A Novel Method for Handwritten Digit Recognition System

NALAIYA THIRAN PROJECT BASED LEARNING

on

PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTREPRENEURSHIP

A PROJECT REPORT

Suvetha Sri RP - 310619104144
Swetha B - 310619104147
Uma Sruthi P - 310619104155
Varshini S - 310619104157
Swetha M -310619104148

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

EASWARI ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

CHENNAI – 600 089

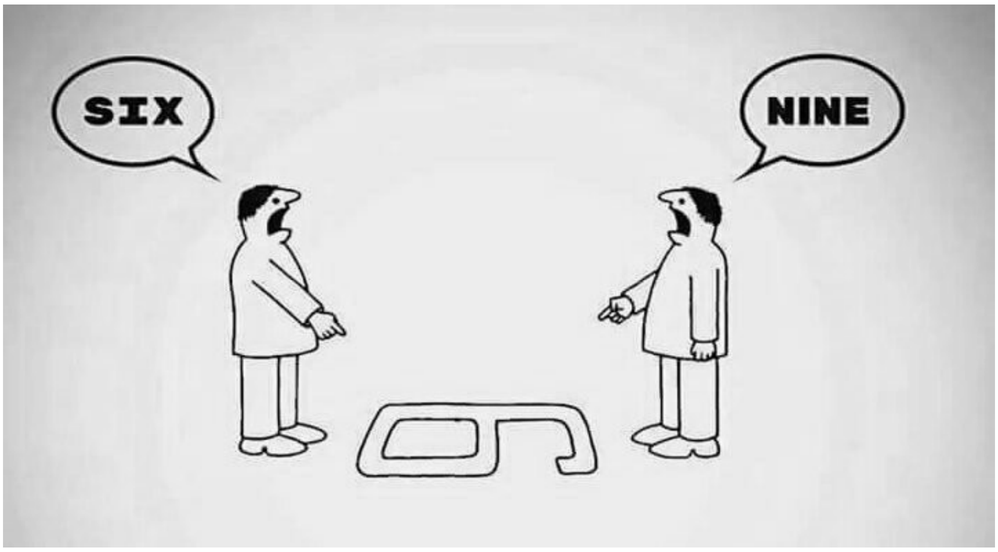
November 2022

Team ID: PNT2022TMID09373
Github Project I D: IBM-EPBL/IBM-Project-27786-1660065920

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

Artificial intelligence and computer technology both heavily rely on machine learning and deep learning. Human effort in identifying, learning, making predictions, and many other areas can be decreased with the application of deep learning and machine learning. The ability of computer systems to recognise handwritten digits from various sources, such as photographs, papers, and so on, is known as handwritten digit recognition. The goal of this project is to enable users to utilise machine learning to eliminate manual digit recognition jobs. Digit recognition systems are able to identify numbers from a variety of sources, including emails, bank checks, papers, images, etc. They can also be used in a variety of real-world situations, such as online handwriting recognition on computer tablets or systems, identifying vehicle licence plates, processing bank cheque amounts, and reading numbers from forms that have been filled out by hand.



CHAPTER 2
OBJECTIVE

The objective of this project is to classify the handwritten digits which is written in the paper and displays which digit is written using the python language.

CHAPTER 3
PROJECT DESIGN AND PLANNING

3.1 IDEATION PHASE

3.1.1 Literature Survey

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

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

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

An Ancient And Improved Scheme For Handwritten Digit Recognition Based On ConvolutionalNeuralNetwork (2019)

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

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

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

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

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

Handwritten Digit Recognition Using Machine And Deep Learning Algorithms (2021)

Pashine, Samay and Dixit, Ritik and Kushwah, Rishika

In this study, they developed three deep and machine learning-based models for handwritten digit recognition using MNIST datasets. To determine which model was the most accurate, they compared them based on their individual properties. Support vector machines are among the simplest classifiers, making them faster than other algorithms and providing the highest training accuracy rate in this situation. However, due to their simplicity, SVMs cannot categorize complicated and ambiguous images as accurately as MLP and CNN algorithms can. In their research, they discovered that CNN produced the most precise outcomes for handwritten digit recognition. This led them to the conclusion that CNN is the most effective solution for all types of prediction issues, including those using picture data. Next, by comparing the execution times of the algorithms, they determined that increasing the number of epochs without changing the configuration of the algorithm is pointless due to the limitation of a certain model, and they discovered that beyond a certain number of epochs, the model begins overfitting the dataset and provides biased predictions.

3.1.2 Problem Statement

- The problem statement is to classify handwritten digits. The goal is to take an image of a handwritten digit and determine what that digit and character is.

- It is easy for the human to perform a task accurately by practicing it repeatedly and memorizing it for the next time. Human brain can process and analyse images easily. Also, recognize the different elements present in the images.

- The goal is to correctly identify digits from a dataset of tens of thousands of handwritten images and experiment with different algorithms to learn first-hand what works well and how techniques compare

- 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.

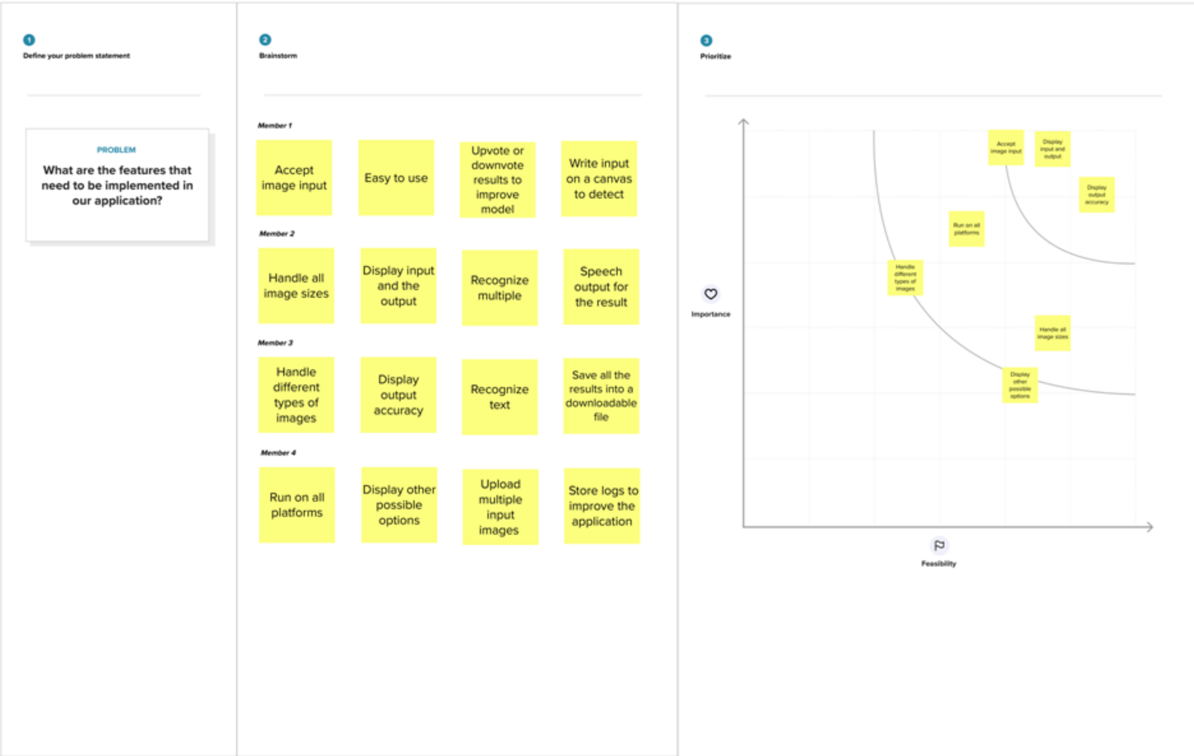
- The handwritten digit recognition system is a way to tackle this problem which uses the image of a digit and recognizes the digit present in the image. Convolutional Neural Network model created using Python library over the MNIST dataset to recognize handwritten digits .

- Handwriting number recognition is a challenging problem researchers had been research into this area for so long especially in the recent years

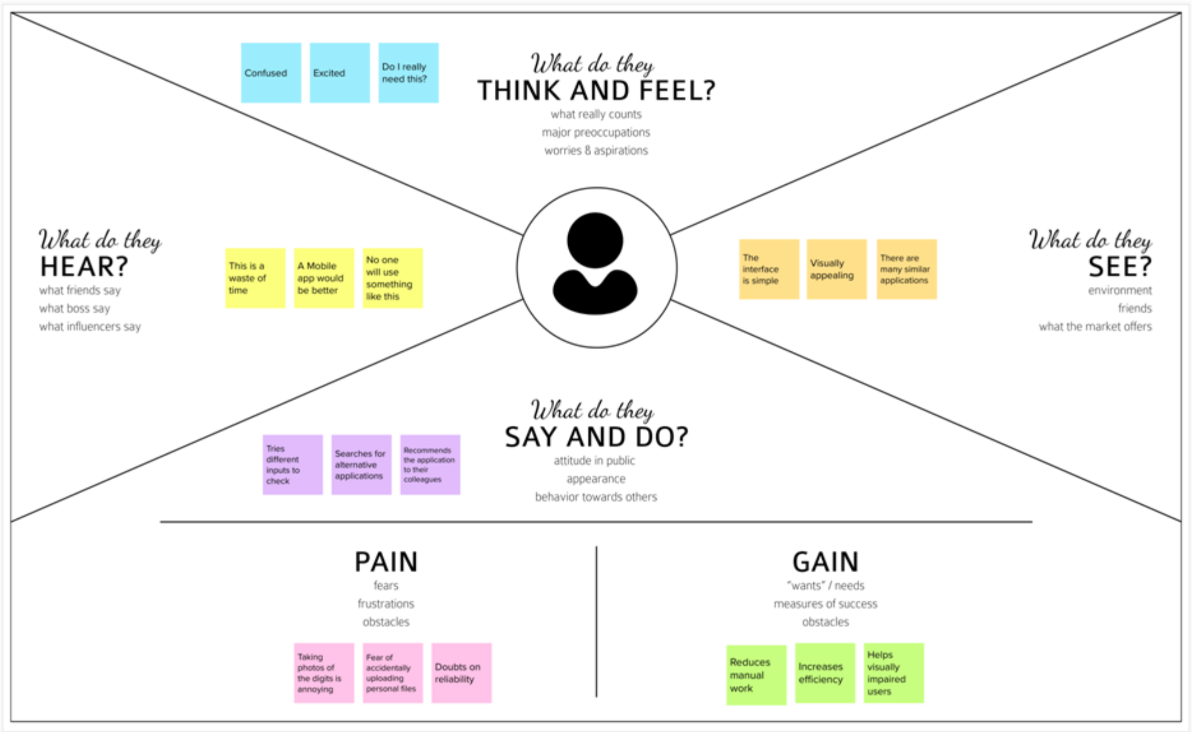
QUESTION	DESCRIPTION
What does the problem affect?	Handwriting recognition tends to have problems when it comes to accuracy. People can struggle to read others' handwriting. How, then, is a computer going to do it? The issue is that there's a wide range of handwriting – good and bad. This makes it tricky for programmers to provide enough examples of how every character might look.
What are the boundaries of the problem?	As the manually written digits aren't of a comparable size, thickness, position and direction, numerous difficulties need to be taken into consideration to decide the problem of handwritten digit recognition and it also involves the difficulty of visual pattern recognition.
What is the issue?	The handwritten digits are not always of the same size, width, orientation and justified to margins as they differ from writing of person to person, so the general problem would be while classifying the digits due to the similarity between digits such as 1 and 7, 5 and 6, 3 and 8, 2 and 5, 2 and 7, etc.

When does the issue occur?	Perhaps the most obvious problem when processing handwritten forms during the data capture process is poor quality or illegible handwriting. We all know the old stereotype about doctors' handwriting, so trying to perform accurate data capture and validation on this type of form-filling may result in little meaningful data being extracted.
Where is the issue occurring?	During the data capture validation stages of any forms processing activity,all required text fields are processed which involves recognition and extracting the written characters.
Why is it important that we fix the problem?	The high variance in handwriting styles across people and poor quality of the handwritten text compared to printed text pose significant hurdles in converting it to machine readable text. Nevertheless it's a crucial problem to solve for multiple industries like healthcare, insurance and banking.

3.1.3 Brainstorm

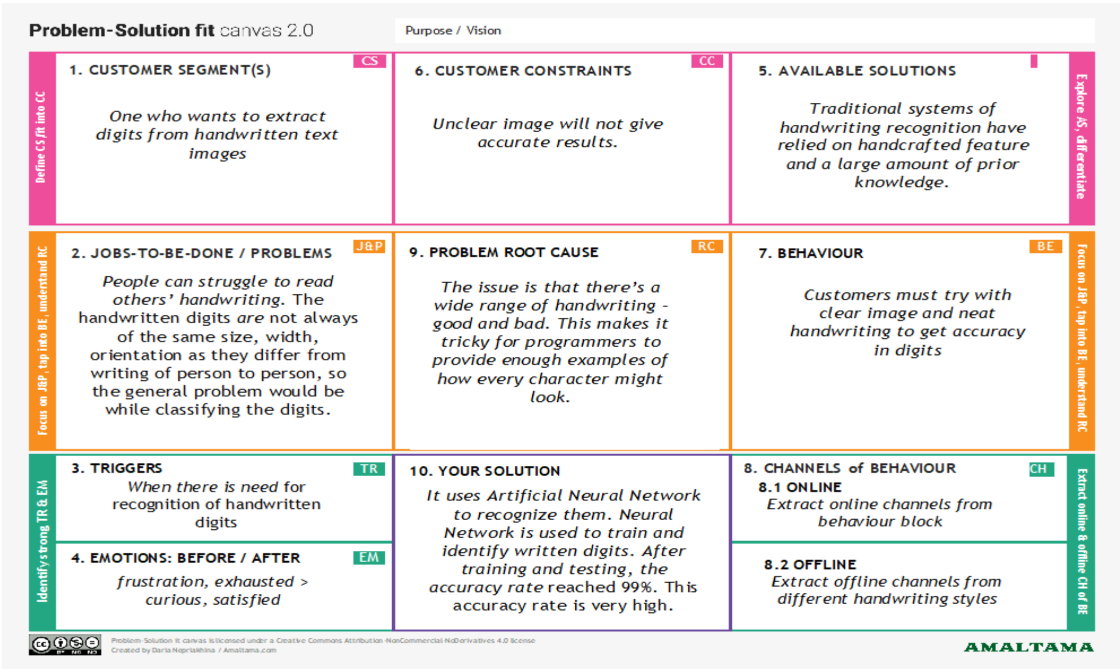


3.1.4 Empathy Map



3.2 Project Design Phase - I

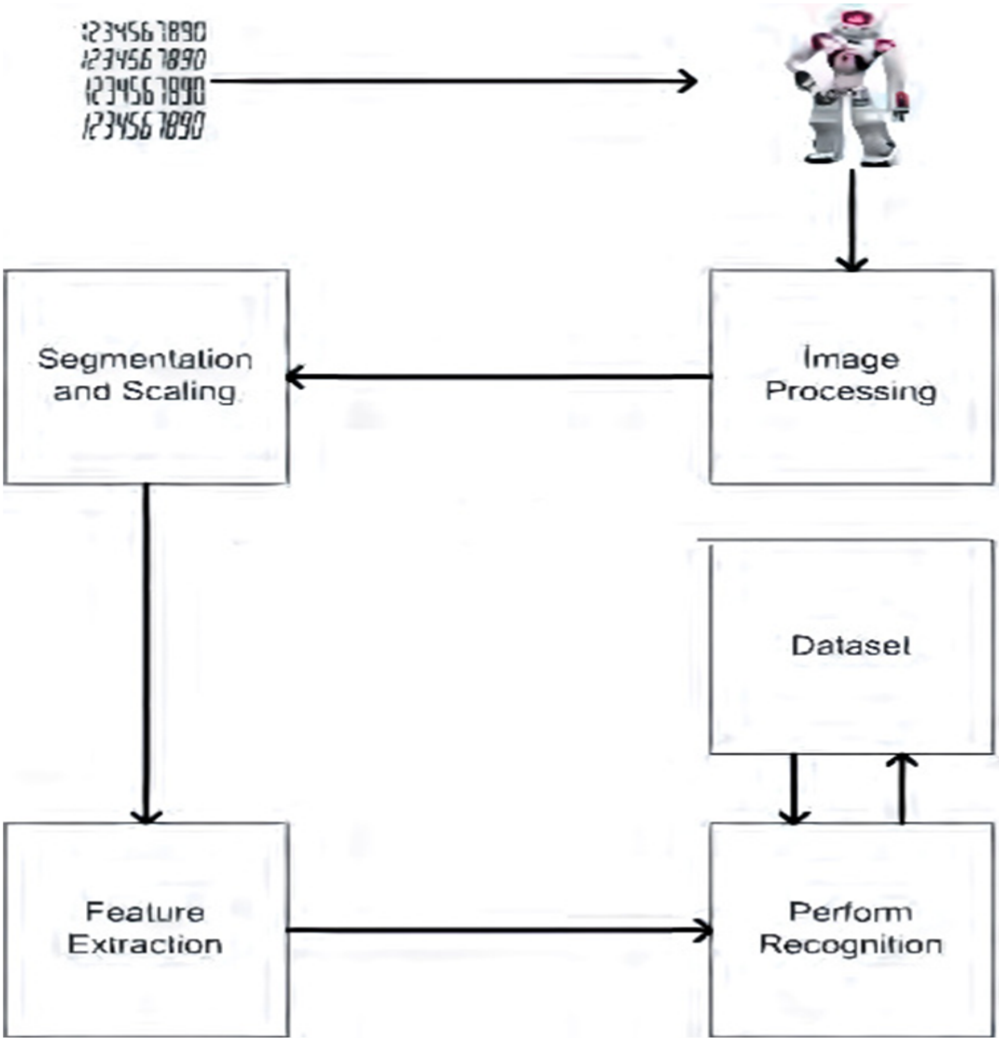
3.2.1 Problem Solution Fit



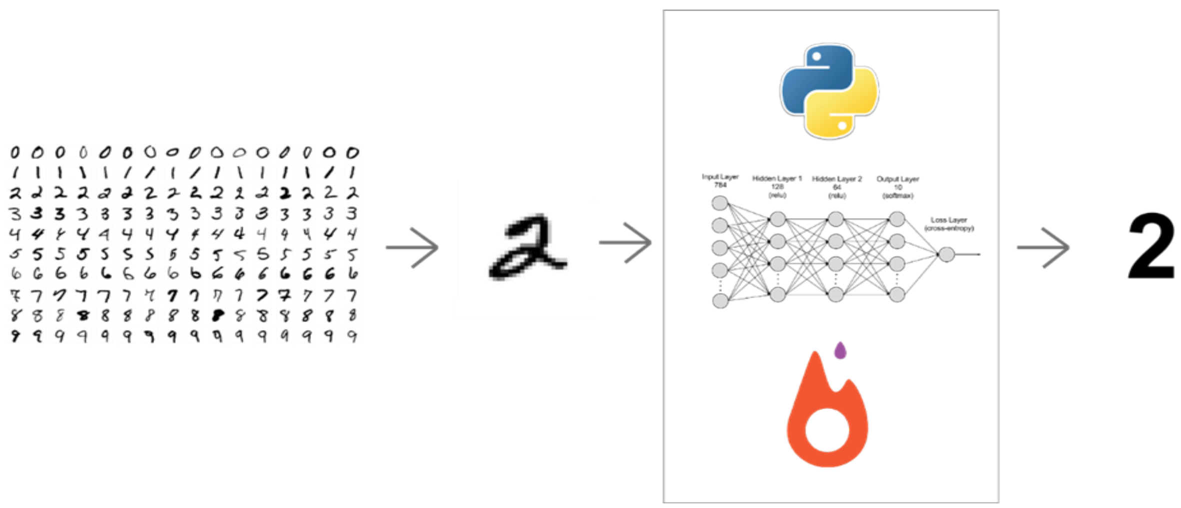
3.2.2 Proposed Solution

S.NO	PARAMETER	DESCRIPTION
1	Problem Statement	To create an application that recognizes handwritten digits
2	Idea / Solution Description	The application takes an image as the input and accurately detects the digits in it.
3	Novelty / Uniqueness	Instead of recognizing every text, the application accurately recognizes only the digits
4	Social Impact / Customer Satisfaction	This application reduces the manual tasks that need to be performed. This improves productivity in the workplace.
5	Business Model	The application can be integrated with traffic surveillance cameras to recognize vehicle number plates The application can be integrated with Postal systems to recognize the pin codes effectively
6	Scalability of the Solution	The application can easily be scaled to accept multiple inputs and process them parallelly to further increase efficiency

3.2.3 Solution Architecture

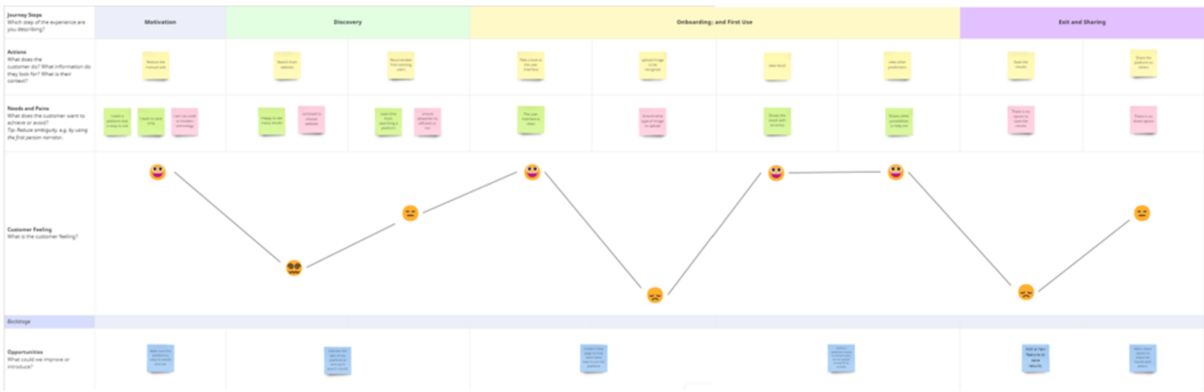


MNIST dataset processing with python



3.3 PROJECT DESIGN PHASE -II

3.3.1 Customer Journey Map



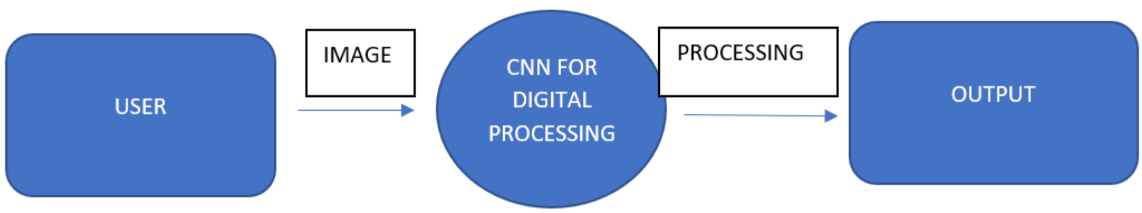
3.3.2 Data Flow Diagram and User Stories

DATA FLOW DIAGRAM

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

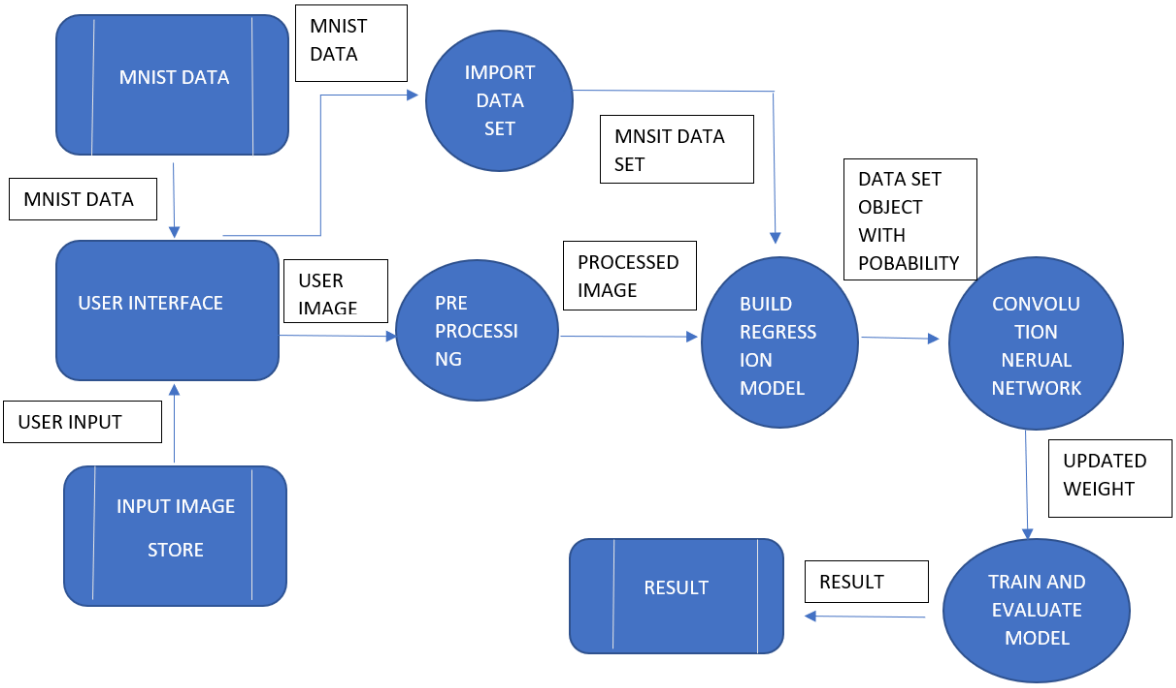
DFD Level-0

The DFD Level-0 consists of two external entities, the UI and the Output, along with a process, representing the CNN for Digit Recognition .Output is obtained after processing.



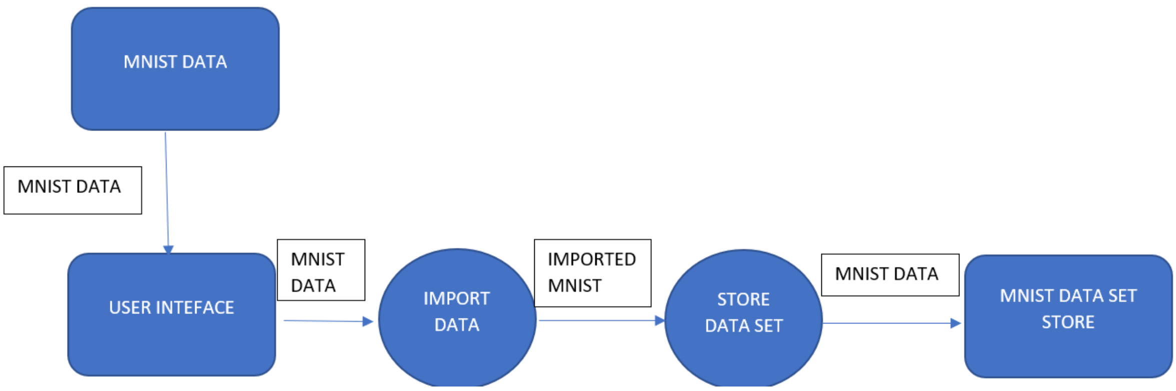
DFD Level-1

The DFD Level-1 consists of 2 external entities, the GUI and the Output, along with five process blocks and 2 data stores MNIST data and the Input image store, representing the internal workings of the CNN for Digit Recognition System. Process block imports MNIST data from library. Process block imports the image and process it and sends it to block where regression model is built. It sends objects with probabilities to CNN where weights are updated and multiple layers are built. Block trains and evaluates the model to generate output.



DFD Level-2

The DFD Level-2 for import data(figure 4) consists of two external data and one entity UI along with three process blocks, representing the three functionalities of the CNN for Digit Recognition System. It imports data from MNIST data store and stores on the system.



USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-2
		USN-3	As a user, I can register for the application through gmail or facebook	I can register & access the dashboard with Facebook Login	Medium	Sprint-2
	Login	USN-4	As a user, I can log into the application by entering email & password	I can login to the application	High	Sprint-1
	Dashboard	USN-5	Go to dashboard and refer the content about our project	I can read instructions also and the home page is user-friendly.	Low	Sprint-1
	Upload Image	USN-6	As a user, I can able to input the images of digital documents to the application	As a user, I can able to input the images of digital documents to the application	High	Sprint-3
	Predict	USN-7	As a user I can able to get the recognised digit as output from the images of digital documents or images	I can access the recognized digits from digital document or images	High	Sprint-3
		USN-8	As a user, I will train and test the input to get the maximum accuracy of output.	I can able to train and test the application until it gets maximum accuracy of the result.	Medium	Sprint-4
Customer (Web user)	Login	USN-9	As a user, I can use the application by entering my email, password.	I can access my account	Medium	Sprint-4
Customer Care Executive	Dashboard	USN-10	upload the image	Recognize and get the output	High	Sprint-1
Administrator	Security	USN-11	updated the features	checking the security	Medium	Sprint-1

3.3.3 Solution Requirement

FUNCTIONAL REQUIREMENTS

FR.NO	FUNCTIONAL REQUIREMENTS	SUB REQUIREMENTS
FR-1	Model Creation	Get access the MNIST dataset
		Analyze the dataset
		Define a CNN model
		Train and Test the Model
FR-2	Application Development	Create a website to let the user recognize handwritten digits.
		Create a home page to upload images
		Create a result page to display the results
		Host the website to let the users use it from anywhere
FR-3	Input Image Upload	Let users upload images of various formats.
		Let users upload images of various size
		Prevent users from uploading unsupported image formats
		Pre-Process the image to use it on the model
		Create a database to store all the input images

FR-4	Display Results	Display the result from the model
		Display input image
		Display accuracy the result
		Display other possible predictions with their respective accuracy

NON-FUNCTIONAL REQUIREMENTS

NFR	NON-FUNCTIONAL REQUIREMENTS	DESCRIPTION
NFR-1	Usability	The application must be usable in all devices
NFR-2	Security	The application must protect user uploaded image
NFR-3	Reliability	The application must give an accurate result as much as possible
NFR-4	Performance	The application must be fast and quick to load up
NFR-5	Availability	The application must be available to use all the time
NFR-6	Scalability	The application must scale along with the user base

3.3.4 Technology Stack

Technical Architecture for Handwritten Digit Recognition System

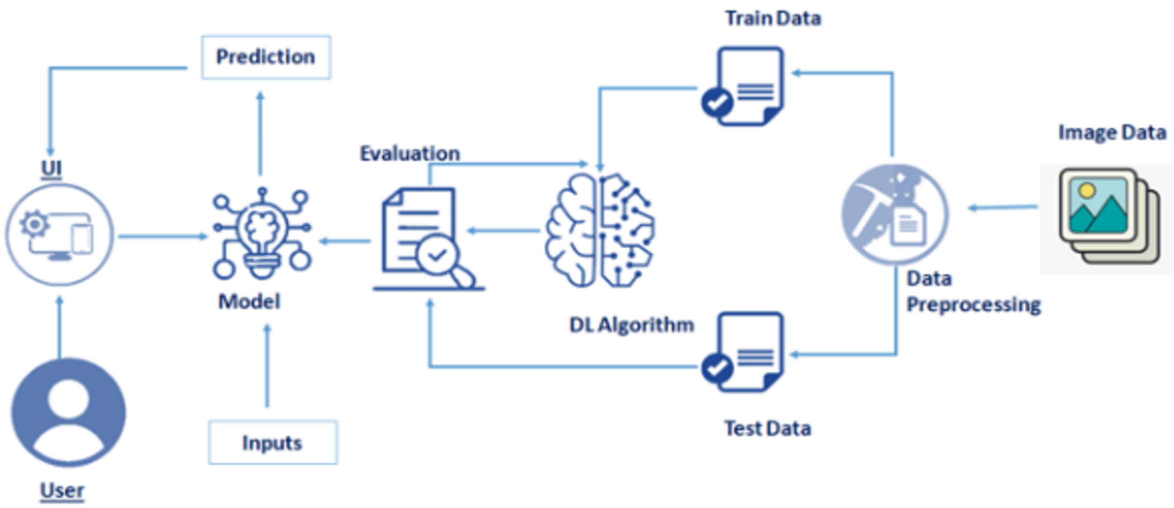


Table-1: Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	User interacts with the application using a web app	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic	Login to access the application	Java / Python
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	Storage of user files of handwritten image	IBM Block Storage or Other Storage Service or Local Filesystem
10.	Machine Learning Model	Machine learning model is used to identify the handwritten image uploaded by users	Object Recognition Model, etc.
11.	Infrastructure (Server / Cloud)	Application Deployment on Local System / AI Local Server Configuration AI Server Configuration	Local, Cloud Foundry, Kubernetes, etc.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Machine learning frameworks is used to train a predictive model	PyTorch, Open-cv
2.	Security Implementations	The system should automatically be able to authenticate all users with their unique username and password	Password based login, Authorization
3.	Scalable Architecture	The website traffic limit must be scalable enough to support 2 lakhs users at a time	3-tier
4.	Availability	The system functionality and services are available for use with all operations.	distributed servers
5.	Performance	The application can give response to requests within 5 sec. It uses fewer features to train the neural network, which results in faster convergence.	number of requests per sec

3.4 PROJECT PLANNING PHASE

3.4.1 Project Planning

Product Backlog, Sprint Schedule, and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	As a user, I can collect the dataset from various resources with different handwritings.	10	Low	Suvetha Sri RP, Swetha B, Uma Sruthi P, Varshini S
Sprint-1	Data Preprocessing	USN-2	As a user, I can load the dataset, handling the missing data, scaling and split data into train and test.	10	Medium	Suvetha Sri RP, Swetha B, Uma Sruthi P, Varshini S
Sprint-2	Model Building	USN-3	As a user, I will get an application with ML model which provides high accuracy of recognized handwritten digit.	5	High	Suvetha Sri RP, Swetha B, Uma Sruthi P, Varshini S
Sprint-2	Add CNN layers	USN-4	Creating the model and adding the input, hidden, and output layers to it.	5	High	Suvetha Sri RP, Swetha B, Uma Sruthi P, Varshini S
Sprint-2	Compiling the model	USN-5	With both the training data defined and model defined, it's time to configure the learning process.	2	Medium	Suvetha Sri RP, Swetha B, Uma Sruthi P, Varshini S
Sprint-2	Train & test the model	USN-6	As a user, let us train our model with our image dataset.	6	Medium	Suvetha Sri RP, Swetha B, Uma Sruthi P, Varshini S
Sprint-2	Save the model	USN-7	As a user, the model is saved & integrated with an android application or web application in order to predict something.	2	Low	Suvetha Sri RP, Swetha B, Uma Sruthi P, Varshini S
Sprint-3	Building UI Application	USN-8	As a user, I will upload the handwritten digit image to the application by clicking a upload button.	5	High	Suvetha Sri RP, Swetha B, Uma Sruthi P, Varshini S
Sprint-3		USN-9	As a user, I can know the details of the fundamental usage of the application.	5	Low	Suvetha Sri RP, Swetha B, Uma Sruthi P, Varshini S
Sprint-3		USN-10	As a user, I can see the predicted / recognized digits in the application.	5	Medium	Suvetha Sri RP, Swetha B, Uma Sruthi P, Varshini S
Sprint-4	Train the model on IBM	USN-11	As a user, I train the model on IBM and integrate flask/Django with scoring end point.	10	High	Suvetha Sri RP, Swetha B, Uma Sruthi P, Varshini S
Sprint-4	Cloud Deployment	USN-12	As a user, I can access the web application and make the use of the product from anywhere.	10	High	Suvetha Sri RP, Swetha B, Uma Sruthi P, Varshini S

Sprint Delivery Plan

Project Tracker, Velocity & Burndown Chart

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

Velocity:

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

Average 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.