A Novel Method for Handwritten Digit Recognition System

IBM Project

Submitted by

CHRISOLUS TIMONSINGH J	**142219104017**
INSUVAI V	**142219104042**
JENVIN SHIRLY R	**142219104046**
KAVIBHARATHI K	**142219104056**

in a partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

SRM VALLIAMMAI ENGINEERING COLLEGE

(AN AUTONOMOUS INSTITUTION)

SRM NAGAR, KATTANKULATHUR,

ANNA UNIVERSITY: CHENNAI 600 025

DEC 2022

ABSTRACT

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 analysed by the model and the detected result is returned on to UI.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
	ABSTRACT	ii
	LIST OF FIGURES	v
	LIST OF TABLES	vi
1	**INTRODUCTION**	1
	1.1 PROJECT OVERVIEW	2
	1.2 PURPOSE	3
2	**LITERATURE SURVEY**	4
	2.1 EXISTING PROBLEM	7
	2.2 REFERENCES	8
	2.3 PROBLEM STATEMENT DEFINITION	8
3	**IDEATION & PROPOSED SOLUTION**	9
	3.1 EMPATHY MAP CANVAS	10
	3.2 IDEATION & BRAINSTORMING	11
	3.3 PROPOSED SOLUTION	14
	3.4 PROBLEM SOLUTION FIT	15
4	**REQUIREMENT ANALYSIS**	16
	4.1 FUNCTIONAL REQUIREMENTS	17
	4.2 NON-FUNCTIONAL REQUIREMENTS	18
5	**PROJECT DESIGN**	21
	5.1 DATA FLOW DIAGRAM	22

	5.2 SOLUTION & TECHNICAL ARCHITECTURE	23
	5.3 USER STORIES	25
6	**PROJECT PLANNING & SCHEDULING**	26
	6.1 SPRINT PLANNING & ESTIMATION	27
	6.2 SPRINT DELIVERY SCHEDULE	27
	6.3 REPORT FROM JIRA	28
7	**CODING & SOLUTIONING**	29
	7.1 CNN MODEL	30
	7.2 THE USER INTERFACE	33
8	**RESULT**	37
	8.1 PERFORMANCE METRICS	38
9	**ADVANTAGES AND DISADVANTAGES**	39
10	**CONCLUSION**	41
11	**FUTURE SCOPE**	43
12	**APPENDIX**	45
	SOURCE CODE	46
	GITHUB LINK	55

LIST OF FIGURES

S.NO	FIG NO	TITLE	PAGE NO
1	3.1	EMPATHY MAP	10
2	3.2	BRAINSTORM	11
3	3.3	GROUP IDEAS	12
4	3.4	PRIORITIZE	13
5	3.5	PROBLEM SOLUTION FIT	15
6	5.1	DATA FLOW DIAGRAM	23
7	5.2	SOLUTION ARCHITECTURE	24
8	5.3	TECHNICAL ARCHITECTURE	25
9	6.1	BURNDOWN CHART	28
10	7.1	HOME PAGE(MAIN.HTML)	34
11	7.2	INDEX6.HTML	35
12	7.3	OUTPUT 1	36
13	7.4	OUTPUT 2	36

LIST OF TABLES

S.NO	TABLE NO	TITLE	PAGE NO
1	3.1	PROPOSED SOLUTION	14
2	5.1	USER STORIES	26
3	6.1	SPRINT PLANNING & ESTIMATION	28
4	6.2	SPRINT DELIVERY SCHEDULE	28

CHAPTER-1
INTRODUCTION

1.1 Project Overview

Handwritten digit recognition is the ability of a computer to recognize the human handwritten digits from different sources like images, papers, touch screens, etc, and classify them into 10 predefined classes (0-9). This has been a topic of boundless-research in the field of deep learning. Digit recognition has many applications like number plate recognition, postal mail sorting, bank check processing, etc. In Handwritten digit recognition, we face many challenges because of different styles of writing of different peoples as it is not an Optical character recognition. This research provides a comprehensive comparison between different machine learning and deep learning algorithms for the purpose of handwritten digit recognition. For this, we have used Convolutional Neural Network. CNN has built-in convolutional layer reduces the high dimensionality of images without losing its information. The main advantage of CNN compared to its predecessors is that it automatically detects the important features without any human supervision. The accuracy of any model is paramount as more accurate models make better decisions. The models with low accuracy are not suitable for real-world applications. Ex- For an automated bank cheque processing system where the system recognizes the amount and date on the check, high accuracy is very critical. If the system incorrectly recognizes a digit, it can lead to major damage which is not desirable. That’s why an algorithm with high accuracy is required in these real-world applications. We use Artificial Convolutional neural network 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

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. A handwriting recognition system handles formatting, performs correct segmentation into characters and finds the most plausible words. The applications of digit recognition include in postal mail sorting, bank check processing, form data entry.

CHAPTER-2
LITERATURE SURVEY

Literature Survey:

1. Handwritten Digit Recognition using Machine Learning

Hand-written character and digit recognition have been one of the most exigent and engrossing field of pattern recognition and image processing. The main aim of this paper is to demonstrate and represent the work which is related to hand-written digit recognition. The hand-written digit recognition is a very exigent task. 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. Advantages: After the completion of pre-processing stage and segmentation stage, the pre-processed images are represented in the form of a matrix which contains pixels of the images that are of very large size Disadvantage: The generative models can perform recognition driven segmentation. The method involves a relatively.

2. Diagonal based feature extraction for handwritten character recognition system using neural network

An off-line handwritten alphabetical character recognition system using multilayer feed forward neural network is described in the paper. A new method, called, diagonal based feature extraction is introduced for extracting the features of the handwritten alphabets. Fifty data sets, each containing 26 alphabets written by various people, are used for training the neural network and twenty different handwritten alphabets characters are used for testing. The proposed recognition system performs quite well yielding higher levels of recognition accuracy compared to the systems employing the conventional horizontal and vertical methods of feature extraction. This system will be suitable for converting handwritten documents into structural text form and recognizing handwritten names. Advantages: The main aim of feature extraction phase is to extract that pattern which is most pertinent for classification Disadvantage: Most importantly, it is not possible to speak to a machine in a natural way due to constraints such as out of vocabulary wards.

3. Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Models

The structural part of the optical models has been modelled with Markov chains, and a Multilayer Perceptron is used to estimate the emission probabilities. This paper also presents new techniques to remove slope and slant from handwritten text and to normalize the size of text images with supervised learning methods. Slope correction and size normalization are achieved by classifying local extrema of text contours with Multilayer Perceptron. Slant is also removed in a nonuniform way by using Artificial Neural Networks. Experiments have been conducted on offline handwritten text lines from the IAM database, and the recognition rates achieved, in comparison to the ones reported in the literature, are among the best for the same task. Advantages: The main aim of feature extraction phase is to extract that pattern which is most pertinent for classification. Disadvantage: As it is noisy to hear someone sitting next to us and talking to his machine. Moreover, anyone who wants to input confidential data to computer is not willing to do it in public places.

4. Recognition of handwritten similar Chinese characters by self-growing probabilistic decision-based neural networks

The self-growing probabilistic decision-based neural network (SPDNN) is a probabilistic type neural networks, which adopts a hierarchical network structure with nonlinear basis functions and a competitive credit-assignment scheme. Based on the SPDNN model, we constructed a three-stage recognition system. The prototype system demonstrates a successful utilisation of SPDNN to similar handwritten Chinese recognition on the public database CCL/HCCRI (5401 characters /spl times/200 samples). Regarding the performance, the experiments on the CCL/HCCRI database demonstrated a 90.12% of recognition accuracy with no rejection and 94.11% of accuracy with 6.7% rejection rates, respectively. Advantages: The difficult task is there are some handwritten digits that often run together or not fully connected. Numeral 5 is an example. But once these tasks have been carried out, the digits are available as individual items. But the digits are still indifferent sizes. Disadvantages: Background noise, cross-talk, accented speech and so on.

2.1 Existing Problem

Number recognition has numerous operations like number plate recognition, postal correspondence sorting, bank check processing, etc. In Handwritten number recognition, we face numerous challenges. Because of different styles of jotting of different peoples as it is not an Optic character recognition. This exploration provides a comprehensive comparison between different machine literacy and deep literacy algorithms for the purpose of handwritten number recognition.

2.2 References

***[1]** Rohan Sethi; Ila Kaushik, "Hand Written Digit Recognition using Machine Learning"*

***[2]** J. Pradeep; E. Srinivasan; S. Himavathi, "Diagonal based feature extraction for handwritten character recognition system using neural network"*

***[3]** S. España-Boquera; M.J. Castro-Bleda; J. Gorbe-Moya; F. Zamora-Martinez, "Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Models"*

***[4]** Hsin-Chia Fu; Y.Y. Xu, "Recognition of handwritten similar Chinese characters by self?growing probabilistic decision-based neural networks*

2.3 Problem Statement Definition

The handwritten digit recognition is the capability of computer applications to recognize the human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different shapes and sizes. The handwritten digit recognition system is a way to tackle this problem which uses the image of a digit and recognizes the digit present in the image. Convolutional Neural Network model created using PyTorch Library over the MNIST dataset to recognize handwritten digits.

CHAPTER-3
IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas:

The user hears that the digit recognition system works perfectly. The algorithm used is convolutional neural network. The digit recognition system is crucial for user need. The users are having many questions in their mind like does it work faster? Will it recognize the digits effectively? Will it be easier to access? The market offers a practical, adaptable, handy system that is convenient to use in various domains. Opinions differ from person to person. User opinions are described in the figure 3.1

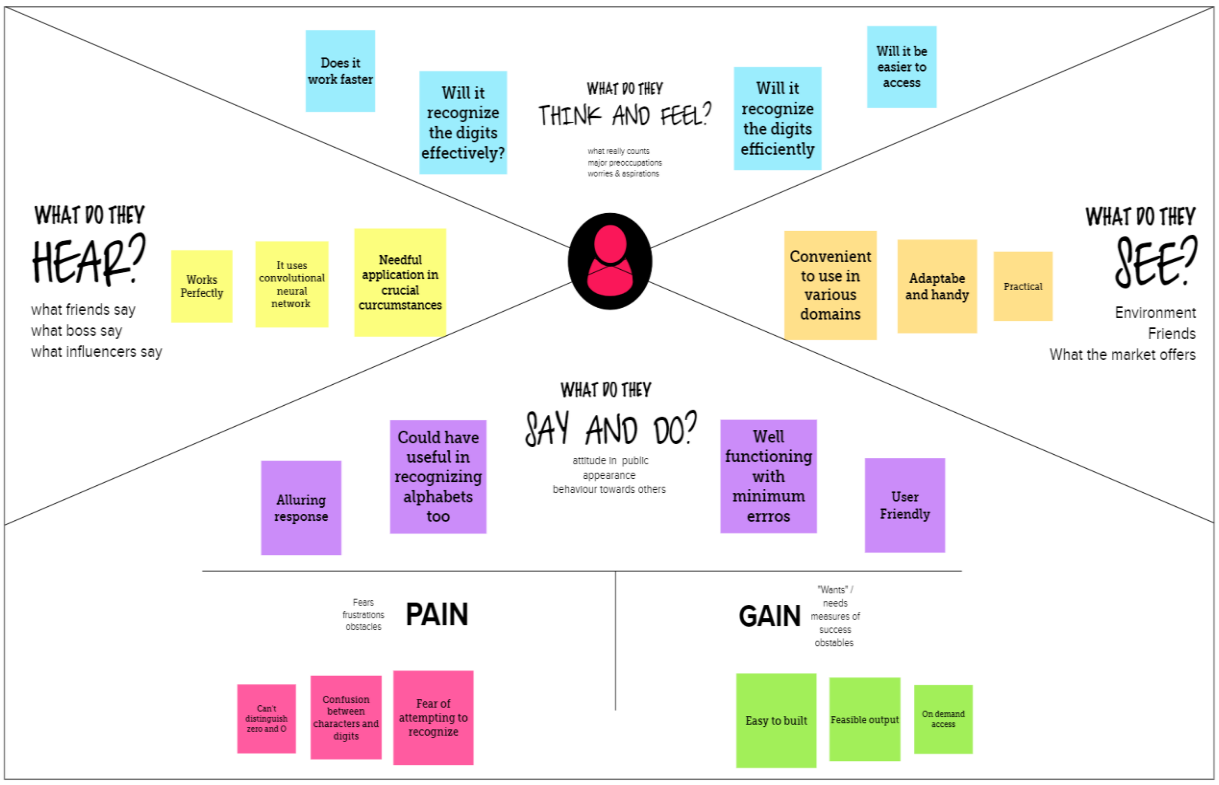
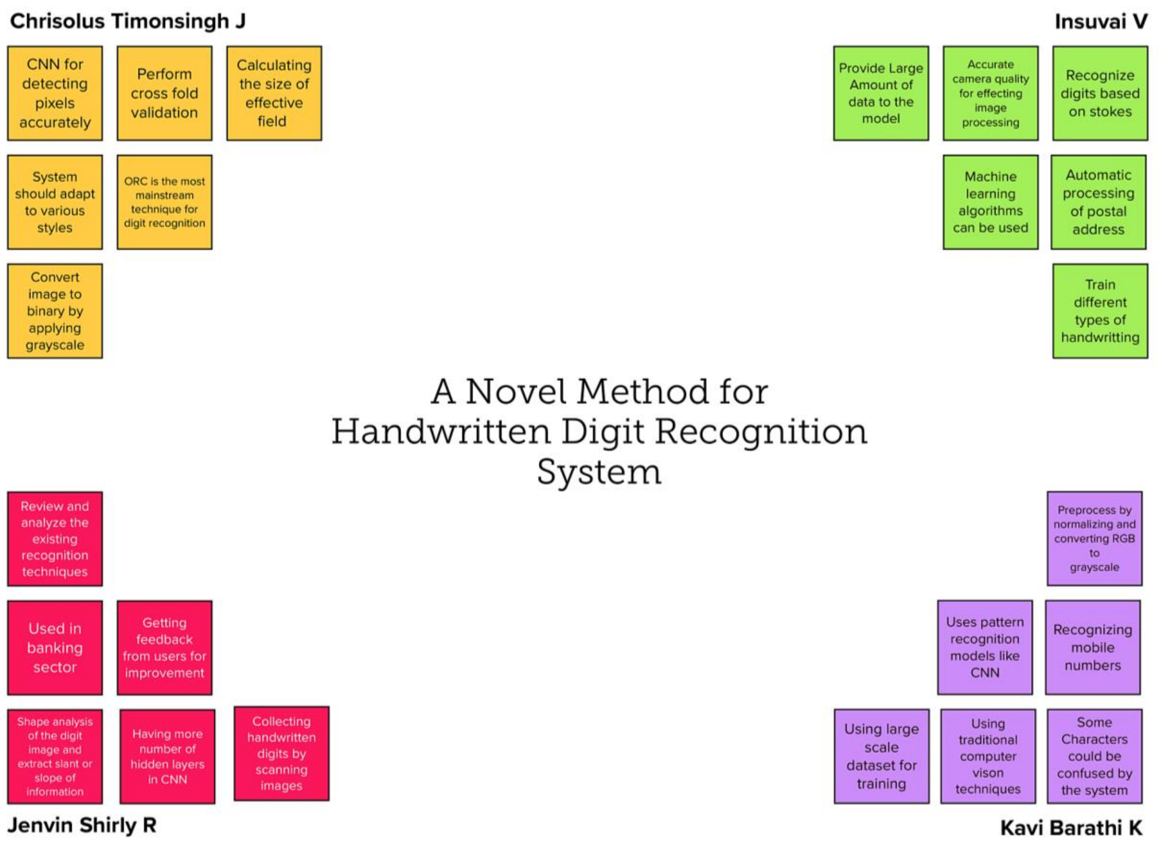


Fig 3.1 Empathy Map

3.2 Ideation & Brainstorming:

The concepts of CNN for precise pixel detection. Validation across folds will be done. It is employed to figure out the size of the effective field. The system ought to accommodate different fashions. The most used method for recognizing digits is ORC. Grayscale was employed by CNN to transform the image to binary. Chrisolus Timon provides these thoughts. The ideas like provide large amount of data to the model, accurate camera quality for effective image processing, train different types of handwriting are provided by Insuvai. The notion like have a greater number of hidden layers in CNN, getting feedback from users for improvement, review and analyse the exisiting recognition techniques are given by Jenvin Shirly. The concepts include employing sizable training datasets, conventional computer vision techniques, identifying mobile numbers, and using pattern recognition models like CNN. These suggestions come from Kavi Bharathi.



Chrisolus Timonsingh J
CNN for detecting pixels accurately
Perform cross fold validation
Calculating the size of effective field
System should adapt to various styles
ORC is the most mainstream technique for digit recognition
Convert image to binary by applying grayscale
Insuvai V
Provide Large Amount of data to the model
Accurate camera quality for effecting image processing
Recognize digits based on stokes
Machine learning algorithms can be used
Automatic processing of postal address
Train different types of handwritting
A Novel Method for Handwritten Digit Recognition System
Review and analyze the existing recognition techniques
Used in banking sector
Getting feedback from users for improvement
Shape analysis of the digit image and extract slant or slope of information
Having more number of hidden layers in CNN
Collecting handwritten digits by scanning images
Jenvin Shirly R
Preprocess by normalizing and converting RGB to grayscale
Uses pattern recognition models like CNN
Recognizing mobile numbers
Using large scale dataset for training
Using traditional computer vison techniques
Some Characters could be confused by the system
Kavi Barathi K

Fig 3.2 Brain storm

Machine learning, image processing, CNN, and deep learning are the technologies that can be employed in our project. The solutions we discussed include mobile number, bank sector, zip code, and postal address. Illegible handwriting, difficult assignment for machines, poor internet connectivity, and poor digital system quality are issues based on our projection.

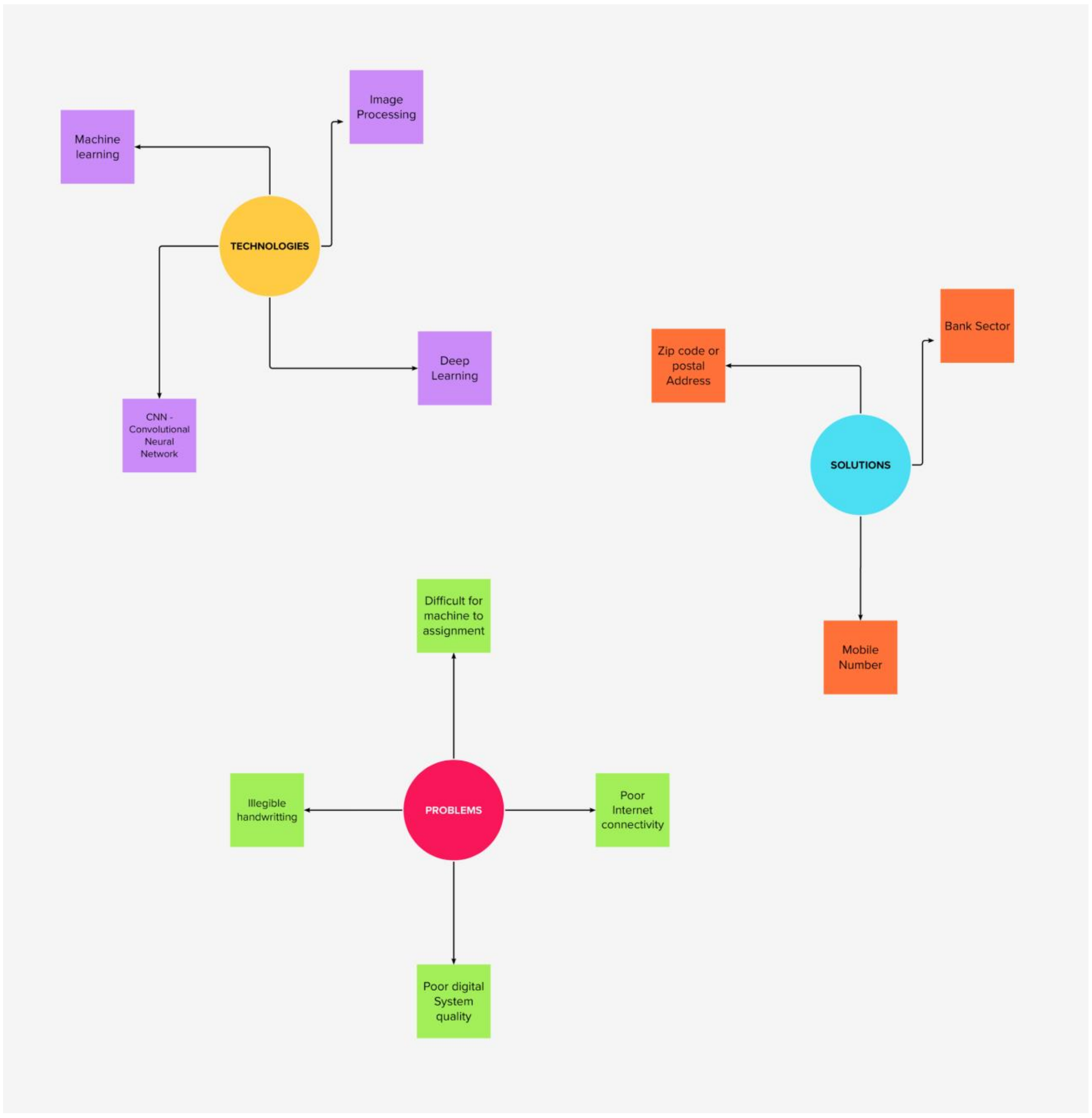

TECHNOLOGIES
Machine learning
Image Processing
Deep Learning
CNN - Convolutional Neural Network
SOLUTIONS
Zip code or postal Address
Bank Sector
Mobile Number
PROBLEMS
Difficult for machine to assignment
Illegible handwritting
Poor Internet connectivity
Poor digital System quality

Fig: 3.3 Group ideas

Our ideas on this grid to determine which ideas are important and which are feasible are mentioned in fig 3.4

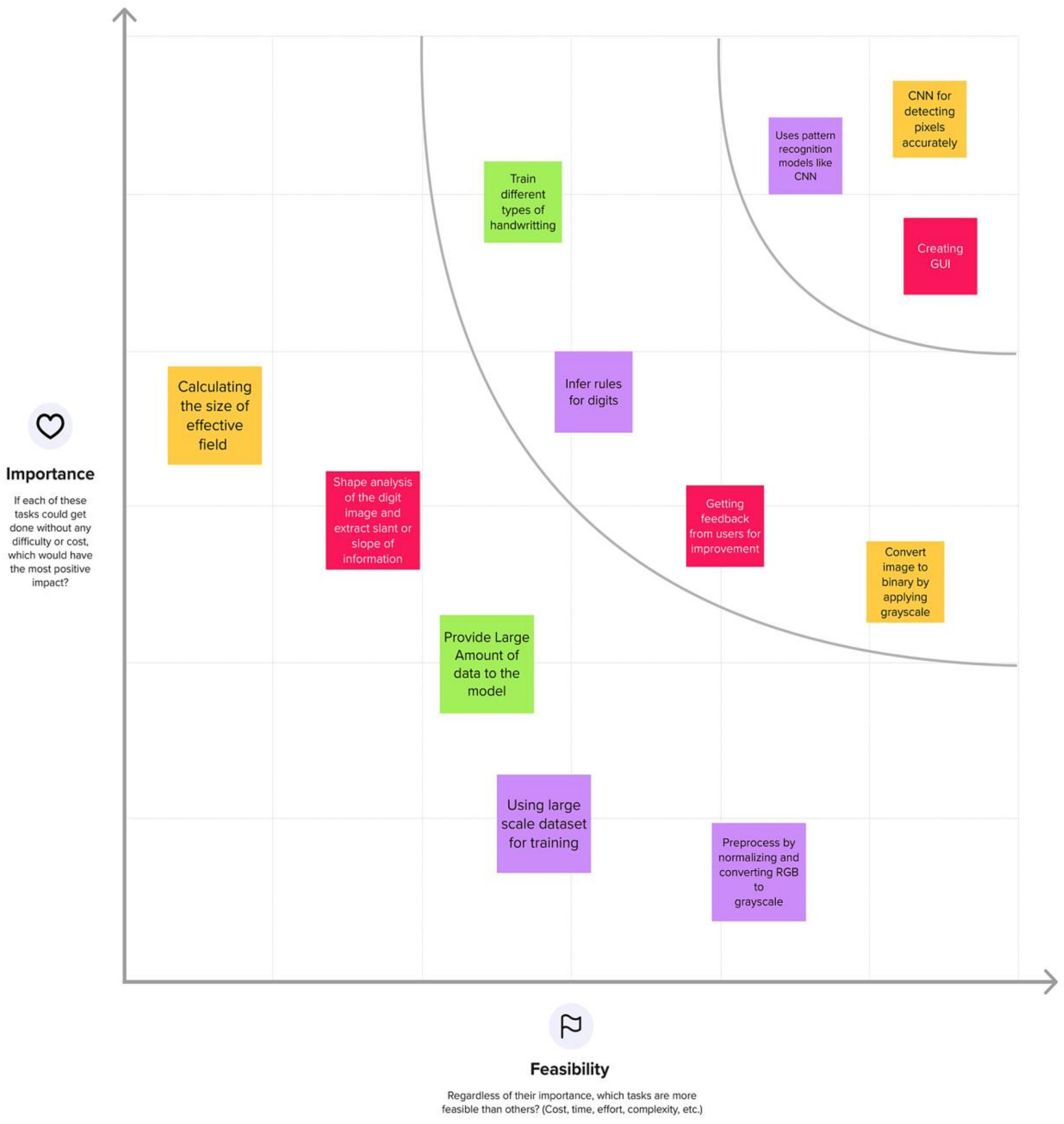

CNN for detecting pixels accurately
Uses pattern recognition models like CNN
Train different types of handwritting
Creating GUI
Infer rules for digits
Calculating the size of effective field
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Shape analysis of the digit image and extract slant or slope of information
Getting feedback from users for improvement
Convert image to binary by applying grayscale
Provide Large Amount of data to the model
Using large scale dataset for training
Preprocess by normalizing and converting RGB to grayscale
Feasibility
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

Fig: 3.4 Prioritize

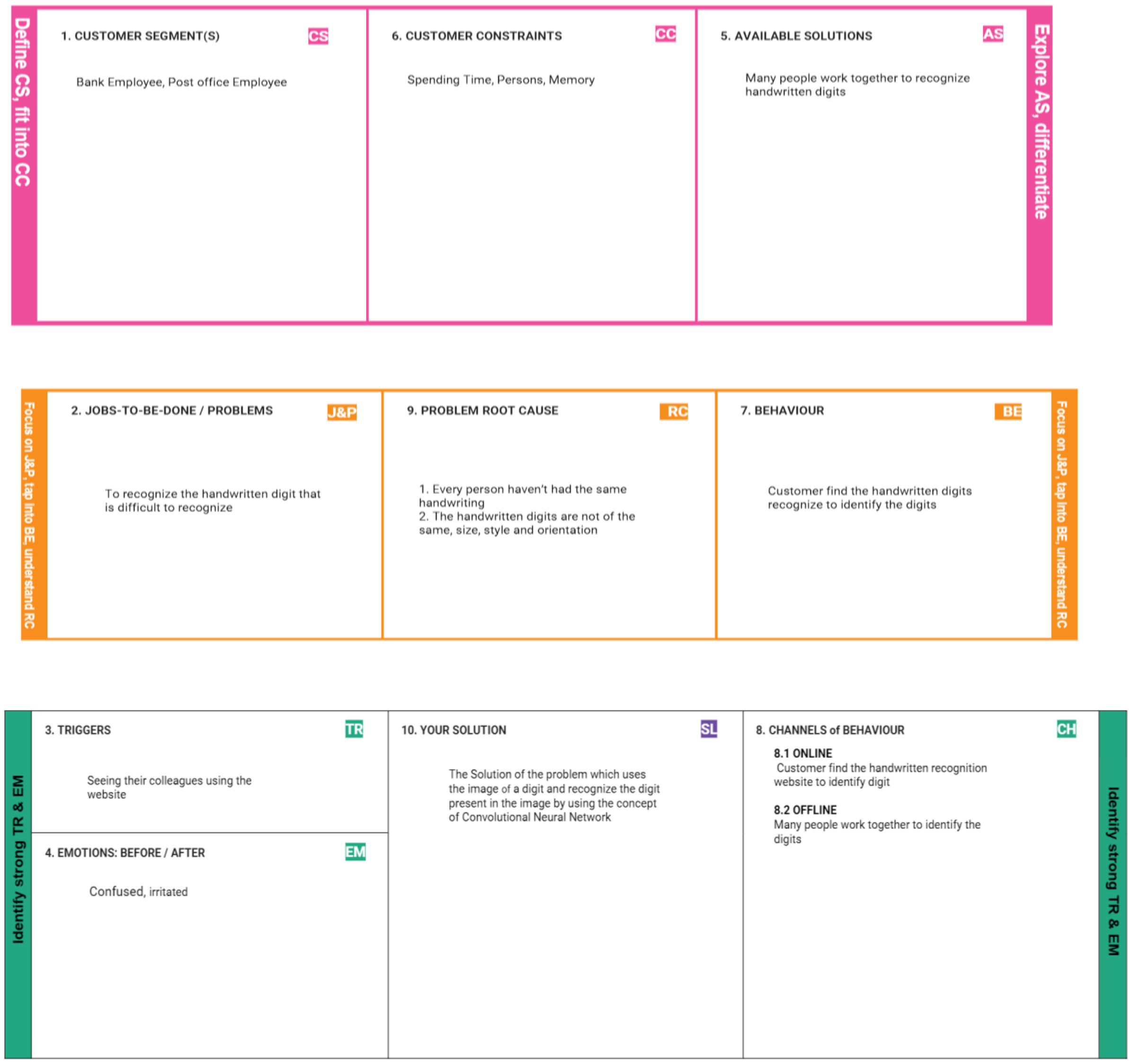
3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The Handwritten digits are not always of the same size, width, orientation and justified to margins as they from writing of person to person
2.	Idea / Solution description	By using MNIST dataset handwritten digits can be recognised. MNIST dataset contains 60000 training images of handwritten digits from Zero to Nine and Ten thousand images for testing.
3.	Novelty / Uniqueness	This system provides authentication for maintaining privacy of the users and user can store data.
4.	Social Impact / Customer Satisfaction	Postal department and courier services can easily find the digits written. Applications of handwriting recognition are numerous: reading postal addresses, bank check amounts, and forms
5.	Business Model (Revenue Model)	Used in Banking sector and Postal sector. In banking sectors, numerous handwritten numbers are involved. Our system reduces the human mistakes
6.	Scalability of the Solution	It recognises the handwritten digit in high level of accuracy.

Table 3.1 Proposed Solution

3.4 Problem Solution Fit

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem.



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Bank Employee, Post office Employee

6. CUSTOMER CONSTRAINTS CC

Spending Time, Persons, Memory

5. AVAILABLE SOLUTIONS AS

Many people work together to recognize handwritten digits

Explore AS, differentiate

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

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

To recognize the handwritten digit that is difficult to recognize

9. PROBLEM ROOT CAUSE RC

1. Every person haven't had the same handwriting
2. The handwritten digits are not of the same, size, style and orientation

7. BEHAVIOUR BE

Customer find the handwritten digits recognize to identify the digits

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

Identify strong TR & EM

3. TRIGGERS TR

Seeing their colleagues using the website

4. EMOTIONS: BEFORE / AFTER EM

Confused, irritated

10. YOUR SOLUTION SL

The Solution of the problem which uses the image of a digit and recognize the digit present in the image by using the concept of Convolutional Neural Network

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE
Customer find the handwritten recognition website to identify digit

8.2 OFFLINE
Many people work together to identify the digits

Identify strong TR & EM

Fig 3.5 Problem Solution Fit

CHAPTER -4
REQUIREMENT ANALYSIS

4.1 Functional Requirements:

Functional requirements are the details and instructions that dictate how software performs and behaves. Functional requirements can vary in behaviours, features, and protocols, depending on the user's industry. Functional requirements help software engineers determine the features that support a system to operate smoothly. User can ensure that software is easy for users to operate using functional requirements. Functional requirements can help you identify the features of a system to see where you may enhance functionality. Functional requirements increase the usability of the software. A software system may include a specific feature that makes the system more convenient for users to operate.

User Input:

User Input is one of the most important aspects of programming concepts. Every program should have some sort of user interaction, from getting a character's name for a game to asking for a password to log into a database. The model uses image as input for processing the functionalities such as training and testing. The input can be based the user convenience. The data is collected from the user in the model using uploads. It is possible to capture user input and set it as user attributes to create conditional flows.

Model:

Models are an estimation of how reality works. These are not just part of AI. Our brains use models every day. AI is the process of building models like this by using data, just like we do in everyday life. Models in AI, models are software that gives a prediction (or estimation) to an example, using data provided to it. The MNIST dataset for novel method of handwritten digit recognition should be trained using CNN to create a trained model.

When you look at everything that comprises an AI model it becomes clear that the analytics industry is too focused on predictive modelling tools and techniques.

There are a ton of data science frameworks and comparatively little innovation happening in decision automation. The result is that machine learning (the process for training a predictive analytic) ends up being disconnected from the rest of the decision-making process.

Prediction:

AI prediction is used to predict the occurrence of various things. AI Builder prediction models analyse patterns in historical data that you provide. Prediction models learn to associate those patterns with outcomes. Then, we use the power of AI to detect learned patterns in new data, and use them to predict future outcomes.AI prediction uses AI to predict future occurrences and future values from data. AI prediction is attracting attention both to avoid entirely personal predictions and to aim for more accurate predictions. The trained model has to be tested by using the test data provided by MNIST and the accuracy of the model should be above 90%.

Evaluation

The evaluation of artificial intelligence systems and components is crucial for the progress of the discipline. Ensure that the output produced by the model is correct.

4.2 Non-Functional Requirements:

Non-functional requirements or NFRs are a set of specifications that describe the system’s operation capabilities and constraints and attempt to improve its functionality. These are basically the requirements that outline how well it will operate including things like speed, security, reliability, data integrity Non-functional requirements specify the quality attributes of the system, hence their second name — quality attributes.

Usability:

A system can have adequate functionality, but inadequate usability because it is too difficult to use. A usability requirement specifies how easy the system must be to use. Usability is a non-functional requirement, because in its essence it doesn't specify parts of the system functionality, only how that functionality is to be perceived by the user, for instance how easy it must be to learn and how efficient it must be for carrying out user tasks.

The usability requirements must be tangible so that we are able to verify them and trace them during the development. They must also be complete so that if we fulfill them, we are sure that we get the usability we intend. Usability can predict digits with accuracy. The model can be used in bank check processing, data entry etc

Security:

Security is a non-functional requirement assuring all data inside the system or its part will be protected against malware attacks or unauthorized access. So, the non-functional requirements part will set up specific types of threats that functional requirements will address in more detail. If your security relies on specific standards and encryption methods, these standards don’t directly describe the behaviour of a system, but rather help engineers with implementation guides. How well are the system and its data protected against attacks. It ensures security as the uploaded image is not stored in any database.

Availability:

Availability describes how likely the system is accessible to a user at a given point in time. While it can be expressed as an expected percentage of successful requests, you may also define it as a percentage of time the system is accessible for operation during some time period. As you can see, these three metrics are closely connected. And more importantly, you should approach them together if you decide to document them as non-functional requirements for your system.

Available for web and mobile browsers NFR-6 Scalability Helps many individuals with low time consumption and high accuracy.

Reliability

Reliability specifies how likely the system or its element would run without a failure for a given period of time under predefined conditions. Reliability is defined as the probability that a product, system, or service will perform its intended function adequately for a specified period of time, or will operate in a defined environment without failure. Example - The system must perform without failure in 95 percent of use cases during a month. Can process confidential information without data leakage as the data is never stored in any database. Performance improvement in fast prediction. We use CNN algorithm for accurate prediction

Maintainability

Maintainability defines the time required for a solution or its component to be fixed, changed to increase performance or other qualities, or adapted to a changing environment. Like reliability, it can be expressed as a probability of repair during some time. For example, if you have 75 percent maintainability for 24 hours, this means that there’s a 75 percent chance the component can be fixed in 24 hours. Maintainability is often measured with a metric like MTTRS — the mean time to restore the system.

Compactability:

Compatibility, as an additional aspect of portability, defines how a system can coexist with another system in the same environment. For instance, software installed on an operating system must be compatible with its firewall or antivirus protection. Portability and compatibility are established in terms of operating systems, hardware devices, browsers, software systems, and their versions. For now, a cross-platform, cross-browsing, and mobile-responsive solution is a common standard for web applications.

CHAPTER-5
PROJECT DESIGN

5.1 Data Flow Diagram

A data flow diagram is a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement. They are often elements of a formal methodology such as Structured Systems Analysis and Design Method. Superficially, DFDs can resemble flow charts or Unified Modelling Language, but they are not meant to represent details of software logic.

In this Data Flow Diagram, MNIST dataset is pre-processed and divided into two for testing and training. Then those data is sent to CNN algorithm and for evaluation. And the evaluated data provide the result that will be viewed by the user and pre-processed by using the trained model which are mentioned in fig 5.1 below.

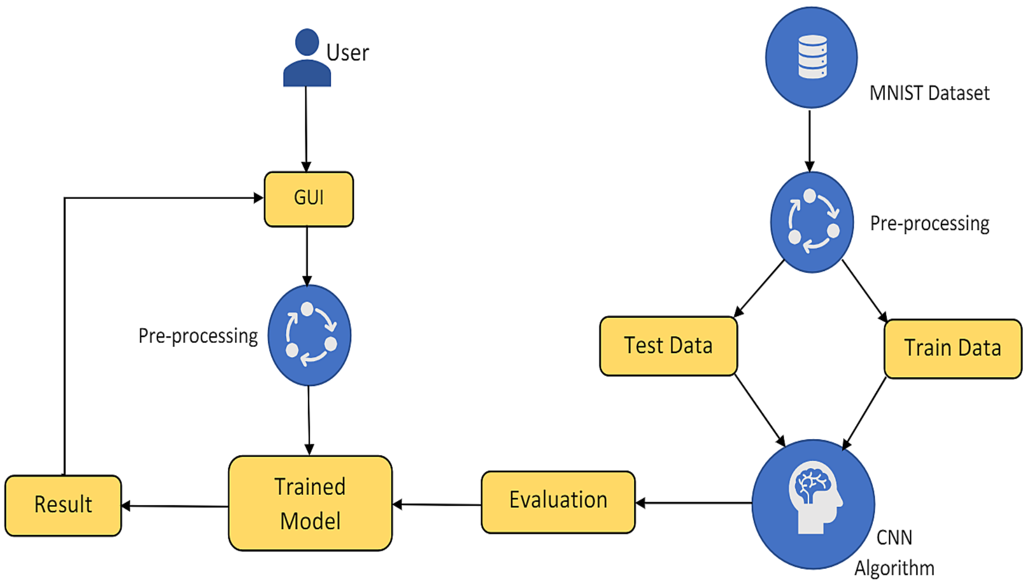


Fig 5.1 Data Flow Diagram

5.2 Solution & Technical Architecture

Solution architecture is the process of developing solutions based on predefined processes, guidelines and best practices with the objective that the developed solution fits within the enterprise architecture in terms of information architecture, system portfolios, integration requirements and many more.

In this Solution Architecture, Image data is pre-process' and trains then sends to DL algorithm and it tests the data. After this Both trained data and tested data are send for evaluation. Then the input is provided to that model and the model is predicted through UI by the user which are mentioned in fig 5.2 below.

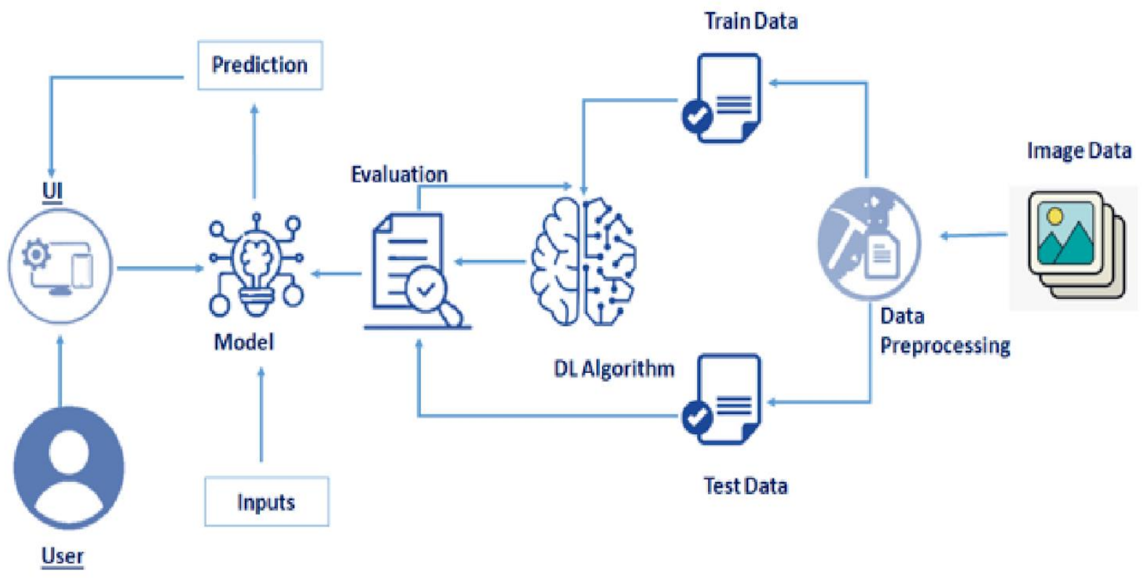


Fig 5.2 Solution Architecture

The technical architecture describes the infrastructure required to support applications, operations, and reporting requirements. A framework for building an enterprise including networking, hardware, operating systems, database management systems, and application development standards.

In this technical architecture, user provide both the input and MNIST data and they are classified after pre-processing, then it trains and evaluates the model to provide accurate result which are mentioned in fig 5.3 below.

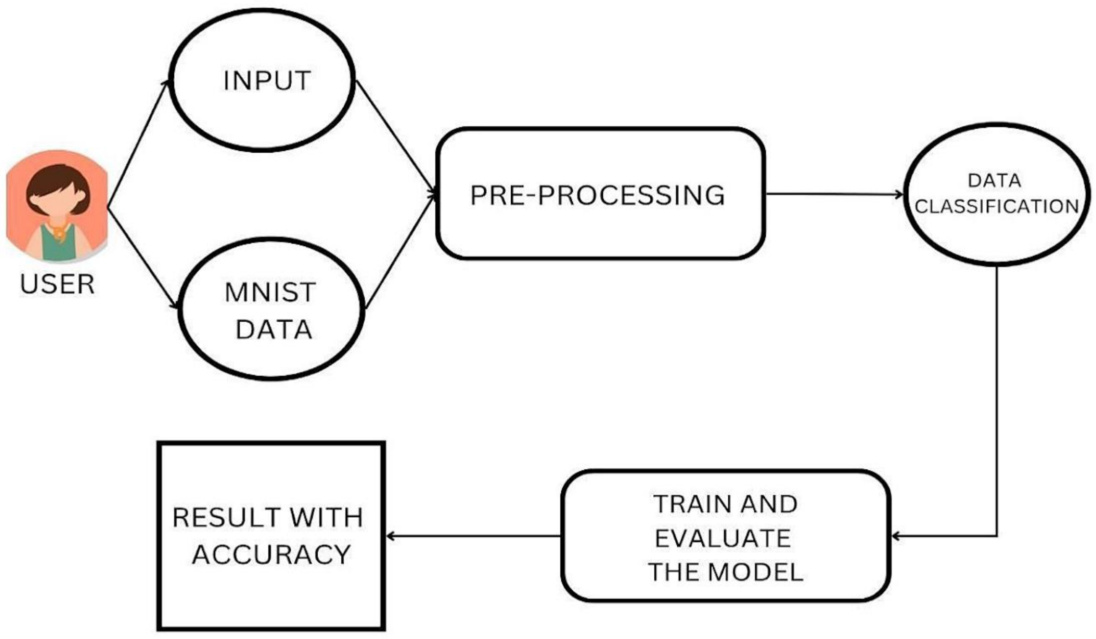
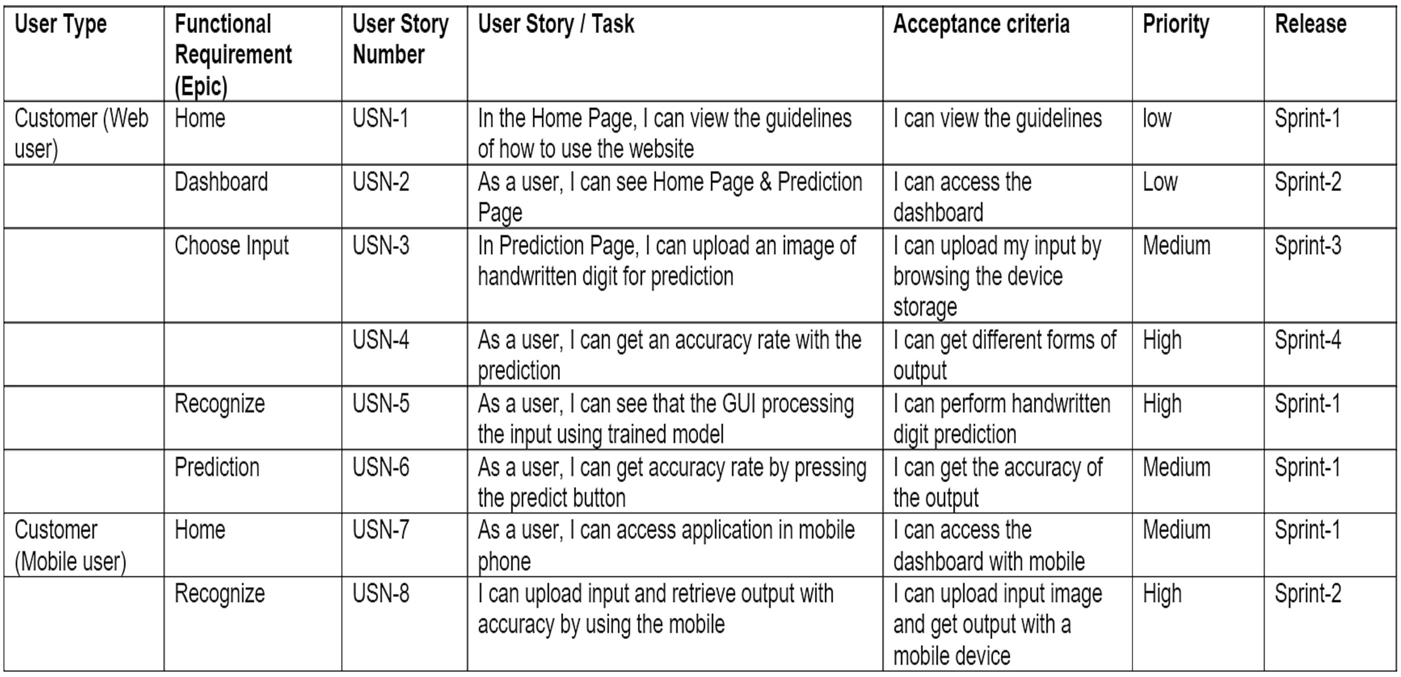


Fig 5.3 Technical Architecture

5.3 User Stories

A user story is an informal, general explanation of a software feature written from the perspective of the end user. Its purpose is to articulate how a software feature will provide value to the customer.



User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Home	USN-1	In the Home Page, I can view the guidelines of how to use the website	I can view the guidelines	low	Sprint-1
	Dashboard	USN-2	As a user, I can see Home Page & Prediction Page	I can access the dashboard	Low	Sprint-2
	Choose Input	USN-3	In Prediction Page, I can upload an image of handwritten digit for prediction	I can upload my input by browsing the device storage	Medium	Sprint-3
		USN-4	As a user, I can get an accuracy rate with the prediction	I can get different forms of output	High	Sprint-4
	Recognize	USN-5	As a user, I can see that the GUI processing the input using trained model	I can perform handwritten digit prediction	High	Sprint-1
	Prediction	USN-6	As a user, I can get accuracy rate by pressing the predict button	I can get the accuracy of the output	Medium	Sprint-1
Customer (Mobile user)	Home	USN-7	As a user, I can access application in mobile phone	I can access the dashboard with mobile	Medium	Sprint-1
	Recognize	USN-8	I can upload input and retrieve output with accuracy by using the mobile	I can upload input image and get output with a mobile device	High	Sprint-2

Table 5.1 User Stories

CHAPTER-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 & pre processing	USN-1	As a user, I can upload any kind of image with the pre-processing step is involved in it.	12	High	Insuvai V
Sprint-1		USN-2	As a user, I can upload the image in any resolution.	8	Medium	
Sprint-2	Building the model	USN-3	As a user, I can make use of the ML model which provides high accuracy of recognized handwritten digit	8	High	Jenvin Shirly R
Sprint-2		USN-4	As a user, I can feed the handwritten digit image to the model for recognizing the digit.	6	Medium	
Sprint-2		USN-5	As a user, I can get the digit recognized with at most accurate	7	Medium	
Sprint-3	Building User Interface Application	USN-6	As a user, I will upload the handwritten digit image to the application by using the upload option provided through the UI	10	High	Chrisolus J
Sprint-3		USN-7	As a user, I can see the predicted / recognized digits in the application	10	High	
Sprint-4	Train and deployment of model in IBM Cloud	USN-8	As a user, I can access the web application and make the use of the product from anywhere	20	High	Kavi Barathi B

Table 6.1 Sprint Planning & Estimation

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	25 Oct 2022	31 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	1 Nov 2022	06 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	08 Nov 2022	13 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

Table 6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

Velocity:

Imagine we have a 6-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 = sprint duration/Velocity

= 20/6

= 3.33

Burndown Chart:

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

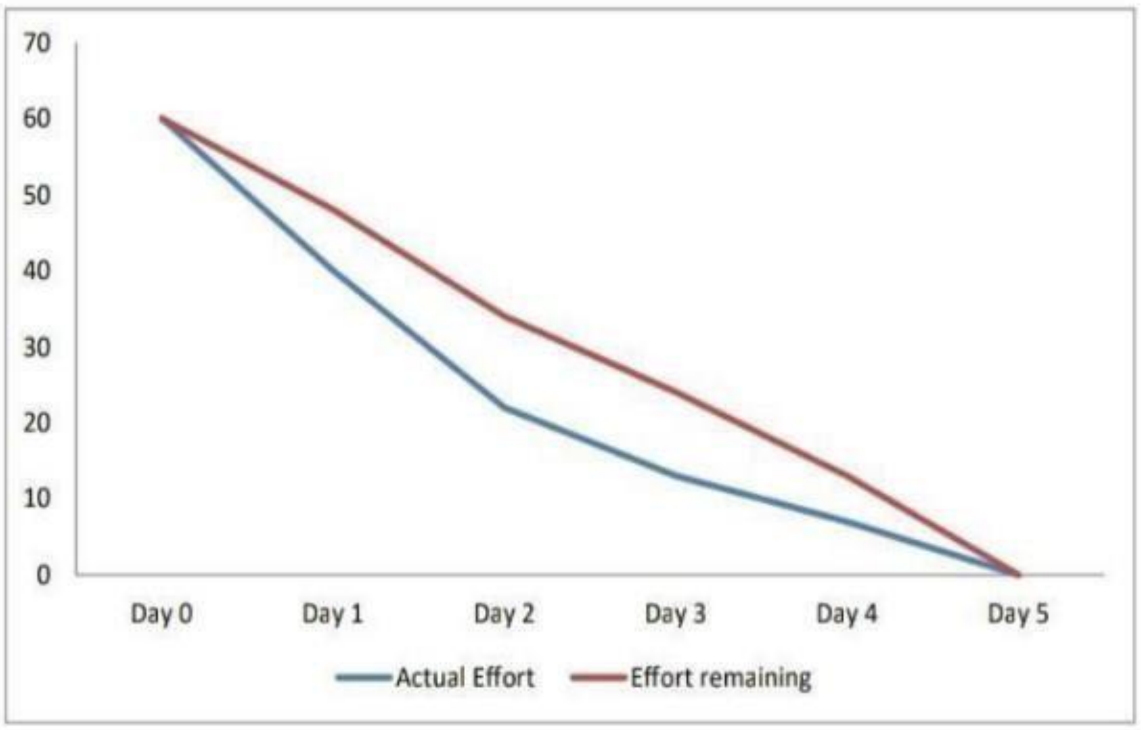


Fig 6.3: Burndown Chart

CHAPTER 7
CODING & SOLUTIONING

7.1 CNN Model

The ML model for handwritten digit recognition is done by using Convolutional Neural Network (CNN). Convolution involves trying every possible match for a feature in a image. This process of Convolution is called a Filtering. It involves the following

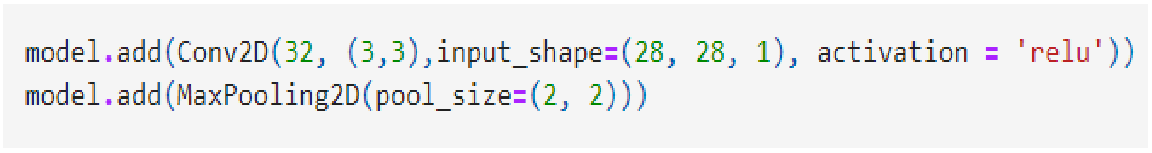
1. Line up the features and image patch
2. Multiply each image pixel by corresponding feature pixel
3. Add them up and divide by total number of pixels in the feature

The images in MNIST datasets are of dimension 28x28 which is filtered using 32 features to produce a stack of 32 filtered images with the dimension 26x26..

Pooling layer is placed after the first Conv2D layer. This layer shrinks the filtered image dimension while preserving the features. This is done by following the steps below

1. Choosing a window of size 2 or 3
2. Walk the window through the filtered image
3. Pick the maximum value from each window

After the application of pooling, the dimension of the filtered images becomes half

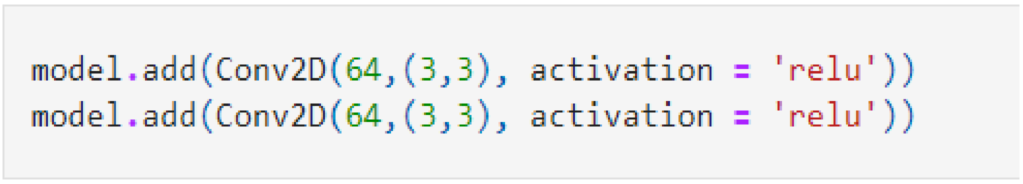


```
model.add(Conv2D(32, (3,3),input_shape=(28, 28, 1), activation = 'relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
```

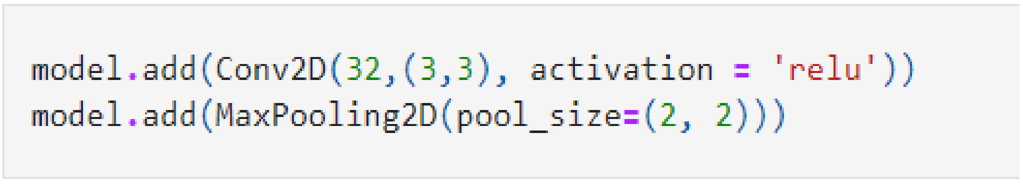
Normalization takes place after every layer where all the negative values become 0. This prevents the machine from reading unwanted features

Before the neural link reaches the output layer, the data is flattened and dense layers are added. This dense layer produces an array of length 10 as output. The dense layer can also be stacked recursively to improve the accuracy. Finally, the prediction is produced as an array of length 10 that represents 0-9. The index with the highest value will be taken as the predicted digit

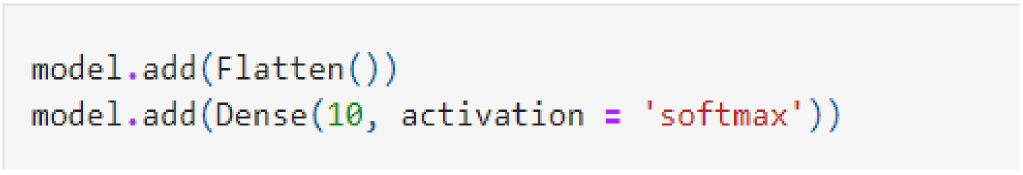
The above-mentioned layers can be stacked is a sequence repeatedly to form hidden layers. With increase in number of hidden layers, the accuracy of the model gets better.



```
model.add(Conv2D(64,(3,3), activation = 'relu'))
model.add(Conv2D(64,(3,3), activation = 'relu'))
```

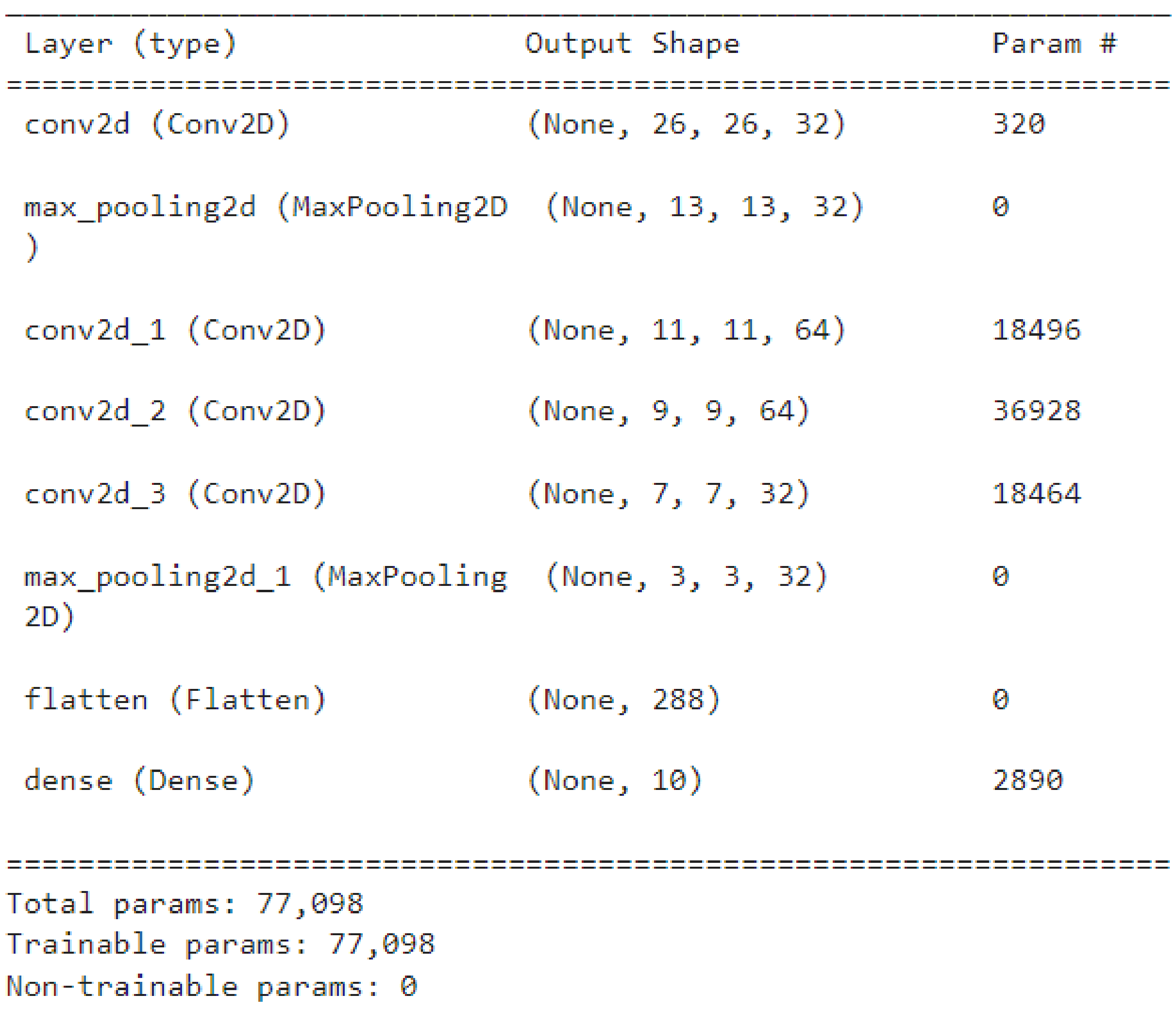


```
model.add(Conv2D(32,(3,3), activation = 'relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
```



```
model.add(Flatten())
model.add(Dense(10, activation = 'softmax'))
```

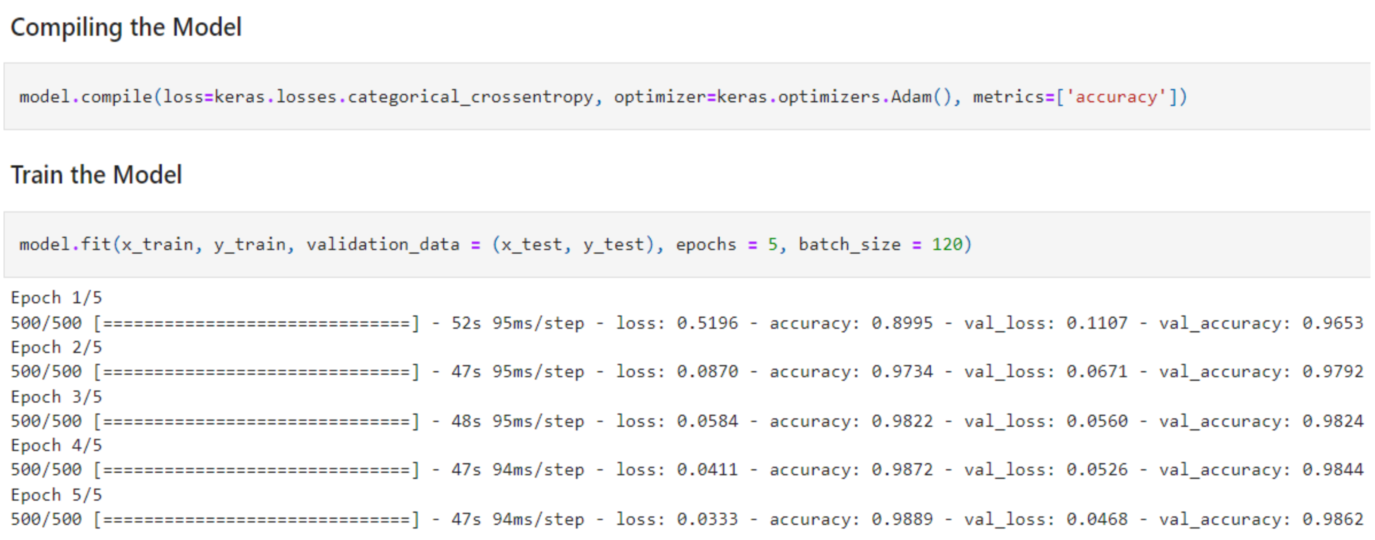
The Model summary of the CNN model used in the current project is as follows



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 32)        320

 max_pooling2d (MaxPooling2D  (None, 13, 13, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 11, 11, 64)        18496

 conv2d_2 (Conv2D)           (None, 9, 9, 64)          36928

 conv2d_3 (Conv2D)           (None, 7, 7, 32)          18464

 max_pooling2d_1 (MaxPooling  (None, 3, 3, 32)         0
 2D)

 flatten (Flatten)           (None, 288)               0

 dense (Dense)               (None, 10)                2890

=================================================================
Total params: 77,098
Trainable params: 77,098
Non-trainable params: 0
```

This model is then trained with epoch of 5 and a batch size of 120 using the 60,000-dataset provided by MNIST

The model is complied using an optimizer called as Adam and trained



Compiling the Model

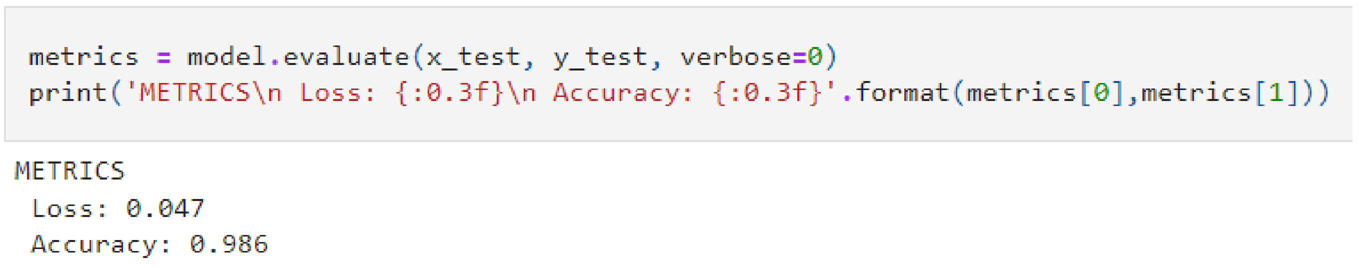
```
model.compile(loss=keras.losses.categorical_crossentropy, optimizer=keras.optimizers.Adam(), metrics=['accuracy'])
```

Train the Model

```
model.fit(x_train, y_train, validation_data = (x_test, y_test), epochs = 5, batch_size = 120)
```

```
Epoch 1/5
500/500 [==============================] - 52s 95ms/step - loss: 0.5196 - accuracy: 0.8995 - val_loss: 0.1107 - val_accuracy: 0.9653
Epoch 2/5
500/500 [==============================] - 47s 95ms/step - loss: 0.0870 - accuracy: 0.9734 - val_loss: 0.0671 - val_accuracy: 0.9792
Epoch 3/5
500/500 [==============================] - 48s 95ms/step - loss: 0.0584 - accuracy: 0.9822 - val_loss: 0.0560 - val_accuracy: 0.9824
Epoch 4/5
500/500 [==============================] - 47s 94ms/step - loss: 0.0411 - accuracy: 0.9872 - val_loss: 0.0526 - val_accuracy: 0.9844
Epoch 5/5
500/500 [==============================] - 47s 94ms/step - loss: 0.0333 - accuracy: 0.9889 - val_loss: 0.0468 - val_accuracy: 0.9862
```

The trained model is tested and observed that it predicts the digits with 98.6 % accuracy



```
metrics = model.evaluate(x_test, y_test, verbose=0)
print('METRICS\n Loss: {:0.3f}\n Accuracy: {:0.3f}'.format(metrics[0],metrics[1]))
```

```
METRICS
 Loss: 0.047
 Accuracy: 0.986
```

7.2 The User Interface

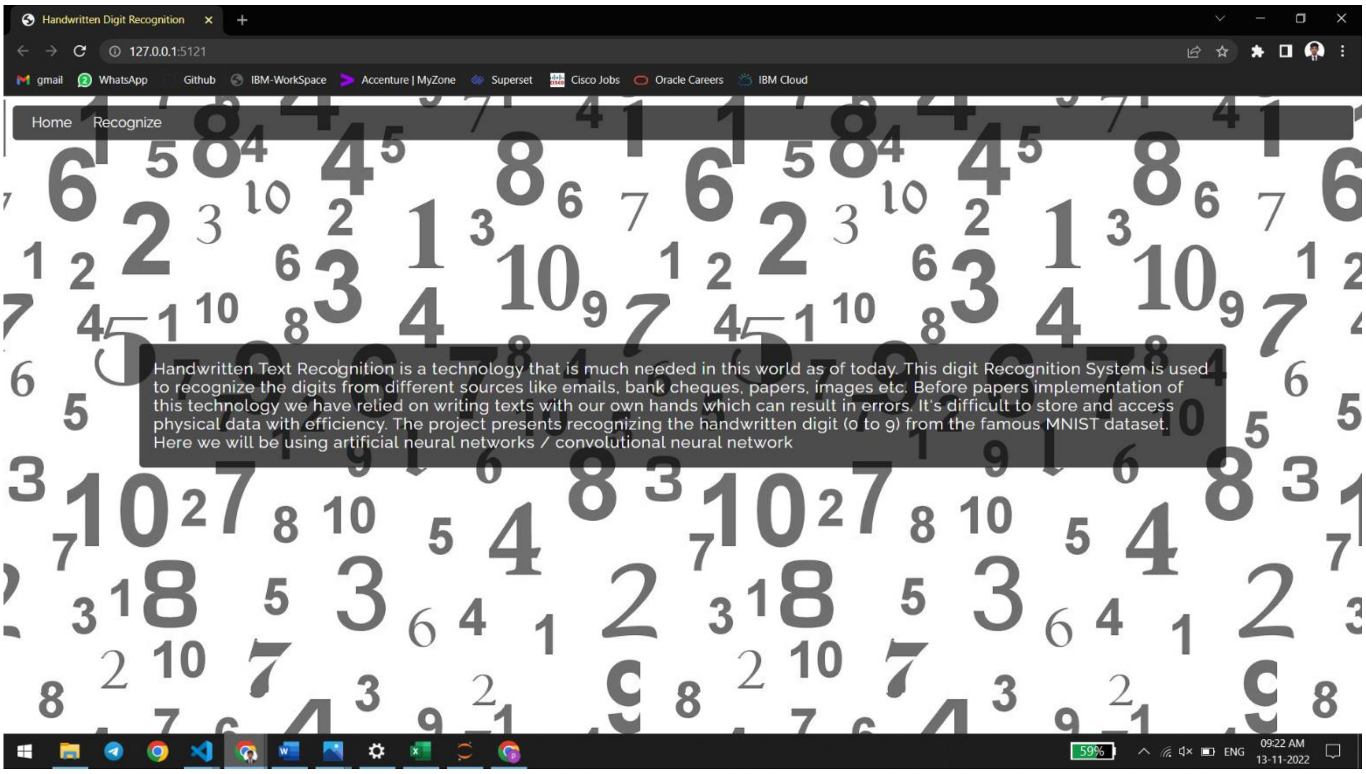
The UI is built using HTML, CSS and JavaScript. The backend of the application is handled by Flask, a python framework for backend application development. The application consists of two pages,

1. main.html
2. index6.html

The main.html is the home page which contains a brief description about Handwritten Digit Recognition

The index6.html is the prediction page where the users could input the image that contains a handwritten digit. This page displays the predicted output to the user

Home page (main.html)



Handwritten Text Recognition is a technology that is much needed in this world as of today. This digit Recognition System is used to recognize the digits from different sources like emails, bank cheques, papers, images etc. Before papers implementation of this technology we have relied on writing texts with our own hands which can result in errors. It's difficult to store and access physical data with efficiency. The project presents recognizing the handwritten digit (0 to 9) from the famous MNIST dataset. Here we will be using artificial neural networks / convolutional neural network

Fig 7.1 Home page(main.html)

Index6.html

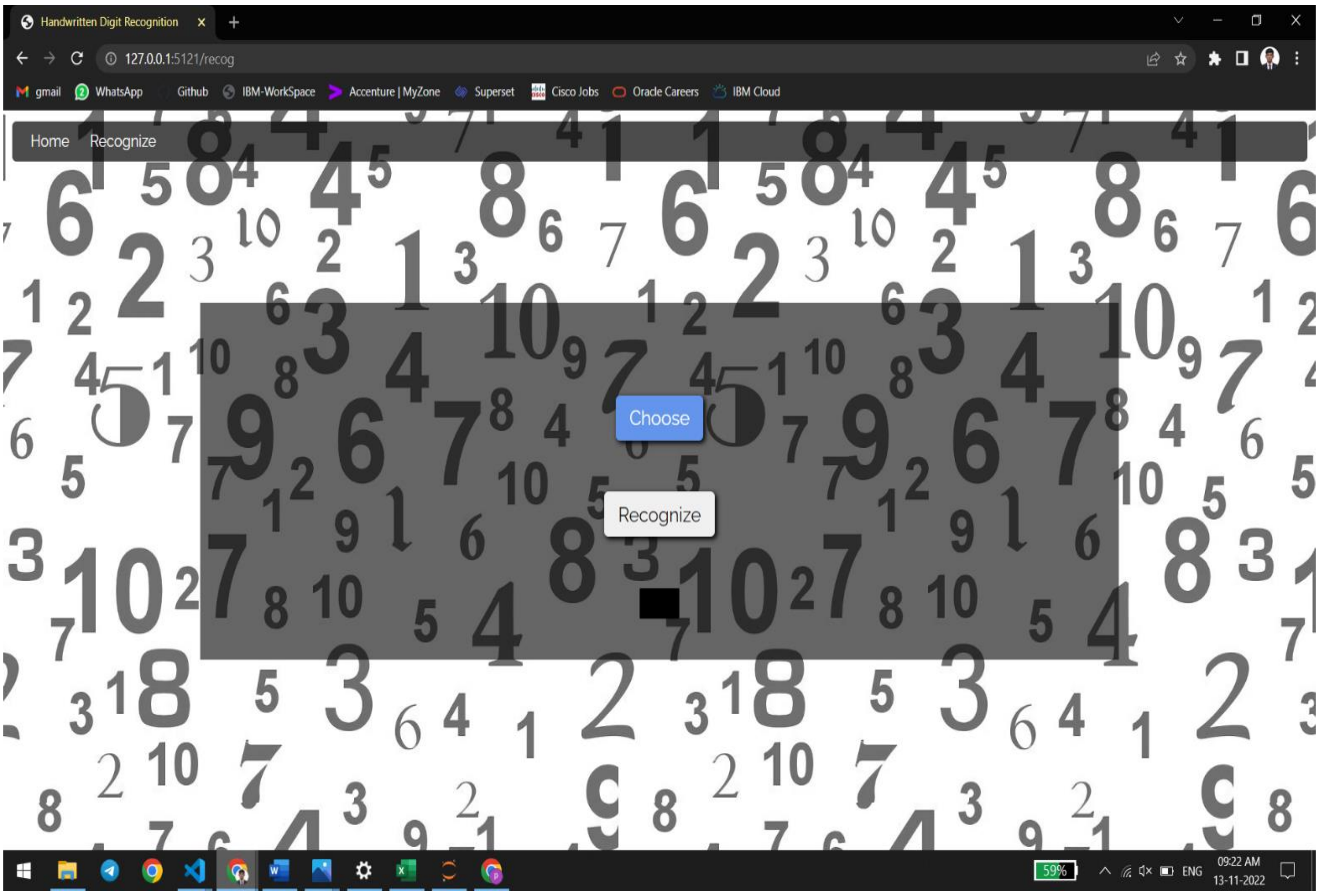
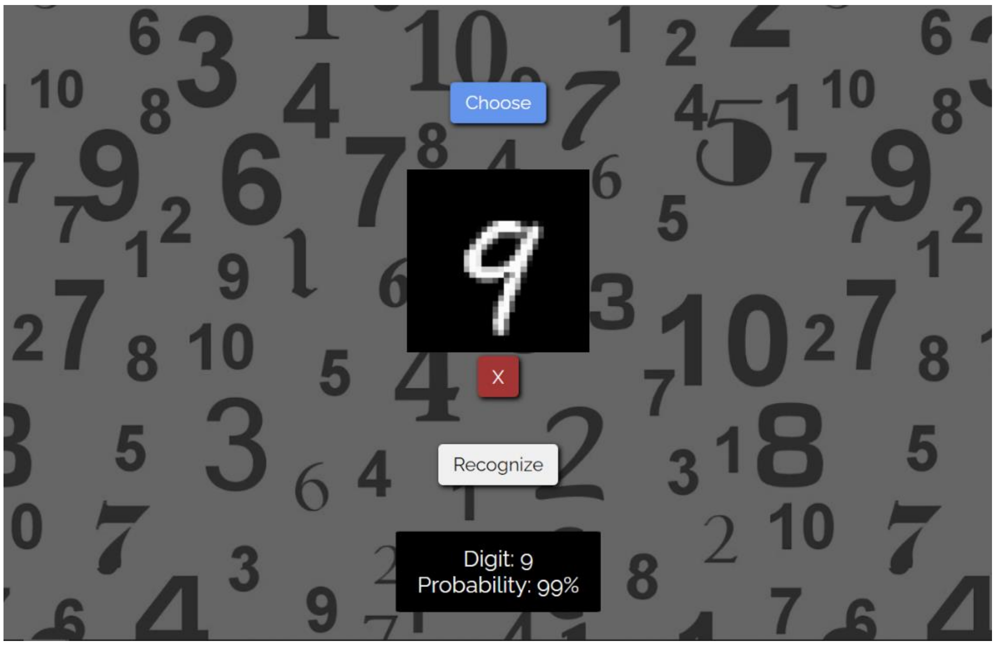


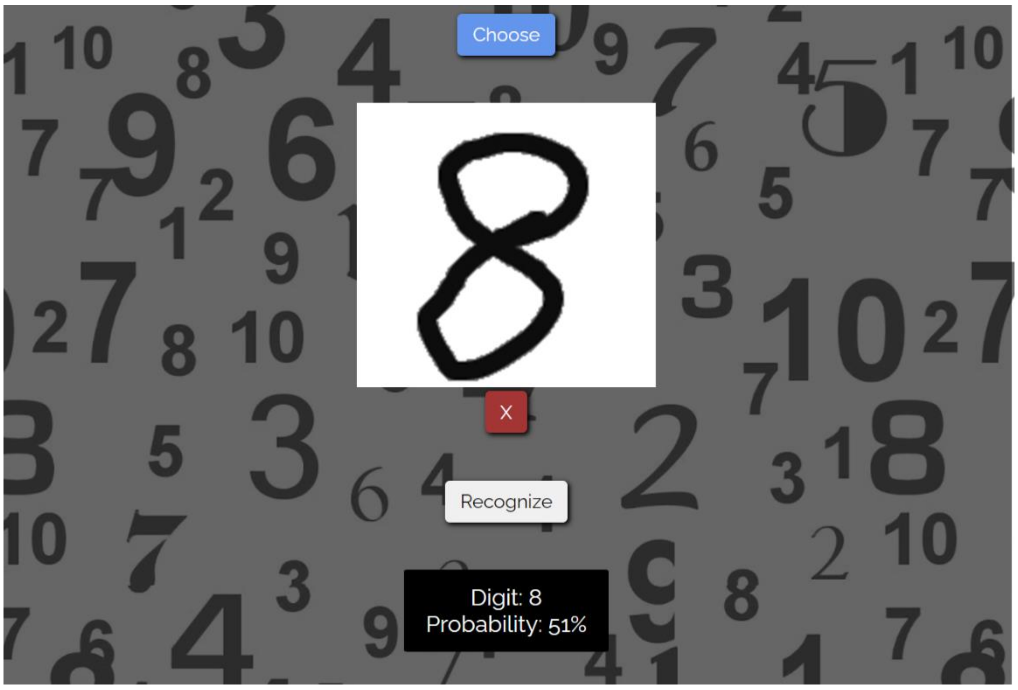
Fig 7.2 Index6.html

Outputs



Digit: 9
Probability: 99%

Fig 7.3 Output 1



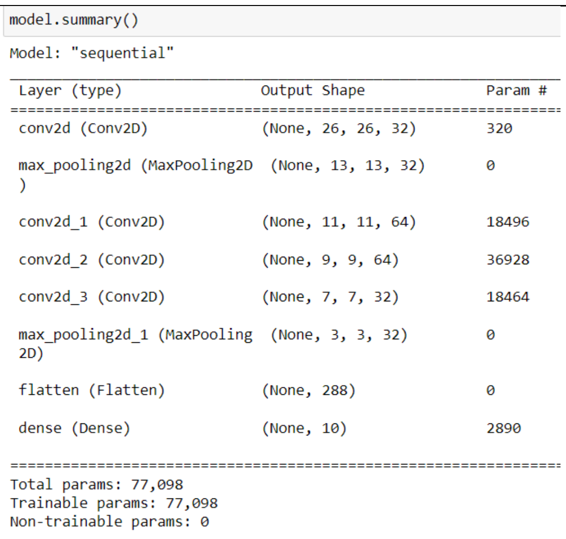
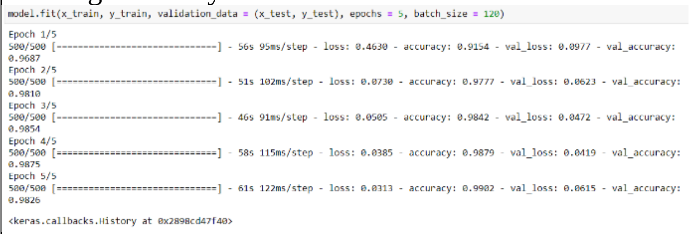
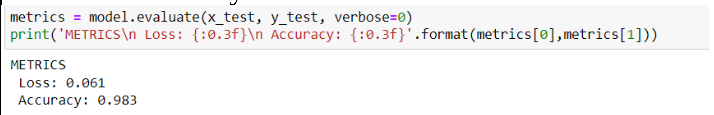
Digit: 8
Probability: 51%

Fig 7.4 Output 2

CHAPTER-8
RESULT

8.1 Performance Metrics

Evaluating the performance of a Machine learning model is one of the important steps while building an effective ML model. The performance metrics help us understand how well our model has performed for the given data. In this way, we can improve the model's performance by tuning the hyper-parameters. Each ML model aims to generalize well on unseen/new data, and performance metrics help determine how well the model generalizes on the new dataset. In Handwritten Digit Recognition accuracy is the main metric that should be considered while building the model. The model developed in this proposed system predicts the digit with an average accuracy of 98.6%

S.No.	**Parameter**	**Values**	**Screenshot**
1.	Model Summary	**Layer 1:** Conv2D (26,26,32) **Layer 2:** Pooling (13,13,32) **Layer 3:** Conv2D (11,11,64) **Layer 4:** Conv2D (9,9,64) **Layer 5:** Conv2D (7,7,32) **Layer 6:** Conv2D (3,3,32) **Layer 7:** Flatten (288) **Layer 8:** Dense (10)	
2.	Accuracy	Training Accuracy – 98.26 % Validation Accuracy – 98.3 5	Training Accuracy:  Validation Accuracy: 

```
model.summary()

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 32)        320

 max_pooling2d (MaxPooling2D  (None, 13, 13, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 11, 11, 64)        18496

 conv2d_2 (Conv2D)           (None, 9, 9, 64)          36928

 conv2d_3 (Conv2D)           (None, 7, 7, 32)          18464

 max_pooling2d_1 (MaxPooling  (None, 3, 3, 32)         0
 2D)

 flatten (Flatten)           (None, 288)               0

 dense (Dense)               (None, 10)                2890

=================================================================
Total params: 77,098
Trainable params: 77,098
Non-trainable params: 0
```

```
model.fit(x_train, y_train, validation_data = (x_test, y_test), epochs = 5, batch_size = 120)

Epoch 1/5
500/500 [==============================] - 56s 95ms/step - loss: 0.4630 - accuracy: 0.9154 - val_loss: 0.0977 - val_accuracy:
0.9687
Epoch 2/5
500/500 [==============================] - 51s 102ms/step - loss: 0.0730 - accuracy: 0.9777 - val_loss: 0.0623 - val_accuracy:
0.9810
Epoch 3/5
500/500 [==============================] - 46s 91ms/step - loss: 0.0505 - accuracy: 0.9842 - val_loss: 0.0472 - val_accuracy:
0.9854
Epoch 4/5
500/500 [==============================] - 58s 115ms/step - loss: 0.0385 - accuracy: 0.9879 - val_loss: 0.0419 - val_accuracy:
0.9875
Epoch 5/5
500/500 [==============================] - 61s 122ms/step - loss: 0.0313 - accuracy: 0.9902 - val_loss: 0.0615 - val_accuracy:
0.9826

<keras.callbacks.History at 0x2898cd47f40>
```

```
metrics = model.evaluate(x_test, y_test, verbose=0)
print('METRICS\n Loss: {:0.3f}\n Accuracy: {:0.3f}'.format(metrics[0],metrics[1]))

METRICS
 Loss: 0.061
 Accuracy: 0.983
```

CHAPTER-9
ADVANTAGES AND DISADVANTAGES

Advantages

The system not only produces a classification of the digit but also a rich description of the instantiation parameters which can yield information such as the writing style. The generative models can perform recognition driven segmentation. The method involves a relatively small number of parameters and hence training is relatively easy and fast and unlike many other recognition schemes, it does not rely on some form of pre-normalization of input images, but can handle arbitrary scaling, translations and a limited degree of image rotation. Compare to other research methods, this method focuses on which classifier works better by improving the accuracy of classification models by more than 99%. A CNN model is able to give accuracy of about 98.3%. It automatically detects the important features without any human supervision.

Disadvantages

The model, though it has a overall accuracy of 99.3% but there are few drawback as whole model as well as individual segments. The prediction of handwritten digits is possible only for single digit, in case if the number has many digits, each digit must be given as separate image to predict the number which takes more time and repetition of same steps.And there is low accuracy in few sets of digits. The numbers three (3), six (6) and eight (8) are similar while written hence, the model's prediction accuracy is low. The number seven (7) and one (1) are misunderstood as the numbers look similar when the number are written with hand. In many cases seven (7) the number are wrongly predicted as one (1) and vice versa.

CHAPTER-10
CONCLUSION

Conclusion

The Handwritten digits are not always of the same size, width, orientation and justified to margins as they from writing of person to person. In this project, the Handwritten Digit Recognition using Deep learning methods has been implemented. The most widely used Machine learning algorithms CNN has been trained and tested on the MNIST dataset. The Solution description is by using the MNIST dataset, for handwritten digits to be recognised. The Novelty or uniqueness in this system provides authentication for maintaining privacy of the users and user can store data. Utilizing this deep learning technique, a high amount of accuracy can be obtained. This model is able to achieve a recognition rate of 98.6% accuracy.

The handwritten digit recognition using convolutional neural networks has proved to be of a fairly good efficiency and application for the purpose of introducing and demonstrating neural networks to the general public. The Social Impact or Customer Satisfaction Postal department and courier services can easily find the digits written. Applications of handwriting recognition are numerous: reading postal addresses, bank check amounts, and forms. Business Model (Revenue Model) Used in Banking sector and Postal sector. In banking sectors, numerous handwritten numbers are involved. Our system reduces the human mistakes. The scalability of the Solution, It recognises the handwritten digit in high level of accuracy.

CHAPTER-11
FUTURE SCOPE

Future work:

The model is trained using CNN model and the accuracy achieved is 98.6%. In future the model is trained and aimed to achieved the highest accuracy for overall model as well as the for each digit. The model must also be trained to predict any set of number with any number of digits in single image. Future study might consider using the architecture of the convolution network which gave the best result on the MNIST database and the proposed recognition system is implemented on handwritten digits. Such more system can be designed for handwritten characters recognition, object recognition, image segmentation, handwriting recognition, text language recognition, and future studies also might consider on hardware implementation on online digit recognition system with more performance and efficiency with live results from live testing case scenarios.

CHAPTER-12
APPENDIX

Source Code:

Handwritten Digit Recognition.ipynb

```
# Undestanding Data
import numpy as np
import tensorflow
import keras
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.layers import Conv2D, MaxPooling2D
from keras.optimizers import Adam
from keras.utils import np_utils
(x_train, y_train), (x_test, y_test) = mnist.load_data()
import matplotlib.pyplot as plt
plt.imshow(x_train[0])

# Reshaping the data
x_train = x_train.reshape(60000, 28, 28, 1).astype('float32')
x_test = x_test.reshape(10000, 28, 28, 1).astype('float32')

# One Hot Encoding
number_of_classes = 10
y_train = np_utils.to_categorical(y_train, number_of_classes)
y_test = np_utils.to_categorical(y_test, number_of_classes)

#Model Building
# adding cnn layer
```

```
model = Sequential()
model.add(Conv2D(32, (3,3),input_shape=(28, 28, 1), activation = 'relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64,(3,3), activation = 'relu'))
model.add(Conv2D(64,(3,3), activation = 'relu'))
model.add(Conv2D(32,(3,3), activation = 'relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(10, activation = 'softmax'))
```

#Compiling and Training

```
model.compile(loss=keras.losses.categorical_crossentropy,optimizer=keras.opti
mizers.Adam(), metrics=['accuracy'])
model.fit(x_train, y_train, validation_data = (x_test, y_test), epochs = 5,
batch_size = 120)
```

#Testing the Model

```
prediction = model.predict(x_test[:4])
np.argmax(prediction, axis=1)
metrics = model.evaluate(x_test, y_test, verbose=0)
print('METRICS\n Loss: {:0.3f}\n Accuracy:
:0.3f}'.format(metrics[0],metrics[1]))
```

#Saving the Model

```
model.save('models/mnistCNN.h5')
```

app.py

```
# Importing the packages
from flask import Flask, jsonify, render_template, request
from PIL import Image
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import load_model

# Initializing the app
app = Flask(__name__)

def img_preprocess(image):
    image = image.resize((28,28))
    im2arr = np.array(image)
    im2arr = im2arr.reshape(1,28,28,1)
    if (im2arr[0][0][0][0]>=170):
        im2arr = im2arr - 255
    return im2arr

# Routing
@app.route('/')
def home():
    return render_template('main.html')

@app.route('/recog',methods=['GET', 'POST'])
def get_recog():
    if request.method == 'GET':
        return render_template('index6.html')
    elif request.method == 'POST':
        try:
            img = Image.open(request.files['img'].stream).convert("L")
            img = img_preprocess(img)
            model = load_model("./models/IBM_mnistCNN.h5")
            pred = model.predict(img)
            val = str(np.argmax(pred, axis=1)[0])
            data = {'num': val, 'prob': int(pred[0][int(val)]*100)}
            print(pred)
        except Exception as e:
            print(e)
            val = 'Invalid Input'
        return jsonify(data)
    else:
        return 'Unknown Request'

# Main
if __name__ == '__main__':
    app.jinja_env.auto_reload = True
    app.config['TEMPLATES_AUTO_RELOAD'] = True
    app.run(port=5121, debug=True)
```

main.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="../static/css/style.css" />
    <title>Handwritten Digit Recognition</title>
  </head>
  <style>
  </style>
  <body>
    <nav>
      <ul>
        <li><a href="/">Home</a></li>
        <li><a href="/recog">Recognize</a></li>
      </ul>
    </nav>
    <section class="des">
 Handwritten Text Recognition is a technology that is much needed in this
world as of today. This digit Recognition System is used to recognize the
digits from different sources like emails, bank cheques, papers, images
etc. Before papers implementation of this technology we have relied on
writing texts with our own hands which can result in errors. It's
difficult to store and access physical data with efficiency. The project
presents recognizing the handwritten digit (0 to 9) from the famous MNIST
dataset. Here we will be using artificial neural networks / convolutional
      neural network
    </section>
  </body>
</html>
```

index6.html

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
  <meta charset="UTF-8" />
  <meta http-equiv="X-UA-Compatible" content="IE=edge" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <link rel="stylesheet" href="../static/css/style.css" />
  <title>Handwritten Digit Recognition</title>
</head>
<body>
  <nav>
    <ul>
      <li><a href="/">Home</a></li>
      <li><a href="/recog">Recognize</a></li>
    </ul>
  </nav>
  <section class="content">
    <center>
      <form
        action="/recog"
        method="POST"
        id="img_form"
        enctype="multipart/form-data"
      >
        <input
          type="file"
          name="img"
          onchange="clean_res()"
          id="in_image"
          class="input"
        />
```

```
          <label class="inline" for="in_image"> <span
class="btn">Choose</span></label>
          <section id="preview">THIS IS THE PREVIEW</section>
          <input type="button" class="btn hide" id="remove" value="X"
title="Remove Image"/>
          <input type="submit" id="sub" class="btn" value="Recognize" />
        </form>
        <section id="prediction">
          <div class="res" id="res_box"></div>
        </section>
      </center>
    </section>
  </body>
  <script
    language="JavaScript"
    type="text/javascript"
    src="//ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js"
  ></script>
  <script
    language="javascript"
    type="text/javascript"
    src="../static/js/fn.js"
  ></script>
</html>
```

fn.js

```
$(function () {
  $("#in_image").change(function () {
    $("#preview").html("");
    var regex = /^([a-zA-Z0-9\s_\\.\-:])+(.jpg|.jpeg|.gif|.png|.bmp)$/;
    if (regex.test($(this).val().toLowerCase())) {
      if ($.browser.msie && parseFloat(jQuery.browser.version) <= 9.0) {
        $("#preview").show();
        $("#preview")[0].filters.item(
          "DXImageTransform.Microsoft.AlphaImageLoader"
        ).src = $(this).val();
      } else {
        if (typeof FileReader != "undefined") {
          $("#preview").show();
          $("#preview").append("<img />");
          var reader = new FileReader();
          reader.onload = function (e) {
            $("#preview img").attr("src", e.target.result);
            $("#remove").show();
          };
          reader.readAsDataURL($(this)[0].files[0]);
        } else {
          alert("This browser does not support FileReader.");
        }
      }
    } else {
      alert("Please upload a valid image file.");
    }
  });
});

$("#remove").on("click", () => {
  document.getElementById("img_form").reset();
  $("#preview").hide();
  $("#remove").hide();
  clean_res();
});

$("#img_form").on("submit", function (ev) {
  document.getElementById("sub").value = "Please Wait....";
  ev.preventDefault();
  var formData = new FormData(this);
  $.ajax({
    url: "/recog",
    type: "POST",
    data: formData,
    success: (val) => build_res(val),
    cache: false,
    contentType: false,
    processData: false,
  });
});

function clean_res() {
  document.getElementById("res_box").innerHTML = "";
}

function build_res(val) {
  var resBox = document.getElementById("res_box");

  if (val != "Invalid Input") {
    var span = document.createElement("SPAN");
    span.setAttribute("id", "result");
    span.innerHTML = "Digit: "+val ['num']+"<br>Probability: 
"+val['prob']+"%";
    resBox.appendChild(span);
  } else {
    resBox.innerHTML = val;
  }
  document.getElementById("sub").value = "Recognize";
}
```

Github: *https://github.com/IBM-EPBL/IBM-Project-8538-1658922703*

Demo Video: *https://bit.ly/3EMdF8r*