-EPBL/IBM-Project-8633-1658926133

https://clipchamp.com/watch/dTb2Asvtp1d