Novel Method for Handwritten Digit Recognition System

PROJECT REPORT

Submitted by

Ms.A.ARCHANA	**Reg.No.422719106001**
Ms.D.DHARINI	**Reg.No.422719104008**
Ms.K.GOWRI	**Reg.No.422719106003**
Ms.M.PUNITHA	**Reg.No.422719104022**
Ms.M.SUBALAKSHMI	**Reg.No.422719104031**

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

V.R.S. COLLEGE OF ENGINEERING AND TECHNOLOGY
ARASUR – 607107, VILUPPURAM DISTRICT

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2022

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **''Novel Method for Handwritten Digit Recognition System''** is the Bonafede work of "Ms.A.ARCHANA [Reg.No.422719106001], Ms.D.DHARINI [Reg.No.422719104008] and

Ms.K.GOWRI[Reg.No.42271910000]Ms.M.PUNITHA[Reg.No.422719104022] Ms.M.SUBALAKSHMI [Reg.No.422719104031]"who carried out the project work under our supervision.

SIGNATURE	**SIGNATURE**
Mr. K. RAMESH, M.E.,	**Mr.N. GOBINATHAM, M.E.,**
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Department of Computer Science and Engineering, V.R.S. College of Engineering and Technology, Arasur-607107, Villupuram.	Department of Computer Science and Engineering, V.R.S. College of Engineering and Technology, Arasur-607107, Villupuram.

Project viva-voce examination held on ………………

INTERNAL EXAMINER **EXTERNAL EXAMINER**

TABLE OF CONTENT

1.INTRODUCTION

1.1 PROJECT OVERVIEW

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.

1.2 PURPOSE

- MNIST is a dataset which is widely used for handwritten digit recognition. The dataset consists of 60,000 training images and 10,000 test images. The artificial neural networks can all most mimic the human brain and are a key ingredient in image processing field.

- Handwritten character recognition is one of the practically important issues in pattern recognition applications. The applications of digit recognition include in postal mail sorting, bank check processing, form data entry, etc.

- Handwritten digit recognition system (HDR) is meant for receiving and interpreting handwritten input in the form of pictures or paper documents. Traditional systems of handwriting recognition have relied on handcrafted features and a large amount of prior knowledge

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

TITLE **: Handwriting Digit Recognition for Banking System , International Journal of Engineering Research & Technology (IJERT) V.Gopalakrishnan ,R.Arun ,L.Sasikumar, Mrs.K.Abirami , 2021.**

A handwriting digit recognition system's goal is to transform handwritten digits into representations that computers can understand. The major goal of this work is to provide efficient and trustworthy methods for handwritten digit recognition and make banking activities simpler and error-free. The Handwritten Digit Recognition System (HDR) is designed to read and understand handwritten input on paper documents or in the form of images. Traditional handwriting recognition algorithms have depended heavily on existing information and customised features. It is difficult to train an optical character recognition (OCR) system using these requirements. Convolutional neural networks (CNNs) are the most efficient at understanding the structure of handwritten symbols and words, which enables automatic feature extraction.

TITLE : **A Novel Handwritten Digit Classification System Based on Convolutional Neural Network Approach A Novel Handwritten Digit Classification System Based on Convolutional Neural Network Approach. Ali Abdullah Yahya , Jieqing Tan and Min Hu , 2021.**

There have been a tonne of CNN classification algorithms put forth in the literature. However, these algorithms do not take into account the proper filter size selection, data preparation, dataset restrictions, or noise. As a result, few algorithms have been able to significantly increase classification accuracy. Our research makes the following improvements to overcome the inadequacies of existing algorithms: First, the size of the effective receptive field (ERF) is determined after taking the domain knowledge into account. The size of the ERF is taken into account while choosing a typical filter size, which improves the classification accuracy of our CNN. Secondly, excessive data produces inaccurate results, which has a impact on categorization accuracy. Thirdly, to decrease the errors of training and validation, and avoid the limitation of datasets, data augmentation has been proposed. Fourthly, to simulate the real-world natural influences that can affect image quality, we propose to add an additive white Gaussian noise with $\sigma = 0.5$ to the MNIST dataset. As a result, our CNN algorithm achieves state of the art results in handwritten digit recognition, with a recognition accuracy of 99.98%, and 99.40% with 50% noise . In our experiments, batch normalization has been used to improve the training performance and enhance the stability of our model. With the usage of batch normalization, we can speed up the training, reduce training and testing time, in addition to lowering the sensitivity initialization. In order to avoid overfitting and underfitting, an early stopping technique has used to determine the optimal number of training epochs.

TITLE **: NOVEL FRAMEWORK FOR HANDWRITTEN DIGIT RECOGNITION THROUGH NEURAL NETWORKS ,Savita Ahlawat,Amit Choughary,Anand Nayyar,Saurabh Singh ,Byungun Yoon .Sensors (Basel) , 2020.**

Accurately identifying and categorising the hand-written characters presents the biggest barrier for natural language processing algorithms. Since each person has a different handwriting style, size, and other handwriting characteristics, accurately reading handwritten characters is a difficult challenge for humans as well. Even though this machine vision task is rather simple, greater accuracy compared to current methods is still preferred. An innovative neural network-based framework for handwritten character recognition is proposed in this manuscript. In order to achieve image flattening, the suggested neural network-based framework converts the raw data set to a NumPy array and feeds the same into a pixel vector before feeding the network. The activation function is used in the neural network to transfer the outcome value to hidden layer where it is further minimized through the use of minimized mean square and back propagation algorithms before applying a stochastic gradient on the resultant mini–batches.

TITLE : **Handwritten Digit Recognition Using Machine Learning Algorithms, S.M.Shamim, Md Badruln Alam Miah, Angona Sarkar, Masud Rana March 2018.**

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, Naïve Bayes, Bayes Net, Random Forest, J48 and Random Tree has been used for the recognition of digits using WEKA . The main objective of this investigation is to find a representation of isolated handwritten digits that allow their effective recognition This work is carried out as an initial attempt, and the aim of the paper is to facilitate for recognition of handwritten numeral without using any standard classification techniques .

TITLE **: A NOVEL METHOD FOR THE RECOGNITION OF ISOLATED HANDWRITTEN ARABIC CHARACTERS , A Sahlol ,C Suen , 2014.**

There are many difficulties facing a handwritten Arabic recognition system such as unlimited variation in human handwriting, similarities of distinct character shapes, interconnections of neighbouring characters and their position in the word. The typical Optical Character Recognition (OCR) systems are based mainly on three stages, pre-processing, features extraction and recognition. This paper proposes new methods for handwritten Arabic character recognition which is based on novel pre-processing operations including different kinds of noise removal also different kind of features like structural, Statistical and Morphological features from the main body of the character and also from the secondary components. Evaluation of the accuracy of the selected features is made. The system was trained and tested by back propagation neural network with CENPRMI dataset. The proposed algorithm obtained promising results as it is able to recognize 88% of our test set accurately. In Comparable with other related works we find that our result is the highest among other published works**.**

TITLE **: A NOVEL METHOD FOR HAND WRITTEN DIGIT RECOGNITION USING DEEP LEARNING**, **Rohini. M , Dr.D.Surendran ,.INTERNATIONAL JOURNAL OF CURRENT ENGINEERING AND SCIENTIFIC RESEARCH (IJCESR) , 2019.**

Handwritten digit recognition has recently been of very interest among the researchers because of the evolution of various Machine Learning, Deep Learning and Computer Vision algorithms. In this report, We compare the results of some of the most widely used Machine Learning Algorithms like CNN- convolution neural networks and with Deep Learning algorithm like multilayer CNN using Kera with Theano and Tensor-flow . MNIST is a dataset which is widely used for handwritten digit recognition. The dataset consist of 60,000 training images and 10,000 test images . The artificial neural networks can all most mimic the human brain and are a key ingredient in image processing field . For example Convolution Neural networks with back propagation for image processing .The applications where these handwritten digit recognition can be used are Banking sector where it can be used to maintain the security pin numbers, it can be also used for blind peoples by using sound output.

TITLE **: A Machine Learning and Deep Learning Approach for Recognizing Handwritten Digits Ayushi Sharma , Harshit Bhardwaj , Arpit Bhardwaj , Aditi Sakalle , Divya Acharya and Wubshet , 2017**.

Optical character recognition (OCR) can be a subcategory of graphic design that involves extracting text from images or scanned documents. We have chosen to make unique handwritten digits available on the Modified National Institute of Standards and Technology website for this project. The Machine Learning and Deep Learning algorithms are used in this project to measure the accuracy of handwritten displays of letters and numbers. Also, we show the classification accuracy comparison between them. .The results showed that the CNN classier achieved the highest classification accuracy of 98.83%.In this paper, we applied machine learning and deep learning techniques to predict the handwritten digits. Popular algorithms such as KNN, SVM, RFC, DECISION TREE, GNB, GP, and CNN were tested to analyse the differences between them. We are using the backend and tensor-flow as the software library. The CNN classier outperforms the other classier with a classification accuracy of 98.83% for the recognition of handwritten digits.

TITLE : **Survey of Handwritten Character Recognition with MNIST and EMNIST Alejandro Baldominos A , Yago Saez and Pedro Isase , 2019 .**

This paper summarizes the top state-of-the-art contributions reported on the MNIST dataset for handwritten digit recognition. This dataset has been extensively used to validate novel techniques in computer vision, and in recent years, many authors have explored the performance of convolutional neural networks (CNNs) and other deep learning techniques over this dataset .This paper makes a distinction between those works using some kind of data augmentation and works using the original dataset out-of-the-box. Also, works using CNNs are reported separately. Nowadays, a significant amount of works have attained a test error rate smaller than 1% on this dataset, which is becoming non-challenging. By mid-2017, a new dataset was introduced . EMNIST, which involves both digits and letters, with a larger amount of data acquired from a database different than MNIST’s. In this paper, EMNIST is explained and some results are survey. This paper has provided an exhaustive review of the state of the art for both the MNIST and EMNIST databases. The MNIST database of handwritten digits was introduced almost two decades ago and has been extensively used to validate computer vision algorithms, and more recently, also as a benchmark to test different convolutional neural networks architectures and approaches. some works are proposing novel developments or improvements, which are often combined with convolutional neural networks, reporting outstanding results. Although accuracy in MNIST and EMNIST is very close to 100% and will hardly increase, these novel developments might hold the key for breaking more complex computer vision problems.

TITLE **: Multi-Language Handwritten Digits Recognition based on Novel Structural Features . Jaafar M. Alghazo Ghazanfar Latif. Loay Alzubaidi Ammar ,March – April 2019 .**

Automated handwritten script recognition is an important task for several applications. In this article, a multi-language handwritten numeral recognition system is proposed using novel structural features. A total of 65 local structural features are extracted and several classifiers are used for testing numeral recognition. Random Forest was found to achieve the best results with an average recognition of 96.73%. The proposed method is tested on six different popular languages, including Arabic Western, Arabic Eastern, Persian, Urdu, Devanagari, and Bangla. In recent studies, single language digits or multiple languages with digits that resemble each other are targeted. In this study, the digits in the languages chosen do not resemble each other. Yet using the novel feature extraction method a high recognition accuracy rate is achieved. Experiments are performed on well-known available datasets of each language. A dataset for Urdu language is also developed in this study and introduced as PMU-UD. Results indicate that the proposed method gives high recognition accuracy as compared to other methods. Low error rates and low confusion rates were also observed using the novel method proposed in this study. In this paper, we proposed a novel Local Feature Extraction method that is used to design a unified multi-language handwritten numeral recognition system. We targeted many languages even though their digits do not resemble each other. The possibility of redesigning the system in a cloud-based environment will also be part of future work in order to achieve a continuous learning curve and obtain a continuous accuracy improvement.

TITLE : **Unknown-Length Handwritten Numeral String Recognition Using Cascade of PCA-SVMNet Classifiers. SALEH ALY AND AHMED MOHAMED , Deanship of Scientic Research (DSR), Majmaah University , April 29, 2019.**

Automatic recognition of handwritten digit string with unknown length has many potential real applications. The most challenging step in this problem is how to efficiently segment connected and/or overlapped digits exhibited in the input image. Most existing numeral string segmentation approaches combine several segmentation hypotheses to handle various types of connected digits. This paper proposes a new handwritten digit string recognition without applying any explicit segmentation techniques. The proposed method uses a new cascade of hybrid principal component analysis network (PCANet) and support vector machine (SVM) classifier called PCA-SVMNet . PCANet is an emerging unsupervised simple deep neural network typically with only two convolutional layers. The proposed PCA-SVMNet model adds a new fully connected layer trained separately using SVM optimization method. Cascaded stages of PCA-SVMNet classifiers are constructed and trained to recognize various types of isolated and connected digits. Every PCA-SVMNet classifier is trained separately using combinations of real and synthetic touching digits. The first 1D-PCA-SVMNet stage is trained to recognize isolated handwritten digits (0 . . . 9) while forwarding non-isolated digits to the next stages. Each of the following stages is designed to recognize a class of connected digits and forwards the higher class to its successor. Multiple stages can be added accordingly to classify more complex touching digits. The experimental results using NIST SD19 real dataset show that the cascade of PCA-SVMNet classifier efficiently recognizes unknown handwritten digit string without applying any sophisticated segmentation methods. The proposed method achieves state-of-the-art recognition accuracy compared to other segmentation-free techniques . In the experiment using NIST SD19 dataset, where most of the strings contain only isolated digits, the method achieves state-of-the-art results compared to segmentation-free methods and comparable results with segmentation-based techniques.

2.2 REFERENCES

S.NO	Author Name	Paper Title	Journal/ Conference Title	PageNo/ Volume No	Year of Publication	Description
1	Savita Ahlawat , Amit Choudhary, Anand Nayyar, Saurabh Singh and Byungu n Yoon.	Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN)	IEEE Sensors Journal		2020	In this paper, with the aim of improving the performance of handwritten digit recognition, they valuated variants of a convolutional neural network to avoid complex preprocessing, costly feature extraction and a complex ensemble (classifier combination) approach 6 of a traditional recognition system.
2	Vijayalaxmi R	Handwritten Digit	International Journal of	Volume-4 Issue-6	2019	In this paper, the most

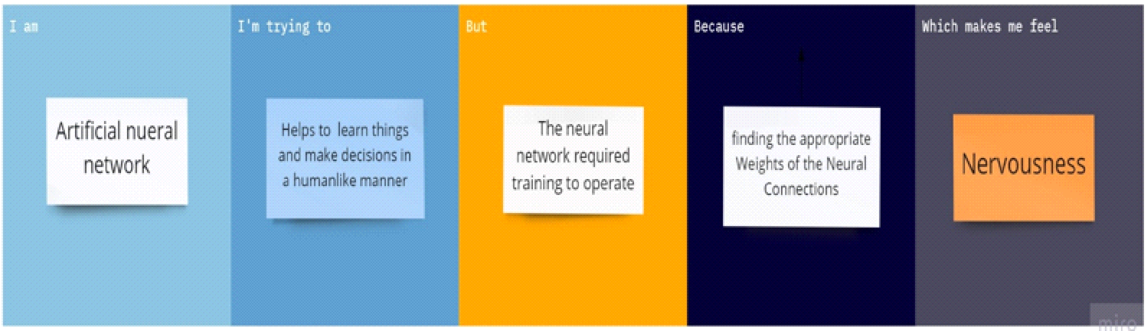
	Rudras wamima th, Bhavani shankar and Channas andra.	Recogniti on using CNN	Innovative Science and Research Technology			widely used Machine learning algorithms, KNN, SVM, RFC and CNN have been trained and tested on the same data in order acquire the comparison between the classifiers
3	Fathma Siddiqu e, Shadma n Sakib and Md. Abu Bakar Siddique.	Recognition of Handwritten Digit using Convolutio na l Neural Network in Python with Tensorflow and Comparison of Performance for Various Hidden Layers	5th Inter national Conference on Advances in Electrical Engineering (ICAEE)		2019	In this paper, they observed the variation of accuracies of CNN to classify handwritten digits for 15 epochs using various numbers of hidden layers and epochs and 7 to make the comparis on between the accuracies. For this performance evaluation of CNN, they

						performed the experiment using Modified National Institute of Standard and Technology (MNIST) dataset.
4	Akanks ha Gupta, Ravindr a Pratap Narwari and Madhav Singh	Review on Deep Learning Handwritten Digit Recogniti on using Convolutio nal Neural Network	International Journal of Recent Technology and Engineering (IJRTE	Volume-9 Issue-5	2021	In this paper, Object Character Recognition (OCR) is used on printed or documented letters to convert them into text. The database has training image database of 60,000 images and 8 testing image database of 10,000 images. The KNN algorithm describes categoric al value by making use of majority of

						votes of K - nearest neighbors, the K value used to differ here.
5	Md. Anwar Hossain and Md. Mohon Ali	Recognition of Handwritten Digit using Convolutional Neural Network (CNN)	Global Journal of Computer Science and Technology: D Neural & Artificial Intelligence	Volume-19 Issue -2	2019	The goal of this work will be to create a model that will be able to identify and determine the handwritten digit from its image with better accuracy using using the concepts of Convolutional Neural Network and MNIST 9 dataset. Later it can be extended for character recognition and realtime person's handwriting. The results can be made more accurate with more convolution

						layers and more number of hidden neurons.

2.3 PROBLEM STATEMENT DEFINITION

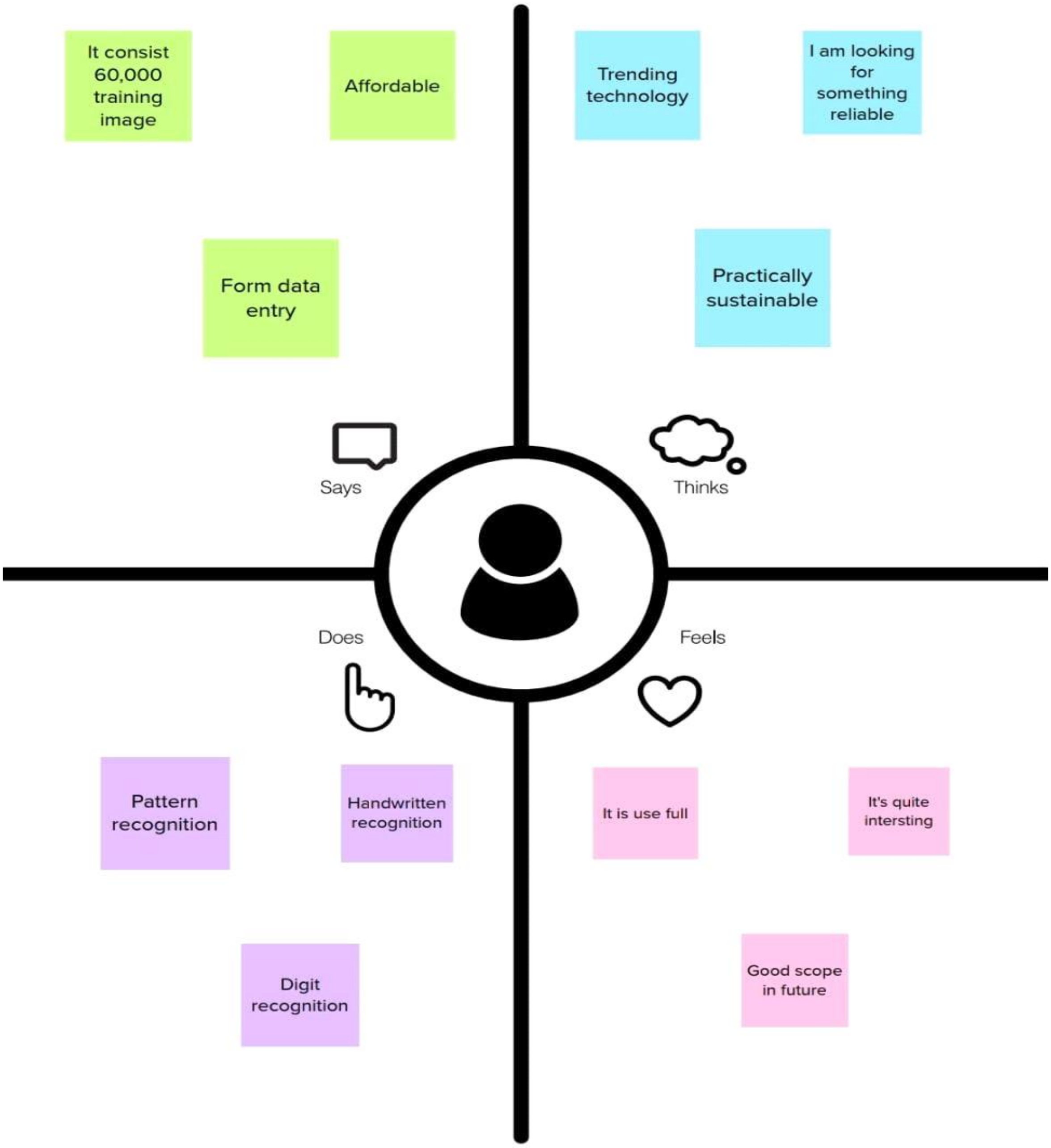


Problem Statement	I am (Customer	I am Trying	But	Because	Which Makes me Feel
PS-1	Artificial Neural Netwo	Helps to learn things and make decisions in humanlike manner	The neura network required training operate	They also n much more Training as compared tc other machi Learning methods	Nervousness
PS-2	Artificial Neural Netwo	Helps to learn things and make decisions in	The neura network required	They also n much more Training as compared tc	Tension

		humanlike manner	training operate	other machi Learning methods	

3. IDEATION & PROPOSED SOLUTION

3.1EMPATHYMAPCANVAS

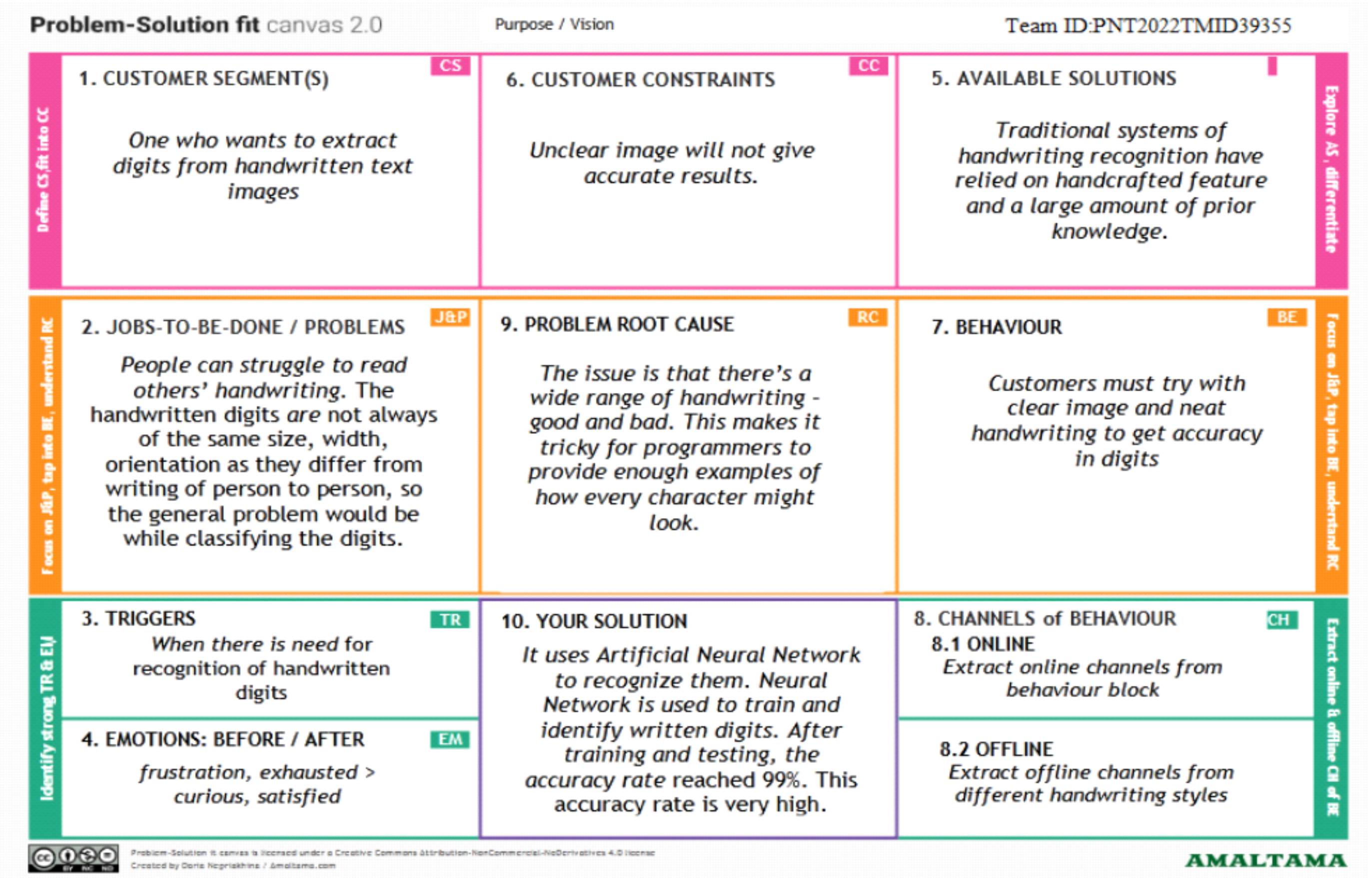


3.2 IDEATION & BRAINSTROMING

3.3 PROPOSED SOLUTION

S.NO	Parameter	Description
1.	Problem Statement (Problem be solved)	**Statement:** The handwritten digit recognition is the capability of computer applications to recognize the human handwritten digits. **Description:** It is a hard task for the machine because handwritten digits are not perfect and can be made with many different shapes and sizes.
2.	Idea / Solution description	1.It is the capability of a computer to fete the mortal handwritten integers from different sources like images, papers, touch defences. 2.It allows user to translate all those signature and notes into electronic words in a text document format and this data only requires far less physical space than the storage of the physical copies.
3.	Novelty / Uniqueness	Accurately recognize the digits rather than recognizing all the characters like OCR.
4.	Social impact / Customer satisfaction	1.Artificial Intelligence developed the app called Handwritten digit Recognizer. 2.It converts the written words into digital approximations and utilizes complex algorithms identify characters before churning out a digital approximation.
5.	Business Model (Revenue Model)	1. This system can be integrated with traffic surveillance cameras to recognize the vehicle’s number plates for effective traffic management. 2. Can be integrated with Postal system to identify and recognize the pin-code details easily.
6.	Scalability of the Solution	1.Ability to recognise digits in more noisy environments. 2.There is no limit in the number of digits it can be recognized

3.4 PROBLEM SOLUTION FIT



4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

Following are the functional requirements of the proposed solution.

FR NO.	Functional Requirement (Epic)	Sub Requirement (Story I Sub-Task)
FR-1	User Input	GUI allows the user to input image b browsing the device storage
FR-2	Model	The MNIST dataset should be trained usi CNN to create a trained model
FR-3	Prediction	The trained model has to be tested by usi the test data provided by MNIST and th accuracy of the model should be above 90%
FR-4	Evaluation	Ensure that the output produced by th model is correct

4.2 NON-FUNCTIONAL REQUIREMENTS

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

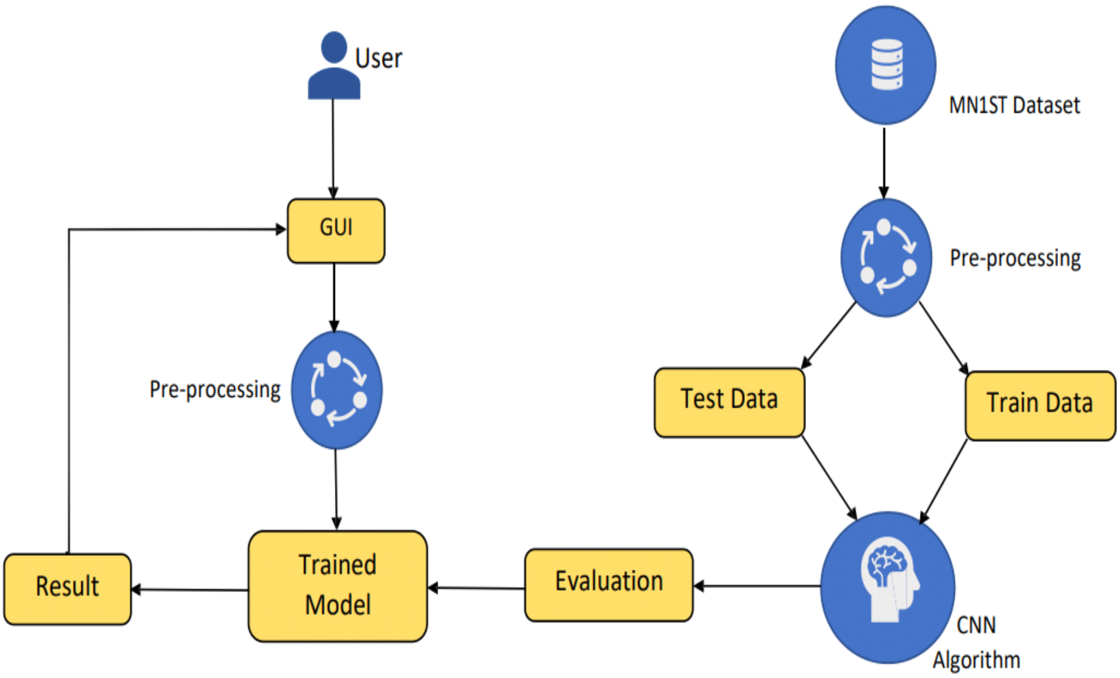
FR NO.	Non Functional Requirement	Description
NFR-1	Usability	Can predict digits with accuracy. The mode can be used in bank check processing, data entry etc..
NFR-2	Security	It ensures security as the uploaded image is not stored in any database
NFR-3	Reliability	Can process confidential information without data leakage as the data is never stored in any database.
NFR-4	Performance	improvement in fast prediction. We use CNN algorithm for accurate prediction
NFR-5	Availability	Available for web and mobile browsers
NFR-6	Scalability	Helps many individuals with low time consumption and high accuracy

5. PROJECT DESIGN

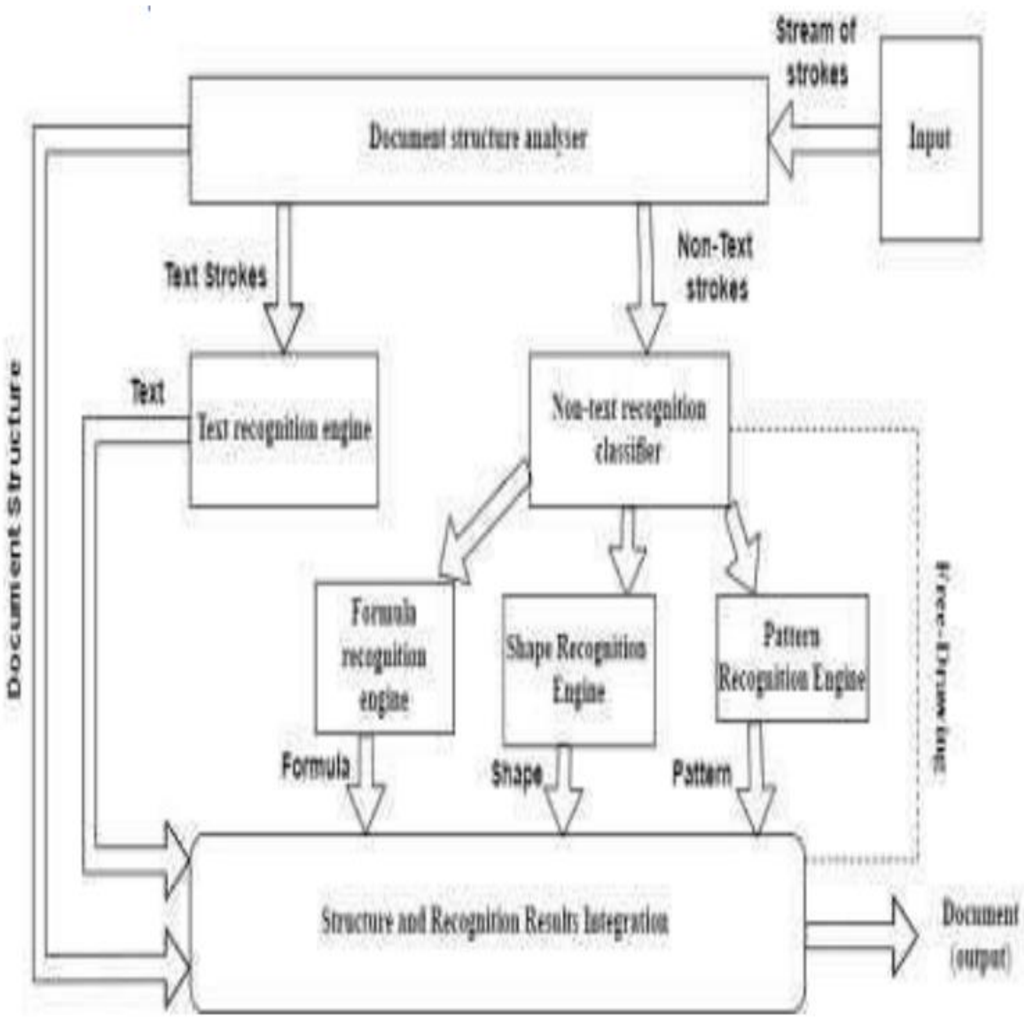
5.1 DATA FLOW DIAGRAMS

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

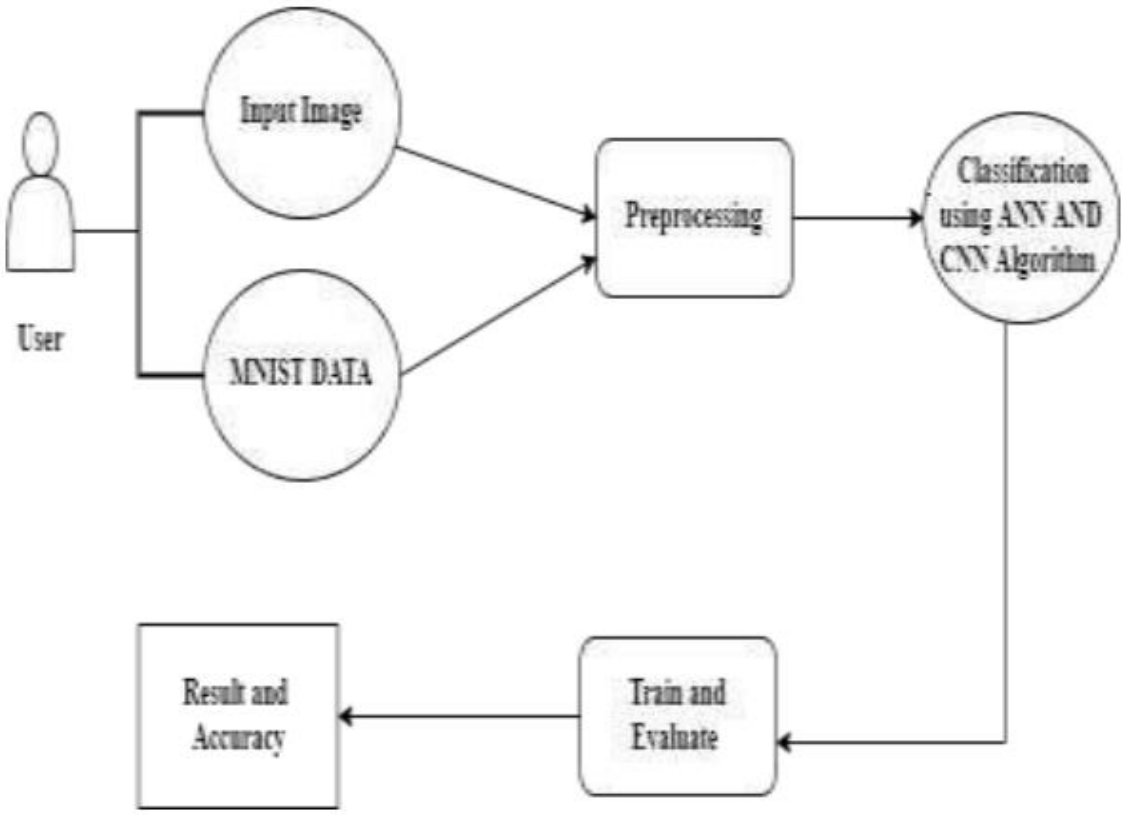
DATA FLOW DIAGRAM



Example: DFD Level 0 (Industry Standard)

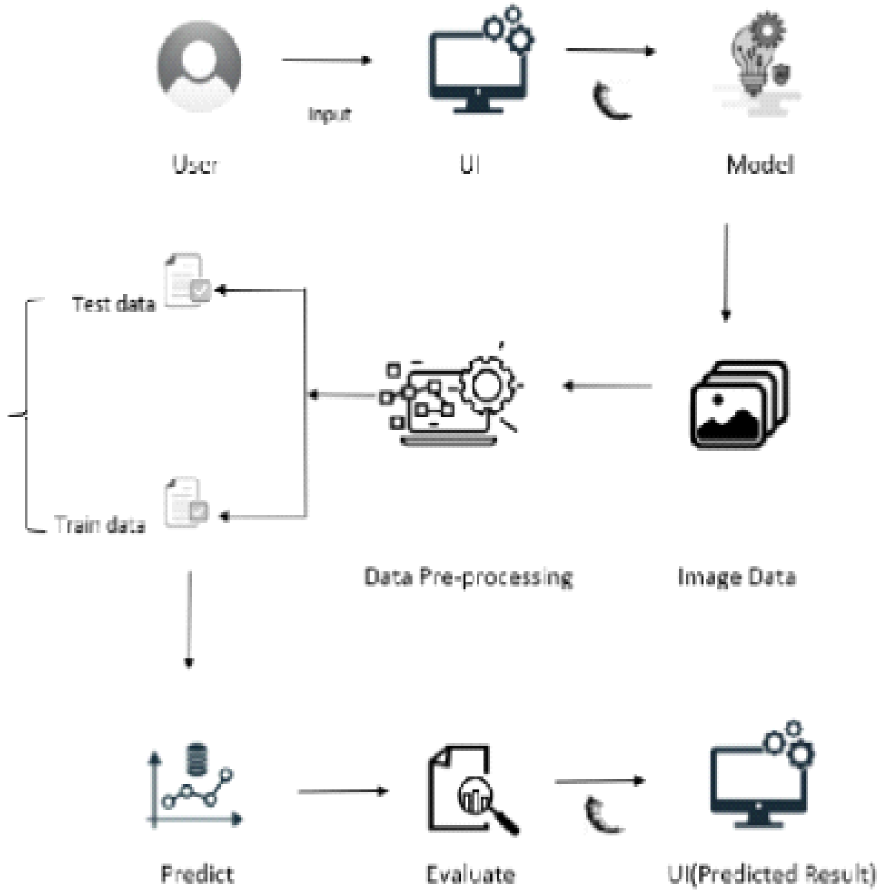


Simplified diagram :



5.2 SOLUTION & TECHNICAL ARCHITECTURE

SOLUTION ARCHITECTURE



TECHNICAL ARCHITECTURE

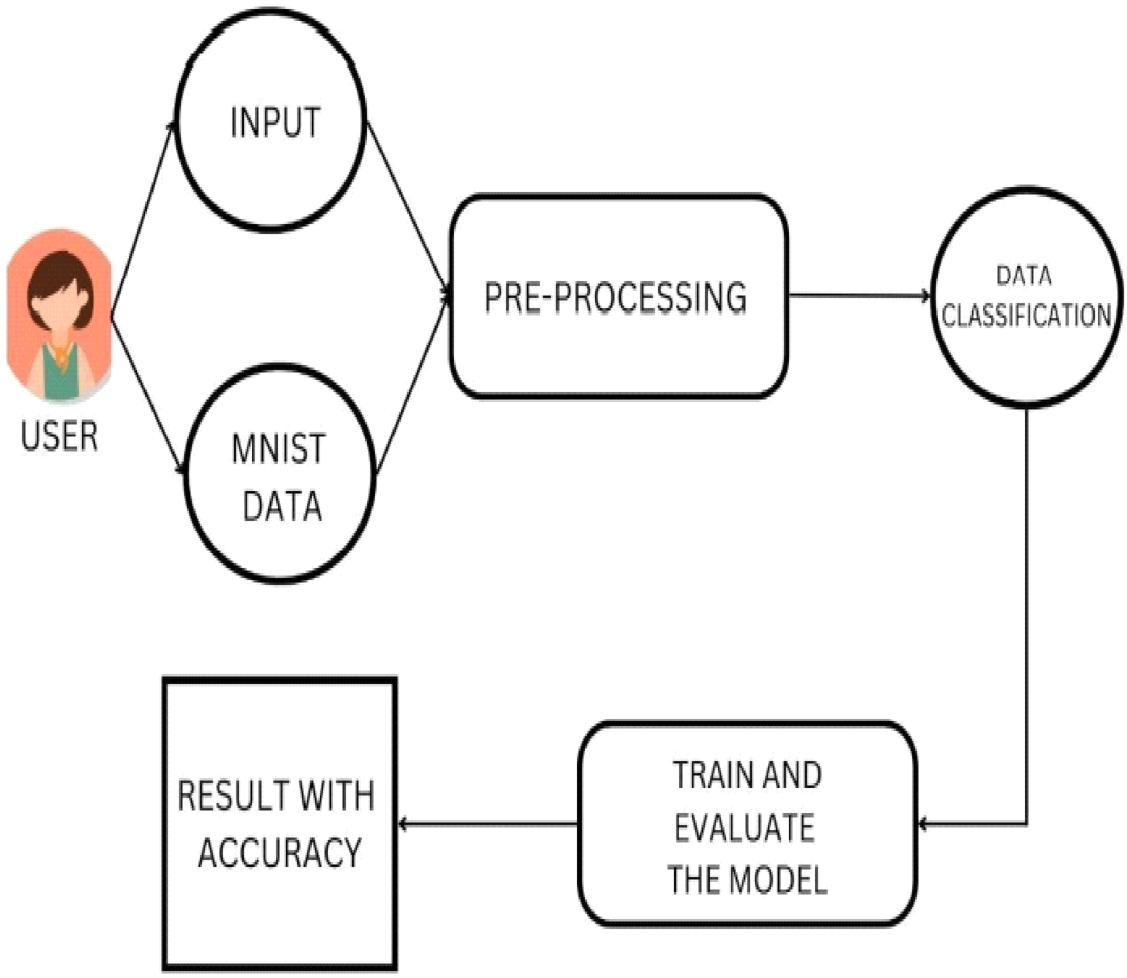


Table -1: Components & Technologies

S.NO	COMPONENT	DESCRIPTION	TECHNOLOGY
1.	User Interface	Allows the user to enter the inp and recognise the input usi GUI	HTML ,CSS, JavaScript
2.	Digit Prediction	Here the digit given as a input is predicted.	Keras ,CNN
3.	Representation	Skeleton, counters, pixels others	Java / Python
4.	Segmentation	Task of clustering parts of an image together that belong to th same object class.	Convolutional neural networks super pixels.
5.	Machine Learning Model	Purpose of Machine Learni Model is to train and test the data and predict the user input.	Classification
6.	Infrastructure	Application deployment on loca system Local serv Configuration: Intel core i5/i3 10th Generation	HTML , CSS
7.	Neutral Network	Automatically infer rules for recognizing handwritten digits Convolutional neur network.	Convolutional Neural Network

Table – 2: Application Characteristics :

S.NO	CHARACTERISTICS	DESCRIPTION	TECHNOLOGY
1.	Pre-processing	Data pre-processing is a process of preparing th raw data and making suitable for a machine learnin model.	Real time online Handwritten character recognitio system, based on an ensemble of neural networks.
2.	Open-Source Frameworks	Enables developers to develop complex co and web application quickly	Open source-Jupiter anaconda navigator, flask framework.
3.	Dataset	It Contains 60,0(training images .	MNIST
4.	Security Implementations	After predicting the dat we don't store any data so w can't manipulate it future.	Encryption
5.	Performance	Neural networks achie an accuracy of ~(98–9 percent in correct classifying the hand written digits	Convolutional Neural Networks

5.3 USER STORIES

User Stories	Functional requirement	User story number	User Story I as	Acceptance criteria	Priority	Release
Customer (Web user)	Home	USN-1	In the Home Page, I can view the guidelin of how to use the websi	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 c Upload an image hand written digit f prediction	I can upload my input by browsing the device storage	Medium	Sprint-3
		USN-4	As a user, I can get ar accuracy rate with the prediction	I can get different forms of output	High	Sprint-4
	Recognise	USN-5	As a user, I can see tl the GUI processing t input using trained model	I can perform handwritten digit prediction	High	Sprint-1
	Prediction	USN-6	As a user, I can get a accuracy rate 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 acces application in mob phone	I can access the dashboard with mobile	Medium	Sprint-1
	Recognise	USN-8	I can upload input anc	I can upload input	High	Sprint-2

			retrieve output with tl accuracy by using the mobile	image and get output with a mobile device		

PROJECT PLANNING & SCHEDULING

Sprint	Functional Requirement (Epic)	User Story Number	User Story/Task	Story points	priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password	2	High	Dharini D
Sprint-1	Login	USN-2	As a user, I can log into the application by entering email &password	1	High	Punitha M
Sprint-2	Upload Image of digital document	USN-3	As a user, I can able to input the images of digital documents to	2	Medium	Gowri K

			the application			
Sprint-2	Prediction	USN-4	As a user, I can predict the word	1	Medium	Archana A
Sprint-3	Upload image of Handwritten document	USN-5	As a user, I can able to input the images of the handwritten documents or images to the application	2	High	Subalakshmi M
Sprint-3	Recognize text	USN-6	As a user, I can able to choose the font of the text to be displayed	1	Medium	Dharini D
Sprint-4	Recognize digit	USN-7	As a user I can able to get recognized digit as output from the images of digital documents or images	**1**	Medium	Punitha M
Sprint-4	Recognize digit	USN-8	As a user I can able to get the recognized digit as output from the images of handwritten documents or images	2	High	Gowri k

6.1 Sprint Planning & Estimation

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	2	6 Days	24 oct 2022	29 oct 2022	2	29 Oct 2022
Sprint	2	6Days	31 Oct 2022	05 Nov 2022	2	05 Nov 2022
Sprint	2	6Days	07 Nov 2022	12 Nov 2022	2	12 Nov 2022
Sprint	2	6Days	14 Nov 2022	19 Nov 2022	2	19 Nov 2022

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

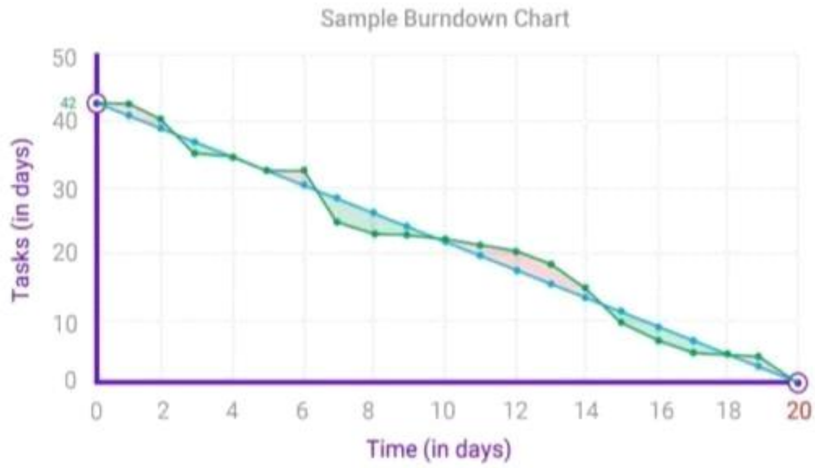
AV = sprint duration/velocity

= 20/10

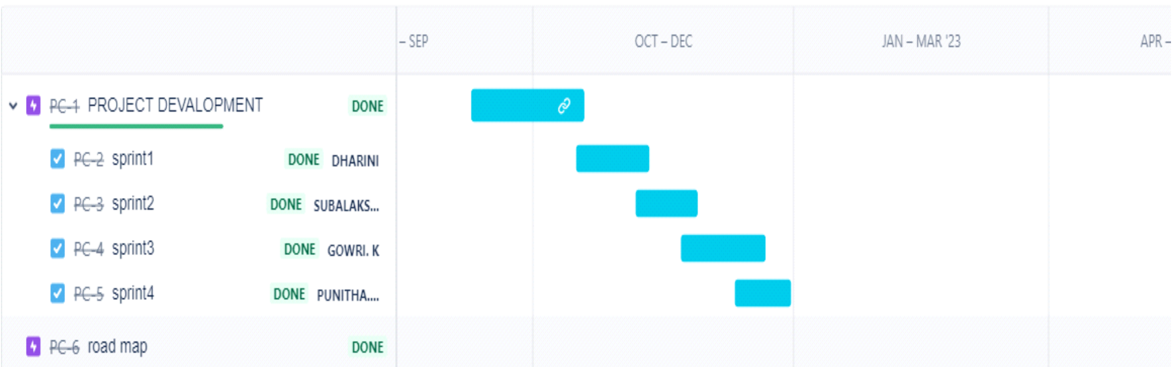
= 2

Burndown Chart:

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



6.3 Reports from JIRA



7.CODING & SOLUTIONING

7.1 Feature 1

Index.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">
    <title>Handwritten Recognition System</title>
    <link rel="stylesheet" href="style.css">
</head>

<body>
    <header class="header">
        <nav class="navbar">
            <ul>
                <li>
                    <a href="#">Home</a>
                </li>
                <li>
                    <a href="recognize.html">Recognize</a>
                </li>
            </ul>
        </nav>
    </header>

    <div class="bg-pic"></div>

    <main class="main">
        <h1 class="main-heading">Handwritten Recognition System</h1>

        <p class="content">
            <em>
                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 cheque,
                papers, images, etc. Before proper 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
digits (0 to 9) from the famous MNIST
                dataset. Here we will be using artificial neural networks
convalution neural network.
 </em>
 </p>
 </main>
 </body>
 </html>
```

Recognize text.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="./recog.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.2.0/css/all.min.css">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@500;700&display=swa
p');
*{
    margin: 0;
    padding: 0;
}
.nav-container{
    display: flex;
```

```
    justify-content: end;
    height: 100px;
}
.nav-list{
    padding-right: 50px;
    display: flex;
    width: 20%;
    justify-content: end;
    align-items: center;
}
.nav-item{
    width: 70%;
    display: flex;
    justify-content: space-around;
}
.nav-item .nav-links {
    list-style: none;
}
.nav-item .nav-links input{
    text-decoration: none;
    font-family: 'Poppins', sans-serif;
    color: black;
    border: none;
    background-color: white;
    font-size: 16px;
    cursor: pointer;
}
.heading{
    display: flex;
    justify-content: center;
}
.heading .sub-div {
    text-align: center;
    width: 40%;
    font-family: 'Poppins', sans-serif;
    font-weight: 700;
    margin-bottom: 50px;
}
.content-container{
    display: flex;
    justify-content: center;
    height: 400px;
}
.content-container .boxs{
    position: relative;
    display:flex;
    justify-content: space-around;
```

```css
    width: 100%;
}
.content-container .boxs .box1{
    align-items: center;
    flex-direction: column;
    display:flex;
   width: 40%;
   background-color: rgba(117, 117, 117, 0.219);
   border-radius: 20px;
   position: relative;
}
.content-container .boxs .box1 .box1-title{
    font-family: 'Poppins', sans-serif;
    margin: 20px;
}
.content-container .boxs .box2 .box2-title{
    font-family: 'Poppins', sans-serif;
    margin: 17px;
}
.content-container .boxs .box2{
    align-items: center;
    flex-direction: column;
    display:flex;
   width: 40%;
   background-color: rgba(117, 117, 117, 0.219);
   border-radius: 20px;
}
.content-container .boxs .box1 .btn-1{
    display: flex;
    justify-content: center;
}
.input-box1{
    width: 400px;
    height:100px;
    position: relative;
    padding-top: 50px;
    text-align: center;
    margin: 30px 0;
    font-family: 'Poppins', sans-serif;
}
.input-box2{
    padding-top: 30px;
    text-align: center;
    width:400px;
    height:100px;
    position: relative;
    margin: 30px 0;
```

```
    font-family: 'Poppins', sans-serif;
}

.btn-1 input{
    border: none;
    width: 130px;
    height: 40px;
    border-radius: 30px;
    background-color: rgb(16, 197, 61);
    color: white;
    font-family: 'Poppins', sans-serif;
}
.btn-2 input{
    border: none;
    margin-top: 10px;
    width: 130px;
    height: 40px;
    border-radius: 30px;
    background-color: rgb(197, 58, 16);
    color: white;
    font-family: 'Poppins', sans-serif;
}
    </style>
</head>
<body>
        <div class="nav-container">
            <div class="nav-list">
                <ul class="nav-item">
                    <li class="nav-links home">
                        <i class="fa-solid fa-house"></i>
                        <input type="submit" value="Home">
                         </li>
                         <form action="/recognize_page" method ="post">
                    <li class="nav-links"><input type="submit"
value="Recognize"></li></form>
                </ul>
            </div>
        </div>
    <div class="heading">
        <div class="sub-div">
            <h1>Handwritten Digit Recognition System</h1>
        </div>
    </div>
    <div class="content-container">
        <div class="boxs">
            <div class="box1">
```

```
                <div class="box1-title"><h3>Recognizing Digits from Drawing
Images</h3></div>
                <div class="input-box1">Draw the digit on the given canva and
click predict to recognize the drawn digit </div>
                <div class="btn-1"><form action="/recognize"
method="post"><input type="submit" value="Recognize"></form></div>
            </div>
            <div class="box2">
                <div class="box2-title"><h3>Recognizing Handwritten Digits
from Uploaded Document</h3></div>
                <div class="input-box2">Upload the image containing the
handwritten digit and click predict to recognize the digit in the image </div>
                <div class="btn-2"><input type="submit"
value="Recognize"></div>
            </div>
        </form>
        </div>
    </div>
</body>
</html>
```

Feature 2

```
import numpy as np
 import pandas as pd
 import matplotlib.pyplot as plt from keras.utils
 import np_utils from tensorflow.keras.datasets
 import mnist from tensorflow.keras.models
 import Sequential
from tensorflow.keras.layers import Conv2D, Dense, Flatten
from tensorflow.keras.optimizers import Adam
 from tensorflow.keras.models import load_model
 from PIL import Image, ImageOps
 import numpy
 (X_train, y_train), (X_test, y_test) = mnist.load_data()
```

Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mni
11490434/11490434 [==============================] - 0s 0us/step

```
print(X_train.shape)
print(X_test.shape)
(60000, 28, 28)
(10000, 28, 28)
X_train[0]
253, 253, 253, 253, 253, 225, 172, 253, 242, 195, 64, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 49, 238, 253, 253, 253, 253, 253, 253, 253, 253, 251, 93, 82, 82, 56, 39, 0, 0, 0, 0,
0],
[ 0, 0, 0, 0, 0, 0, 0, 18, 219, 253, 253, 253, 253, 253, 198, 182, 247, 241, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, 80, 156, 107, 253, 253, 205, 11, 0, 43, 154, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 1, 154, 253, 90, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 139, 253, 190, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 11, 190, 253, 70, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 35, 241, 225, 160, 108, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 81, 240, 253, 253, 119, 25, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 45, 186, 253, 253, 150, 27, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 16, 93, 252, 253, 187, 0, 0, 0, 0, 0, 0, 0, 0] ,
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 249, 253, 249, 64, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 46, 130, 183, 253, 253, 207, 2, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 39, 148, 229, 253, 253, 253, 250, 182, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 24, 114, 221, 253, 253, 253, 253, 201, 78, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, 23, 66, 213, 253, 253, 253, 253, 198, 81, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 18, 171, 219, 253, 253, 253, 253, 195, 80, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 55, 172, 226, 253, 253, 253, 253, 244, 133, 11, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 136, 253, 253, 253, 212, 135, 132, 16, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
y_train[0]
5
plt.imshow(X_train[0])
X_train = X_train.reshape(60000, 28, 28, 1).astype('float32')
X_test = X_test.reshape(10000, 28, 28, 1).astype('float32')
number_of_classes = 10
Y_train = np_utils.to_categorical(y_train, number_of_classes)
Y_test = np_utils.to_categorical(y_test, number_of_classes)
Y_train[0]
array([0., 0., 0., 0., 0., 1., 0., 0., 0., 0.], dtype=float32)
model = Sequential()
model.add(Conv2D(64, (3, 3), input_shape=(28, 28, 1), activation="relu"))
model.add(Conv2D(32, (3, 3), activation="relu"))
model.add(Flatten())
```

model.add(Dense(number_of_classes, activation="softmax"))
model.compile(loss='categorical_crossentropy', optimizer="Adam", metrics=["accuracy"])
model.fit(X_train, Y_train, batch_size=32, epochs=5, validation_data=(X_test,Y_test))
Epoch 1/5
1875/1875 [==============================] - 192s 102ms/step - loss: 0.2245 – accuracy
Epoch 2/5
1875/1875 [==============================] - 197s 105ms/step - loss: 0.0685 - accurac
Epoch 3/5
1875/1875 [==============================] - 190s 101ms/step - loss: 0.0468 - accurac
Epoch 4/5
1875/1875 [==============================] - 190s 102ms/step - loss: 0.0351 – accuracy
Epoch 5/5
1875/1875 [==============================] - 191s 102ms/step - loss: 0.0270 - accurac
metrics = model.evaluate(X_test, Y_test, verbose=0)
print("Metrics (Test Loss & Test Accuracy): ")
print(metrics)
Metrics (Test Loss & Test Accuracy):
[0.10052110999822617, 0.9764000177383423]
prediction = model.predict(X_test[:4])
print(prediction)
1/1 [==============================] - 0s
92ms/step
[[1.5678695e-09 1.6640128e-14 2.0494097e-12 1.5698962e-08 5.4015579e-15 3.6338055e-13 2.2240399e-20 1.0000000e+00 2.9577885e-08 1.9005494e-08] [5.8188578e-09 1.2512093e-10 9.9999821e-01 7.4831279e-09 1.0770124e-10 2.9252167e-18 1.6483800e-06 1.5410843e-14 1.2811967e-07 3.3103555e-12] [1.2689595e-09 9.9028254e-01 3.9091717e-08 1.3732340e-10 9.6216686e-03 2.9094124e-07 1.9340013e-10 4.5208512e-07 9.5003670e-05 2.4108826e-10] [1.0000000e+00 7.3556976e-16 3.5439882e-12 4.7910155e-14 3.2022885e-12 1.5000925e-12 1.5939531e-11 4.1566353e-14 7.7353792e-12 1.2456662e-09]] print(numpy.argmax(prediction, axis=1)) print(Y_test[:4]) 11/9/22, 10:12 PM Sprint 4 -PNT2022TMID39355.ipynb - Colaboratory https://colab.research.google.com/drive/1mz1KPu_TE342fmzwSMhWKi6ukA3UM6sf#printMode=true 4/5 [7 2 1 0] [[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.] [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.] [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.] [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]] model.save("model.h5") model=load_model("model.h5") from keras.datasets import mnist from matplotlib import pyplot (X_train,y_train),(X_test,y_test)=mnist.load_data() print('X_train:' +str(X_train.shape)) print('y_train:' +str(y_train.shape)) print('X_test:' +str(X_test.shape)) print('y_test:' +str(y_test.shape)) from matplotlib import pyplot for i in range(9): pyplot.subplot(330+1+i) pyplot.imshow(X_train[i],cmap=pyplot.get_cmap('gray')) pyplot.show(

8.TESTING

8.1 Test Cases

Test Case ID	Feature Type	Component	Test Scenario	Expected Result	Actual Result	Status
Homepage_TC_OO1	Functional	Home Page	Verify user is able to see the Homepage when clicked on the link	Home Page should be displayed	Working as expected	pass
Homepage_TC_OO2	UI	Home Page	Verify the UI elements in Homepage	Application should show below UI elements: a.choose file button b.predict button c.clear button	Working as expected	pass
Homepage_TC_OO3	Functional	Home Page	Verify user is able to choose file from the local system and click on predict	Choose file popup screen must be displayed and user should be able to click on predict button	Working as expected	pass
Homepage_TC_OO4	Functional	Home Page	Verify user able to select invalid file format	Application won't allow to attach formats other than ".png, .jiff, .pjp, .jpeg, .jpg, .pjpeg"	Working as expected	pass
Homepage_TC_OO5	Functional	Predict Page	Verify user is able to navigate to the predict to	User must be navigated to the predict	Working as expected	pass

			and view the predicted result	page and must view the predicted result		

8.2User Acceptance Testing

Defect Analysis

Resolution	**Severity 1**	**Severity 2**	**Severity 3**	**Severity 4**	**Subtotal**
By Design	0	0	0	0	0
Duplicate	0	0	0	0	0
External	0	0	0	0	0
Fixed	0	0	0	0	0
Not Produced	0	0	0	0	0
Skipped	0	0	0	0	0
Wont't Fix	0	0	0	0	0
Total	0	0	0	0	0

Test Case Analysis

Section	**Total Cases**	**Total Cases**	**Fail**	**Pass**
Client Application	5	0	0	5
Security	5	0	0	5
Final Report Output	5	0	0	5
Performance	5	0	0	5

9. RESULTS

9.1 PERFORMANCE METRICS

Handwritten digit recognition is in the forefront of optical character recognition for a long time. Starting from a postal mail sorting to bank cheque processing it has many applications. In this paper, a research is done into several machine learning algorithms for this purpose. As unsupervised learning algorithms, we explore K-means clustering techniques along with principal components analysis to reduce the dimensionality of the data. For supervised learning, we begin with a linear classifier and then explore neural network, support vector machines and nearest neighbour based algorithms. For benchmarking the learning algorithms, we used a well-researched dataset named MNIST and compare the performances using prediction accuracy. In this work, we discuss theoretical details of the methods under consideration and whether any tuning parameters need to be chosen by the user for improved performance. Since this is a huge database, we also discuss some computation issues regarding the implementation of the algorithms. All computations are performed using R.

10.ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- Handwriting recognition helps to transform the writings in the papers to a text document format which can also be said as readable electronic format. By this way, historical facts can be stored, reviewed and shared easily too many people.
- Textual studies.

- 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.

DISADVANTAGES

- Plus, sometimes, characters look very similar, making it hard for a computer to recognise accurately.
- Joined-up handwriting is another challenge for computers. When your letters all connect, it makes it hard for computers to recognise individual characters.
- It is not done in real time as a person writes and therefore not appropriate for immediate text input.

11. CONCLUSION

This project demonstrated a web application that uses machine learning to recognize handwritten numbers. Flask, HTML, CSS, JavaScript, and a few other technologies were used to create this project. The model predicts the handwritten digit using a CNN network. During testing, the model achieved a 99.61% recognition rate. The proposed project is scalable and can easily handle a huge number of users. Since it is a web application, it is compatible with any device that can run a browser. This project is extremely useful in realworld scenarios such as recognizing number plates of vehicles, processing bank cheque amounts, numeric entries in forms filled up by hand (tax forms) and so on. There is so much room for improvement, which can be implemented in subsequent versions

12.FUTURE SCOPE

This project is far from complete and there is a lot of room for improvement. Some of the improvements that can be made to this project are as follows:ssss

- Add support to detect from digits multiple images and save the results
- Add support to detect multiple digits
- Improve model to detect digits from complex images
- Add support to different languages to help users from all over the world This project has endless potential and can always be enhanced to become better. Implementing this concept in the real world will benefit several industries and reduce the workload on many workers, enhancing overall work efficiency.

13.APPENDIX

13.1 Source Code

HTML AND CSS:

<u>index.html:</u>

```
<html>

<head>

  <title>Digit Recognition WebApp</title>

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

  <!-- GoogleFont -->
```

```
  <link
href="https://fonts.googleapis.com/css2?family=Prompt:wght@600&display=s
wap" rel="stylesheet">
  <link
href="https://fonts.googleapis.com/css2?family=Varela+Round&display=swap"
rel="stylesheet">
  <link
href="https://fonts.googleapis.com/css2?family=Source+Code+Pro:wght@500
&display=swap" rel="stylesheet">
  <link
href="https://fonts.googleapis.com/css?family=Calistoga|Josefin+Sans:400,700|
Pacifico&display=swap" rel="stylesheet">
  <!-- bootstrap -->
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2
MZw1T" crossorigin="anonymous">
  <link rel="stylesheet" type= "text/css" href= "{{
url_for('static',filename='css/style.css') }}">
  <!-- fontawesome -->
  <script src="https://kit.fontawesome.com/b3aed9cb07.js"
crossorigin="anonymous"></script>
```

```
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6j
izo" crossorigin="anonymous"></script>

  <script
>src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js
" integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIH
NDz0W1" crossorigin="anonymous"></script>

  <script
>src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07j
RM" crossorigin="anonymous"></script>

  <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest"></script>

  <style>

  #clear_button{

  margin-left: 15px;

  font-weight: bold;

  color: blue;

}

#confidence{

  font-family: 'Josefin Sans', sans-serif;

  margin-top: 7.5%;
```

```
}
#content{
  margin: 0 auto;
  padding: 2% 15%;
  padding-bottom: 0;
}
welcome{
   text-align: center;
   position: relative;
   color: black;
   background-color: rgba(0, 0, 0, 0.068);
   padding-top: 1%;
   padding-bottom: 1%;
   font-weight: bold;
   font-family: 'Prompt', sans-serif;
}
#team_id{
   text-align: right;
   font-size: 25px;
   padding-right: 3%;
}
```

Predict.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Prediction</title>
</head>
<style>
    body{
     background-image: url('index6.jpg');
     background-repeat: no-repeat;
     background-size: cover;
    }
    #rectangle{
     width:400px;
     height:150px;
     background-color: #5796a5;
     border-radius: 25px;
     position:absolute;
     top:25%;
     left:50%;
     transform:translate(-50%,-50%);
    }
```

```
    #ans{
  text-align: center;
  font-size: 40px;
  margin: 0 auto;
  padding: 3% 5%;
  padding-top: 15%;
  color: white;
    }
</style>
<body>
    <div id="rectangle">
        <h1 id="ans">Predicted Number : {{num}}</h1>
    </div>
</body>
</html>
```

Style.css

```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@500;700&display
=swap');
*{
    margin: 0;
    padding: 0;
```

```
}
body{
    background-image: url(./bg.jpg);
    background-size: cover;
}
.nav-container{
    display: flex;
    justify-content: end;
    height: 100px;
}
.nav-list{
    padding-right: 50px;
    display: flex;
    width: 20%;
    justify-content: end;
    align-items: center;
}
.nav-item{
    width: 70%;
    display: flex;
    justify-content: space-around;
}
```

```css
.nav-item .nav-links {
    list-style: none;
}
.nav-item .nav-links a{
    text-decoration: none;
    font-family: 'Poppins', sans-serif;
    color: black;
}
.heading{
    display: flex;
    justify-content: center;
}
.heading .sub-div {
    text-align: center;
    width: 40%;
    font-family: 'Poppins', sans-serif;
    font-weight: 700;
    margin-bottom: 50px;
}
.content{
    display: flex;
    justify-content: center;
```

```
}
.des{
    width: 70%;
    text-align: center;
    font-family: 'Poppins', sans-serif;
    justify-content: center;
    display: flex;
    background-color: rgba(0, 0, 0, 0.126);
    border-radius: 30px;
}
.sub-des{
   font-size: 20px;
   padding: 70px 0;
   width: 80%;
}
```

FLASK:

app.py:

```
from unittest import result
from flask import Flask,render_template,request,redirect,url_for
#import numpy as np
from PIL import Image
```

```
from tensorflow.keras.models
#import load_model
# from tensorflow.k
app = Flask(__name__)
@app.route('/',methods=["GET"])
def index():
    return render_template('index.html')
@app.route('/predict',methods=["POST","GET"])
def predict():
    if request.method == "POST":
        print(request.files['image'])
        img = Image.open(request.files['image'].stream).convert("L")
        img = img.resize((28,28))
        imgToArr = np.array(img)
        imgToArr = imgToArr.reshape(1,28,28,1)
        pred = model.predict([imgToArr])
        print(pred)
        y_pred = np.argmax(pred,axis=1)
        print("The image is "+str(y_pred))
        #return redirect('/output',message = y_pred)
        return redirect( url_for('.output',number = str(y_pred[0])))
    if request.method=="GET":
```

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

@app.route('/output',methods=["GET"])

def output():

    val = request.args.get('number')

    if  val :

        print(val)

            return render_template('result.html',result = val)

    return redirect('/')

if __name__=="__main__":

    model = load_model('Sprint 3\models\mnistCNN.h5')

# Show the model architecture

    app.run(debug=True)
```

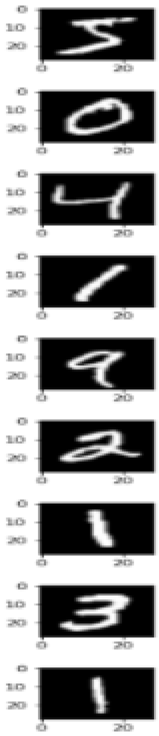
<u>MODEL CREATION</u>

Test with Saved the Model

In [34]:

```
from keras.datasets import mnist
from matplotlib import pyplot
(X_train,y_train),(X_test,y_test)=mnist.load_data()
print('X_train:' +str(X_train.shape))
print('y_train:' +str(y_train.shape))
print('X_test:' +str(X_test.shape))
print('y_test:' +str(y_test.shape))
from matplotlib import pyplot
for i in range(9):
  pyplot.subplot(330+1+i)
  pyplot.imshow(X_train[i],cmap=pyplot.get_cmap('gray'))
  pyplot.show()
```

```
X_train:(60000, 28, 28)
y_train:(60000,)
X_test:(10000, 28, 28)
y_test:(10000,)
```



Train the Model

```
In [17]: model.fit(X_train, Y_train, batch_size=32, epochs=5, validation_data=(X_test,Y_test))
```

```
Epoch 1/5
1875/1875 [==============================] - 194s 103ms/step - loss: 0.2567 - accuracy: 0.9506 - val_loss: 0.0980 - val_accuracy: 0.9693
Epoch 2/5
1875/1875 [==============================] - 196s 105ms/step - loss: 0.0695 - accuracy: 0.9791 - val_loss: 0.0983 - val_accuracy: 0.9735
Epoch 3/5
1875/1875 [==============================] - 196s 105ms/step - loss: 0.0494 - accuracy: 0.9842 - val_loss: 0.0906 - val_accuracy: 0.9755
Epoch 4/5
1875/1875 [==============================] - 192s 102ms/step - loss: 0.0375 - accuracy: 0.9882 - val_loss: 0.0913 - val_accuracy: 0.9787
Epoch 5/5
1875/1875 [==============================] - 196s 104ms/step - loss: 0.0306 - accuracy: 0.9903 - val_loss: 0.1032 - val_accuracy: 0.9743
```

Test the Model

```
In [29]: prediction = model.predict(X_test[:4])
         print(prediction)

1/1 [==============================] - 0s 64ms/step
[[4.77358049e-11 1.26020884e-14 2.23637656e-07 2.59297366e-07
  1.53105145e-18 1.41474479e-13 2.73819453e-19 9.99999523e-01
  5.75746352e-12 1.40723442e-08]
 [3.92702641e-05 3.63764530e-09 9.99928832e-01 1.10518204e-06
  3.28396650e-11 1.87219923e-13 3.02575540e-06 4.75269130e-12
  2.79003762e-05 1.17118581e-09]
 [3.37602168e-11 9.99982953e-01 7.10459869e-09 3.63090309e-13
  1.67968246e-05 6.36366426e-09 4.59948364e-11 2.65287614e-09
  2.72516672e-07 1.53049936e-12]
 [9.99999762e-01 1.02759820e-17 6.89465485e-10 4.13503087e-14
  3.53135576e-12 2.56500203e-11 6.89072754e-09 4.50628203e-14
  8.74276596e-10 1.82247064e-07]]
```

```
In [22]: print(numpy.argmax(prediction, axis=1))
         print(Y_test[:4])

[7 2 1 0]
[[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
```

13.2 .GitHub & Project Demo Link

GitHub Link: https://github.com/IBM-EPBL/IBM-Project-47359-1660798512

Demo Video:https : //youtu.be/VVrMrif3MUY