19L039 – PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTREPRENEURSHIP

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM

IBM TEAM ID: PNT2022TMID12771

KIRUTHIKA S (19L122)

SIVA SURYA S (19L134)

SWATI M (19L142)

KALITHASAN K (20L406)

Project report submitted in partial fulfilments of the requirements for the degree of

BACHELOR OF ENGINEERING

Branch: ELECTRONICS AND COMMUNICATION ENGINEERING

NOVEMBER 2022

PSG COLLEGE OF TECHNOLOGY

(Autonomous Institution)

COIMBATORE – 641 004.

PSG COLLEGE OF TECHNOLOGY

(Autonomous Institution**)**

COIMBATORE – 641 004

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION SYSTEM

Bonafide record of work done by

KIRUTHIKA S (19L122)

SIVA SURYA S (19L134)

SWATI M (19L142)

KALITHASAN K (20L406)

Project report submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF ENGINEERING

Branch: ELECTRONICS AND COMMUNICATION ENGINEERING

of Anna University

NOVEMBER 2022

……...…….	**……..…….**
Dr.A.KANNAMMAL	**Dr.V.KRISHNAVENI**
Faculty Guide	Head of the Department

……..…….	**…….…….**
PRADEEPTHI	**Dr.T.KESAVAMURTHY**
Industry Mentor	Internal Evaluator

CONTENTS

CHAPTER **Page No**

CHAPTER – 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 analyzed by the model and the detected result is returned on to UI.

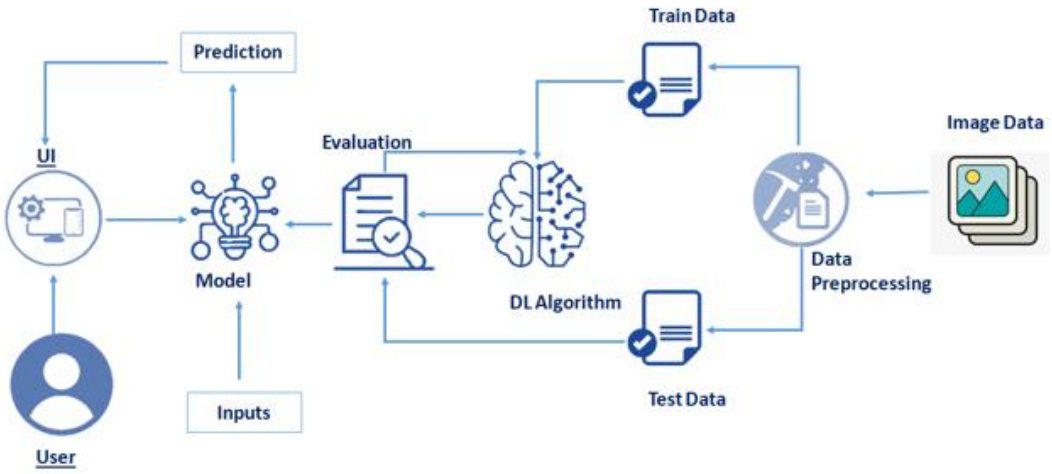


Figure 1.1. Technical Architecture

Handwritten digit recognition is the ability of a computer system to recognize the handwritten inputs like digits, characters etc. from a wide variety of sources like emails, papers, images, letters etc. This has been a topic of research for decades. Some of the research areas include signature verification, bank check processing, postal address interpretation from envelopes etc. 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.

1.2 Purpose

To purpose of a handwritten text recognition system is to convert the images into machine readable formats and then convert into audio signal for blind people to hear the audio. The Blind people are not able to read the digits and handwritten digits so we found one solution for that problem. So, we are going to use some Deep Learning and Machine Learning algorithms to convert the handwritten digits of images into computer understandable images and then convert into audio. The main challenge is to convert the handwriting style of a different people into computer recognising format as an input of image. The next challenge is to convert the image into audio signal.

Digit recognition from handwritten images has received greater attention in research community of pattern recognition due to vast applications and ambiguity in learning methods. Primarily, two steps including digit recognition and feature extraction are required based on some classification algorithm for handwritten digit recognition. The aim of the proposed endeavour was to make the path toward digitalization clearer by providing high accuracy and faster computational for recognizing the handwritten digits. The present research employed convolutional neural network as classifier, MNIST as dataset with suitable parameters for training and testing and DL4J framework for hand written digit recognition.

CHAPTER – 2
LITERATURE SURVEY

2.1 Existing Problem

Almost everyone around as have different handwriting. Reading it might be easy for you, but when it comes to extract information out of it digitally might be bit tricky. Well not anymore, in this tutorial I will guide you with how you can use TensorFlow to train your machine with Handwritten Digit Recognition dataset with 98% accuracy. You can later modify it train on your own custom dataset.

MNIST ("Modified National Institute of Standards and Technology") is considered an unofficial computer vision "hello-world" dataset. This is a collection of thousands of handwritten pictures used to train classification models using Machine Learning techniques. As a part of this problem statement, we will train a multi-layer perceptron using TensorFlow to recognize the handwritten digits.

2.2 References

2.2.1 - PAPER – 1

TITLE: HDSR-Flor: A Robust End-to-End System to Solve the Handwritten Digit String Recognition Problem in Real Complex Scenarios

AUTHOR: Arthur Flor De Sousa Neto, Byron Leite Dantas Bezerra (Member, IEEE), Estanislau Baptista Lima and Alejandro Héctor Toselli

YEAR OF PUBLICATION: 2020

JOURNAL NAME: IEEE Access

DESCRIPTION:

Automatic handwriting recognition systems are of interest for academic research fields and for commercial applications. Recent advances in deep learning techniques have shown dramatic improvement in relation to classic computer vision problems, especially in Handwritten Text Recognition (HTR). However, several approaches try to solve the problem of deep learning applied to Handwritten Digit String Recognition (HDSR), where it has to deal with the low number of trainable data, while learning to ignore any writing symbol around the digits (noise). In this context, we present a new optical model architecture (Gated-CNN-BGRU), based on HTR workflow, applied to HDSR. The International Conference on Frontiers of Handwriting Recognition (ICFHR) 2014 competition on HDSR were used as baselines to evaluate the effectiveness of our proposal, whose metrics, datasets, and recognition methods were adopted for fair comparison. Furthermore, we also use a private dataset (Brazilian Bank

Check - Courtesy Amount Recognition), and 11 different approaches from the state-of-the-art in HDSR, as well as 2 optical models from the state-of-the-art in HTR. Finally, the proposed optical model demonstrated robustness even with low data volume (126 trainable data, for example), surpassing the results of existing methods with an average precision of 96.50%, which is equivalent to an average percentage of improvement of 3.74 points compared to the state-of-the-art in HDSR. In addition, the result stands out in the competition's CVL HDS set, where the proposed optical model achieved a precision of 93.54%, while the best result so far had been from Beijing group (from the competition itself), with 85.29%.

2.2.2 - PAPER – 2

TITLE: Deep Convolutional Self-Organizing Map Network for Robust Handwritten Digit Recognition

AUTHOR: Saleh Aly, (Associate Member, IEEE), And Sultan Almotairi

YEAR OF PUBLICATION: 2020

JOURNAL NAME: IEEE Access

DESCRIPTION:

Deep Convolutional Neural Networks (DCNN) are currently the predominant technique commonly used to learn visual features from images. However, the complex structure of most recent DCNNs impose two major requirements namely, huge labelled dataset and high computational resources. In this paper, we develop a new efficient deep unsupervised network to learn invariant image representation from unlabelled visual data. The proposed Deep Convolutional Self-Organizing Maps (DCSOM) network comprises a cascade of convolutional SOM layers trained sequentially to represent multiple levels of features. The 2D SOM grid is commonly used for either data visualization or feature extraction. However, this work employs high dimensional map size to create a new deep network. The N-Dimensional SOM (ND-SOM) grid is trained to extract abstract visual features using its classical competitive learning algorithm. The topological order of the features learned from ND-SOM helps to absorb local transformation and deformation variations exhibited in the visual data. The input image is divided into an overlapped local patch where each local patch is represented by the Ncoordinates of the winner neuron in the ND-SOM grid. Each dimension of the NDSOM can be considered as a non-linear principal component and hence it can be exploited to represent the input image using N-Feature Index Image (FII) bank. Multiple convolutional SOM layers can be cascaded to create a deep network structure. The output layer of the DCSOM network computes local histograms of each FII bank in the final convolutional SOM layer. A set of experiments using MNIST handwritten digit database and all its variants are conducted to evaluate the robust representation of the proposed DCSOM network. Experimental results reveal that the performance of DCSOM outperforms state-of-the-art

methods for noisy digits and achieve a comparable performance with other complex deep learning architecture for other image variations.

2.2.3 - PAPER – 3

TITLE: A Novel Learning Algorithm to Optimize Deep Neural Networks: Evolved Gradient Direction Optimizer (EVGO)

AUTHOR: Ibrahim Karabayir, Oguz Akbilgic, and Nihat Tas

YEAR OF PUBLICATION: 2020

JOURNAL NAME: IEEE Transaction on Neural Networks and Learning Systems

DESCRIPTION:

Gradient-based algorithms have been widely used in optimizing parameters of deep neural networks' (DNNs) architectures. However, the vanishing gradient remains as one of the common issues in the parameter optimization of such networks. To cope with the vanishing gradient problem, in this article, we propose a novel algorithm, evolved gradient direction optimizer (EVGO), updating the weights of DNNs based on the first-order gradient and a novel hyperplane we introduce. We compare the EVGO algorithm with other gradient-based algorithms, such as gradient descent, RMSProp, Adagrad, momentum, and Adam on the well-known Modified National Institute of Standards and Technology (MNIST) data set for handwritten digit recognition by implementing deep convolutional neural networks. Furthermore, we present empirical evaluations of EVGO on the CIFAR-10 and CIFAR-100 datasets by using the well-known AlexNet and ResNet architectures. Finally, we implement an empirical analysis for EVGO and other algorithms to investigate the behaviour of the loss functions. The results show that EVGO outperforms all the algorithms in comparison for all experiments. We conclude that EVGO can be used effectively in the optimization of DNNs, and, the pro-posed hyperplane may provide a basis for future optimization algorithms.

2.2.4 - PAPER – 4

TITLE: Deblur GAN-CNN: Effective Image Denoising and Recognition for Noisy Handwritten Characters

AUTHOR: Sarayut Gonwirat, and Olarik Surinta

YEAR OF PUBLICATION: 2022

JOURNAL NAME: IEEE Access

DESCRIPTION:

Many problems can reduce handwritten character recognition performance, such as image degradation, light conditions, low-resolution images, and even the quality of the capture devices. However, in this research, we have focused on the noise in the character images that could decrease the accuracy of handwritten character recognition. Many types of noise penalties influence the recognition performance, for example, low resolution, Gaussian noise, low contrast, and blur. First, this research proposes a method that learns from the noisy handwritten character images and synthesizes clean character images using the robust deblur generative adversarial network (Deblur GAN). Second, we combine the Deblur GAN architecture with a convolutional neural network (CNN), called Deblur GAN-CNN. Subsequently, two state-of-the-art CNN architectures are combined with Deblur GAN, namely DeblurGAN-DenseNet121 and DeblurGAN-MobileNetV2, to address many noise problems and enhance the recognition performance of the handwritten character images. Finally, the Deblur GAN-CNN could transform the noisy characters to the new clean characters and recognize clean characters simultaneously. We have evaluated and compared the experimental results of the proposed Deblur GAN-CNN architectures with the existing methods on four handwritten character datasets: n-THI-C68, n-MNIST, THI-C68, and THCC-67. For the n-THI-C68 dataset, the Deblur GAN-CNN achieved above 98% and outperformed the other existing methods. For the n-MNIST, the proposed Deblur GAN-CNN achieved an accuracy of 97.59% when the AWGN+Contrast noise method was applied to the handwritten digits. We have evaluated the Deblur GAN-CNN on the THCC-67 dataset. The result showed that the proposed Deblur GAN-CNN achieved an accuracy of 80.68%, which is significantly higher than the existing method, approximately 10%.

2.2.5 - PAPER – 5

TITLE: Fragmented handwritten digit recognition using grading scheme and fuzzy rules

AUTHOR: Jyotismita Chaki and Nilajan Dey

YEAR OF PUBLICATION: 2020

JOURNAL NAME: IEEE Access

DESCRIPTION:

The handwritten digit recognition issue turns into one of the well-known issues in machine learning and computer vision applications. Numerous machine learning methods have been utilized to resolve the handwritten digit recognition problem. However, sometimes the digit is not completely present in the image due to issues related to scanning or environmental conditions (light, illumination, dirt, etc.). Although different efficient methodologies of handwritten digit recognition are proposed, there is not much work done on fragmented handwritten digit recognition. The objective of the proposed research work is to handle this circumstance to assemble a consistent digit recognition system that can precisely handle three

types (English, Bangla, and Devanagari) of fragmented handwritten digit images. To solve the confusion, a technique is created to classify handwritten digits based on geometrical functions that are utilized to calculate handwritten digit features to assess if a digit belongs to a specific class. A grading scheme and a set of specified fuzzy rules determine the performance of classification. Experiments have been directed on the three familiar datasets, i.e., MNIST database (English), NumtaDB (Bangla) and Deva numeral database (Devanagari). Since fragmented digit delivers a lesser amount of information, the work also attempts to create a tentative size threshold above which outcomes become erratic and whether such thresholds are standardized or vary depending on other factors. Since the fragmented handwritten digital image does not have a public database, a method is formed to produce repeatable fragmented handwritten digital images from the entire image. Experimental outcomes validate that the proposed approach is effective in recognizing fragmented handwritten digits to an acceptable degree of fragmentation.

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 sizes and shapes. 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 database to recognize handwritten digits.

Handwritten Digit Recognition is the capability of computer to fete the mortal handwritten integers from different sources like images, papers, touch defences, etc, and classify them into 10 predefined classes (0-9). This has been a content of bottomless-exploration in the field of deep literacy. 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. Vector Machine, Multilayer Perceptron, and Convolutional Neural Network. The comparison between these algorithms is carried out on the base of their delicacy, crimes and testing-training time corroborated by plots and maps that have been constructed using matlablib for visualization.

CHAPTER – 3

IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

The handwritten digit recognition is not a new technology, but it has not gained public attention until recently. The ultimate goal of designing a handwritten recognition system with an accuracy rate of 90% to 99% is quite illusionary, because even human beings are not able to recognize every handwritten text without any doubt.

Figure 3.1. Empathy Map

In the current age of digitization, handwriting recognition plays an important role in information processing. A lot of information is available on paper, and processing of digital files is cheaper than processing traditional paper files. The aim of a handwriting recognition system is to convert handwritten characters into machine readable formats. The main applications are vehicle license-plate recognition, postal letter-sorting services, Cheque truncation system (CTS) scanning and historical document preservation in archaeology departments, old documents automation in libraries and banks, etc.

3.2 Ideation & Brainstorming

The handwritten digit recognition is the capability of computer application to recognize the human hand written digit. It is hard task for the machine because handwritten digits are not perfect and can be made with many different shapes and sizes.

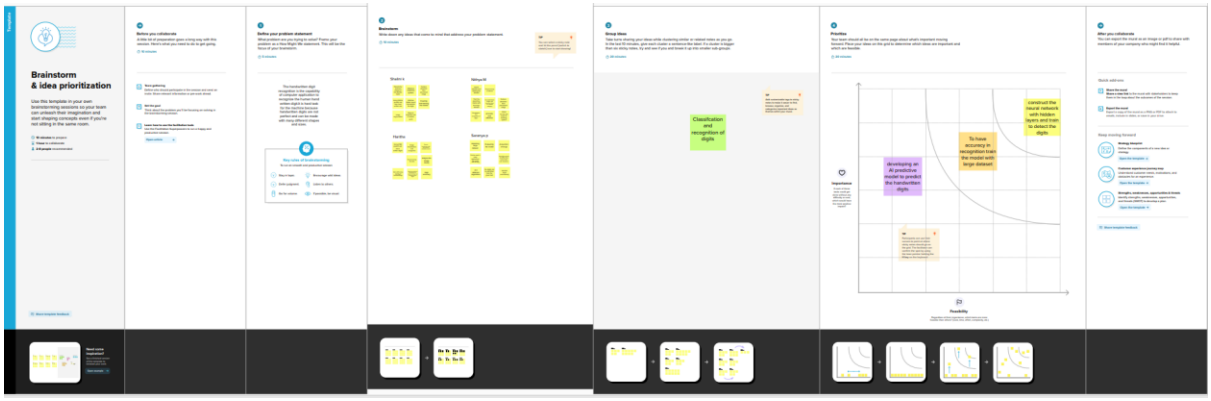


Figure 3.2. Project Ideation & Brain Storming

One of the very popular applications in computer vision is Handwritten Digits Classification or Recognition (HDR) in the field of character recognition. Digits like other universal symbols are widely used in technology, bank, OCR, analysing of digits in engineering, postal service, numbers in plate recognition, etc. They are some of the famous applications on HDR. There are 10 classes corresponding to the handwritten digits from ‘0’ to ‘9’ which are very depend on the handwritten. The main difficulty in the handwritten digit recognition is different handwritten style which is a very personal behaviour where there are a lot of models for numbers based on the angles, length of the segments, stress on some parts of numbers, etc.

3.3 Proposed Solution

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The problem statement aims at developing a novel handwritten recognition system using ML. The handwritten digit recognition system is a way to tackle the problem which uses the image of a digit and recognizes the digit present in the image .
2.	Idea / Solution description	Developing an AI predictive model to predict the handwritten digits and to construct a neural network with hidden layers and train to detect the digits.
3.	Novelty / Uniqueness	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
4.	Social Impact / Customer Satisfaction	Handwritten digits can be recognised easily without any strenuous efforts. This reduces time and improves productivity for people.
5.	Business Model (Revenue Model)	It is used in the detection of vehicle numbers, banks for reading cheques, post offices for arranging letters, and many other tasks.
6.	Scalability of the Solution	To attain higher performances in the domain of character recognition and pattern recognition, due to its excellent feature extraction and working as best classifier characteristics. There is no limit in the number of digits that can be recognized.

3.4 Problem Solution Fit

The best solution for handwritten digit recognition is Convolutional neural networks (CNNs) are very effective in perceiving the structure of handwritten characters/words in ways that help in automatic extraction of distinct features and make CNN the most suitable approach for solving handwriting recognition problems.

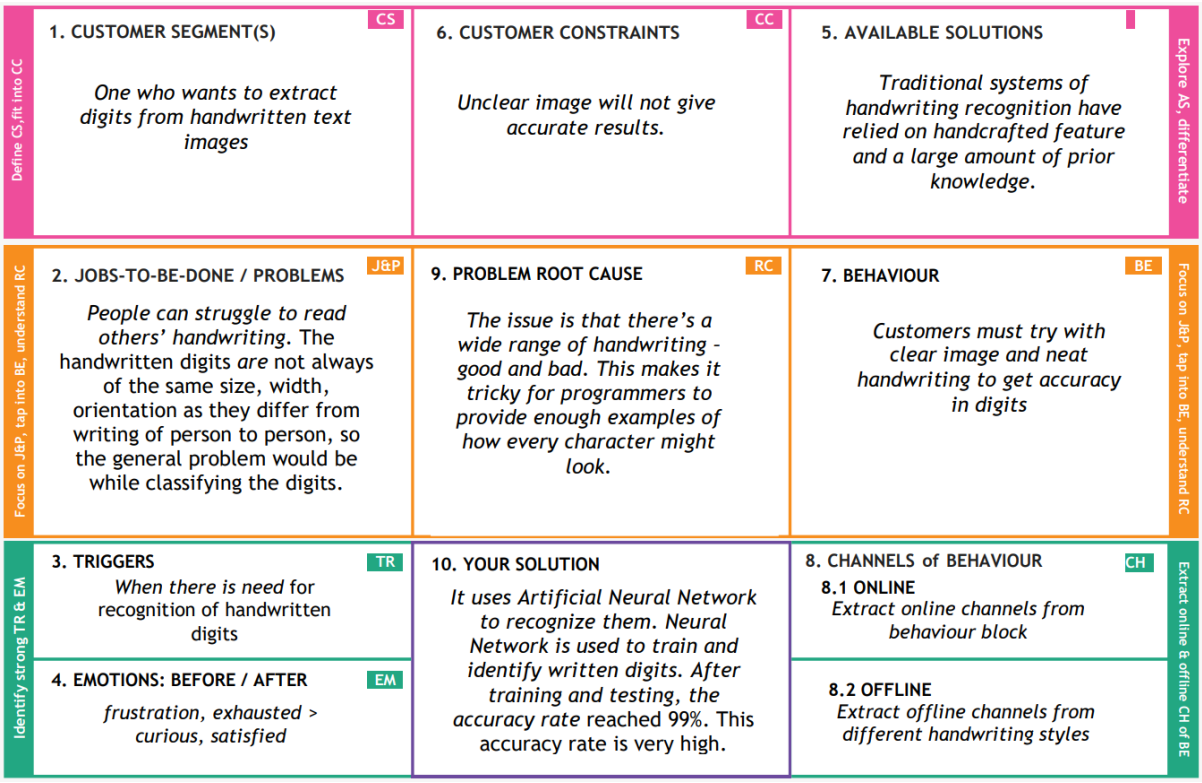


Figure 3.3. Problem Solution Fit

Machine learning and deep learning plays an important role in computer technology and artificial intelligence. With the use of deep learning and machine learning, human effort can be reduced in recognizing, learning, predictions and many more areas. This article presents recognizing the handwritten digits (0 to 9) from the famous MNIST dataset, comparing classifiers like KNN, PSVM, NN and convolution neural network on basis of performance, accuracy, time, sensitivity, positive productivity, and specificity with using different parameters with the classifiers.

CHAPTER – 4

REQUIREMENT ANALYSIS

4.1 Functional Requirement

S No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
2	User Confirmation	Confirmation via Email Confirmation via OTP
3	Login	Login using credentials
4	Upload Input	Upload image Upload via on-screen
5	Train	Multiple inputs to train
6	Test	Test via fresh data
7	Maintenance	Handle all user data
8	Update	Update if any new feature available

4.2 Non-Functional Requirement

S. No.	Non-Functional Requirement	Description
1	**Usability**	It will be easily accessible by the user. Simple and easy to understand.
2	**Security**	The data and input given by the user will be protected. Password-protected and only the particular user can alter their data.
3	**Reliability**	It is highly reliable and the accuracy can be increased with training.
4	**Performance**	Good performance with short time to run.
5	**Availability**	It is easily available on all platforms. Available on web.

6	**Scalability**	It is scalable and new features can be integrated. Multiple digits can be recognised at a time, real time recognition can be done.

CHAPTER - 5

PROJECT DESIGN

5.1 .DATA FLOW DIAGRAM

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

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.

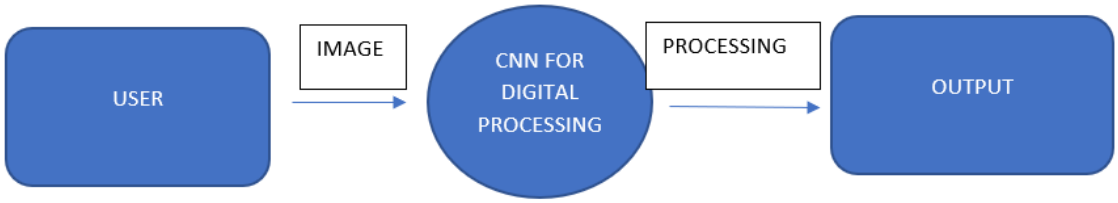


Figure 5.1. Data Flow diagram Level - 0

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.

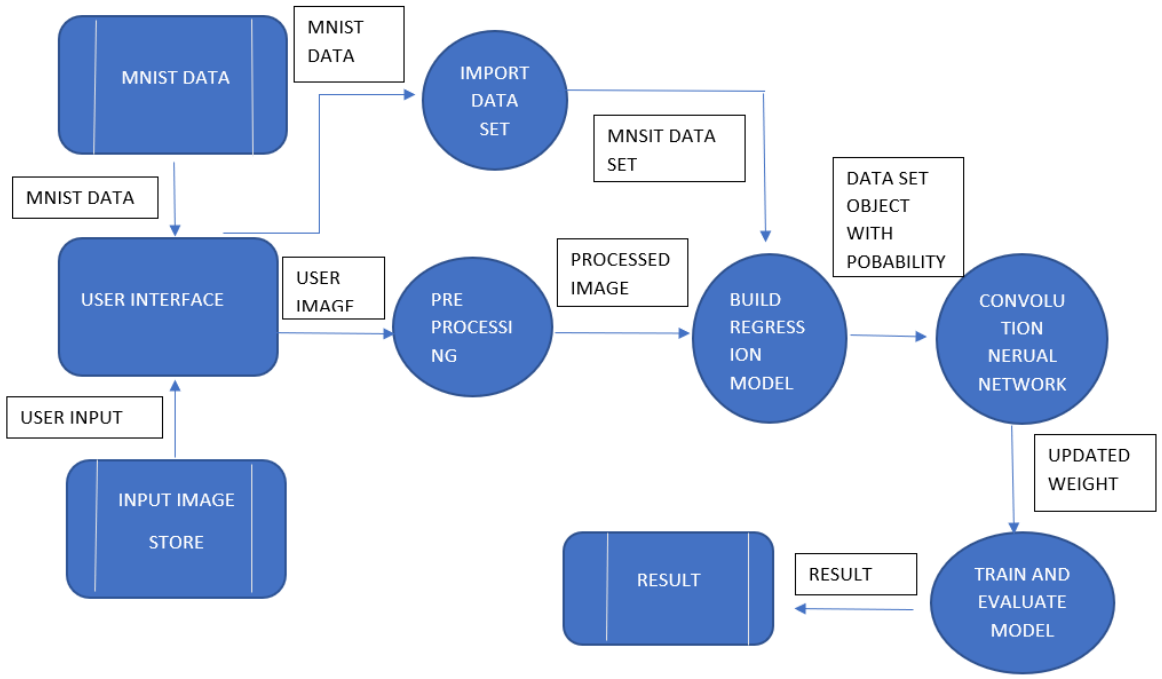


Figure 5.2. Data Flow diagram Level - 1

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.

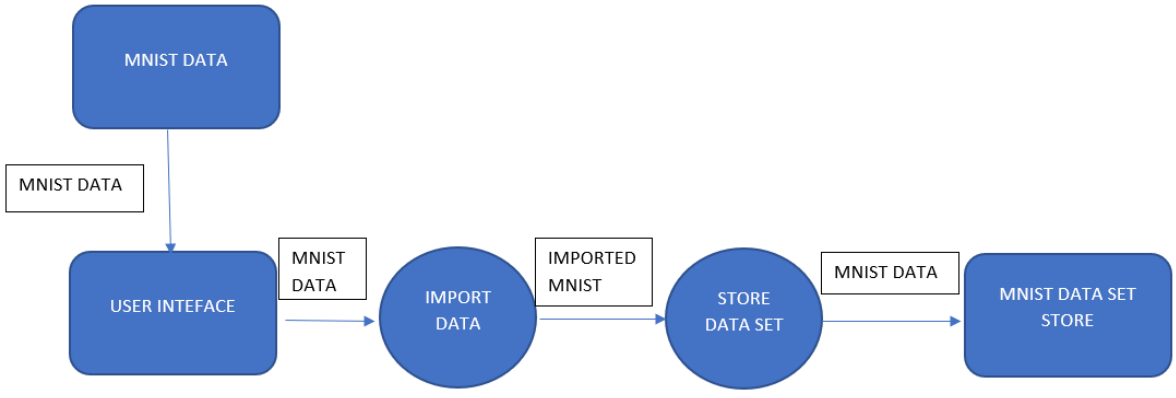


Figure 5.3. Data Flow diagram Level - 2

5.2 Solution & Technical Architecture

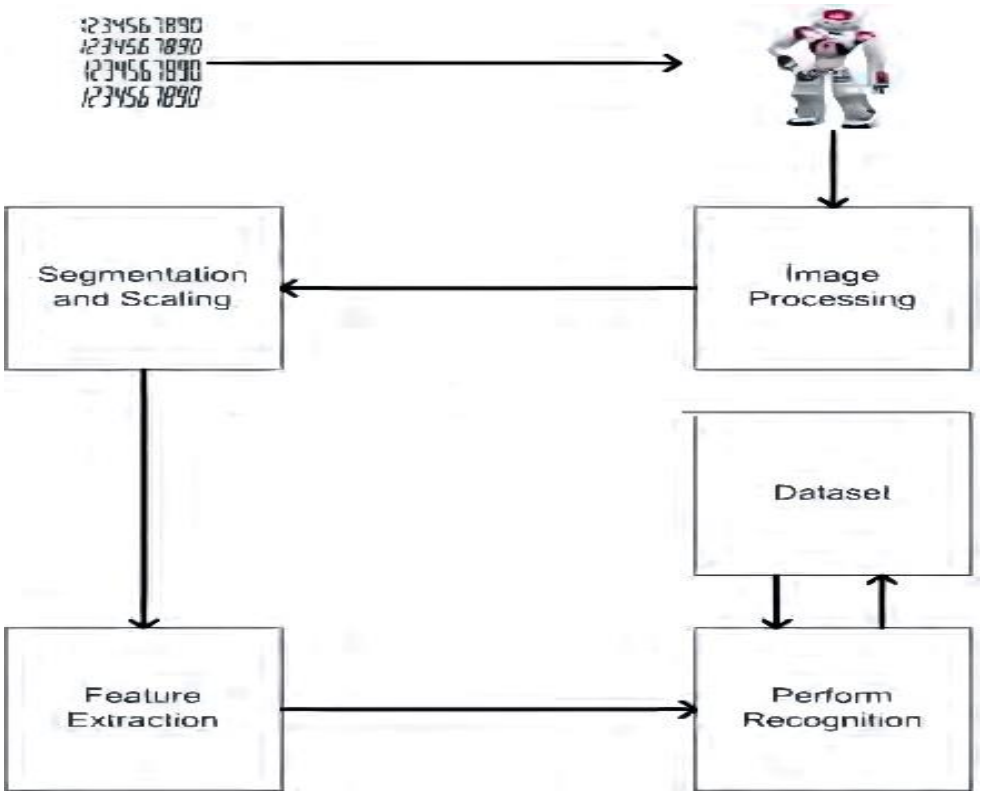


Figure 5.4. Solution and Technical Architecture

MNIST dataset processing with python

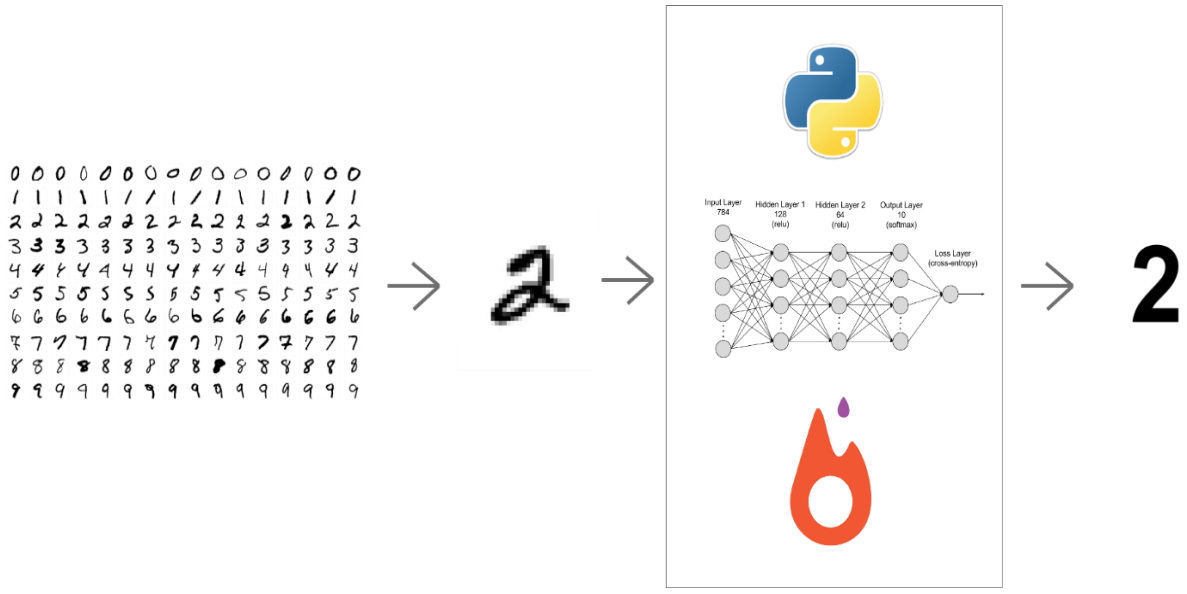


Figure 5.5. MNIST Dataset

5.3.USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-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

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	USN-1	As a user, I can collect the dataset from various resources with different handwritings.	10	Low	Swati M Kiruthika 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	Kalithasan K Siva Surya 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	Swati M Kiruthika S Kalithasan K
Sprint-2	Add CNN layers	USN-4	Creating the model and adding the input, hidden, and output layers to it.	5	High	Siva Surya S Swati M Kiruthika 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	Kalithasan K Siva Surya S
Sprint-2	Train & test the model	USN-6	As a user, let us train our model with our image dataset.	6	Medium	Swati M Kiruthika 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	Kalithasan K
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	Siva Surya S
Sprint-3		USN-9	As a user, I can know the details of the fundamental usage of the application.	5	Low	Swati M

Sprint-3		USN-10	As a user, I can see the predicted / recognized digits in the application.	5	Medium	Kiruthika S Kalithasan K
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	Siva Surya S Kalithasan K
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	Kiruthika S Swati M

6.2 Sprint Delivery Schedule

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

Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports from JIRA

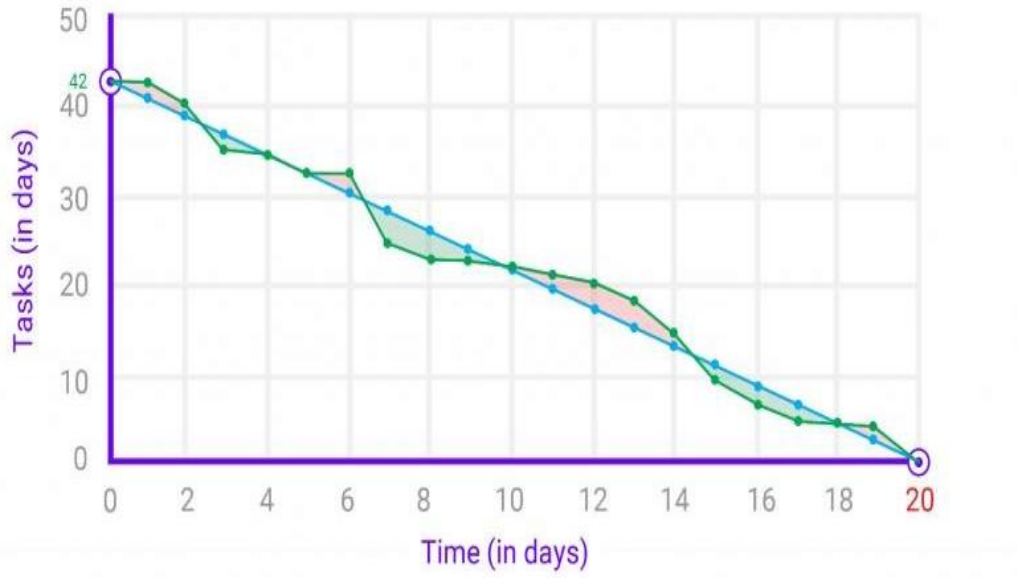
Velocity:

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.



CHAPTER -7

CODING & SOLUTIONING

7.1 Feature 1

ML depends heavily on data, without data, it is impossible for a machine to learn. It is the most crucial aspect that makes algorithm training possible. In Machine Learning projects, we need a training data set. It is the actual data set used to train the model for performing various actions. TensorFlow already has MNIST Data set so there is no need to explicitly download or create Dataset. The MNSIT dataset contains ten classes: Digits from 0-9. Each digit is taken as a class. The required libraries are imported which are required for the model to run. It is a dataset of 60,000 small square 28×28 pixel grayscale images of handwritten single digits between 0 and 9. The task is to classify a given image of a handwritten digit into one of 10 classes representing integer values from 0 to 9, inclusively. The dataset for this model is imported from the Keras module. The data is split into train and test. Using the training dataset, the model is trained and the testing dataset is used to predict the results. Basically, the pixel values range from 0-255. The value of each image is stored is y_train. The model is built with convolutional, pooling, and dense layers. The created model is then compiled and saved.

7.2 Feature 2

Only Python is used to create the web pages for the front end. An html page that takes in image files as input using form and submits to back end is created. A flask app is created using python flask, where it receives the image files from the templates, html pages and the prediction operation is done over this image. Later the predicted output is sent to the result page.

CHAPTER - 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
Predict_TC_OO5	Functional	Predict page	Verify user is able to navigate to the predict to and view the predicted result	User must be navigated to the predict page and must view the predicted result	Working as expected	Pass

8.2 User 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 Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	0	0	0	0	0

Test Case Analysis

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

CHAPTER - 9

RESULTS

9.1 Performance Metrics

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 64)        640

 conv2d_1 (Conv2D)           (None, 24, 24, 32)        18464

 flatten (Flatten)           (None, 18432)             0

 dense (Dense)               (None, 10)                184330

=================================================================
Total params: 203,434
Trainable params: 203,434
Non-trainable params: 0
_________________________________________________________________
None
```

CHAPTER - 10

ADVANTAGES AND DISADVANTAGES

Advantages

This approach has many advantages,

1) 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.
2) The generative models can perform recognition driven segmentation.
3) It saves times for arranging and sorting huge amount of data.
4) Only requires far less physical space than the storage of the physical copies.
5) Recognising multiple digits on a single frame using sequential model in Keras.
6) Data storage, for an example, there are many files, contracts and some personal records that contains some handwritten digits.
7) It reduces human effort and labour cost.
8) This can be used for sorting through mail by postal code.

Disadvantages

The main difficulty in the handwritten digits,

1) Recognition is different handwritten style which is a very personal behaviour.
2) There are a lot of models for numbers based on the angles, length of the segments, stress on some parts of numbers, etc.
3) The system build is complex and holds difficulty
4) The handwriting of every individual varies which proves to be a challenge for the system to predict
5) Possible unemployment of labour that is typical of technology growth
6) The accuracy is not guarantees and there is risk of errors.

CHAPTER - 11

CONCLUSION

Handwriting recognition is undoubtedly one of the most challenging areas of the pattern recognition. The goal of the project is to classify numeric samples which are mostly saved as digital images. Several pattern recognitions approaches have been applied to both online and offline handwriting recognition on the basis of unique patterns. The process of recognition consists of several steps such features extraction and recognition with voice alert. Handwritten Digit Recognition has various real-life time uses. It is used in the detection of vehicle number, banks for reading cheques, post offices for arranging letter, and many other tasks. The application of digit recognition includes in postal mail sorting, bank check processing, from data entry, etc.

The heart of problem lies within the ability to develop an efficient algorithm that can recognize hand written digit and which is submitted by users by the way of a scanner, tablet, and other digital devices. These systems also have some limitations too. Handwriting recognition tends to have problems when it comes to accuracy. People can struggle to read others' handwriting.

CHAPTER - 12

FUTURE SCOPE

The goal of the project is to convert the handwritten digits into audio. As a first step the recognition of digits have been done with an accuracy of around 92%. Further extension of the project includes extending the model to convert the handwritten digit into text file and then convert the text file into audio using text to speech module so that blind people will make use of the project. In the future, we can work on a denser or hybrid algorithm than the current set of algorithms with more manifold data to achieve the solutions to many problems. In future, the application of these algorithms lies from the public to high-level authorities, as from the differentiation of the algorithms above and with future development we can attain high-level functioning applications which can be used in the classified or government agencies as well as for the common people. Currently only the digits are recognized. In future the all the characters in all the language can be predicted with high accuracy rate.

CHAPTER - 13

APPENDIX

Source Code

Main Code:

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

UPLOAD_FOLDER = 'D:\Final_Deliverables\Final code\data'


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

model = load_model("./models/mnistCNN.h5")


@app.route('/')
def index():
    return render_template('index.html')


@app.route('/predict', methods=['GET', 'POST'])
def upload():
    if request.method == "POST":
        f = request.files["image"]
        filepath = secure_filename(f.filename)
        f.save(os.path.join(app.config['UPLOAD_FOLDER'], filepath))

        upload_img = os.path.join(UPLOAD_FOLDER, filepath)
        img = Image.open(upload_img).convert("L")  # convert image to
monochrome
        img = img.resize((28, 28))  # resizing of input image

        im2arr = np.array(img)  # converting to image
        im2arr = im2arr.reshape(1, 28, 28, 1)  # reshaping according to our
requirement

        pred = model.predict(im2arr)

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

        return render_template('predict.html', num=str(num[0]))


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

Index.HTML

```
<html>

<head>
  <title>IBM PROJECT</title>

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

  <link
href="https://fonts.googleapis.com/css2?family=Prompt:wght@600&display=swap
" 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&dis
play=swap" rel="stylesheet">
  <link
href="https://fonts.googleapis.com/css?family=Calistoga|Josefin+Sans:400,70
0|Pacifico&display=swap" rel="stylesheet">

  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.
css" integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
  <link rel="stylesheet" type= "text/css" href= "{{
url_for('static',filename='css/style.css') }}">

  <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+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min
.js" integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous"></script>
  <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js
" integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
  <script
src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest"></script>
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.c
ss">
  <script
src="https://cdn.jsdelivr.net/npm/jquery@3.6.0/dist/jquery.slim.min.js"></s
cript>
  <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js">
</script>
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.
min.js"></script>
```

```
</head>
<style>
    body{
     background-image: url('static/images/background.jpeg');
     background-repeat: no-repeat;
     background-size: cover;
    }
</style>

<script>
  function preview() {
    frame.src=URL.createObjectURL(event.target.files[0]);
}

    $(document).ready(function() {
          $('#clear_button').on('click', function() {
              $('#image').val('');
              $('#frame').attr('src',"");
            });
        });

</script>

<body>
    <h1>HandWritten Digit Recognizer</h1>
        <div class="container p-3 my-3 text-white">
            <p>TEAM ID : PNT2022TMID12771</p>
            <p>Team Members: </p>
            <p>19L122- Kiruthika S</p>
            <p>19L134- Siva Surya S</p>
            <p>19L142- Swati M</p>
            <p>20L406- Kalithasan K</p>
        </div>
        <section id="content">

            <div class="leftside">
            <form action="/predict" method="POST" enctype="multipart/form-
data">
            <label style="color:white;">Select a image:</label>
            <input id="image" type="file" name="image" accept="image/png,
image/jpeg" onchange="preview()"><br><br>
              <img id="frame" width="100px" height="100px"/>
              <div class="buttons_div">
                <button type="submit" class="btn btn-
light">Predict</button>
                <button type="button" class="btn btn-light">  Clear
 </button>
              </div>
            </form>
            </div>
      </section>


</body>

</html>
```

Predict.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Prediction</title>
</head>

<style>
    body{
     background-image: url('static/images/bc1.jpg');
     background-repeat: no-repeat;
     background-size: cover;
    }

    #rectangle{
     width:600px;
     height:150px;
     background-color: #000000;
     border-radius: 25px;
     position:absolute;
     box-shadow: 0px 0px 10px 5px white;
     top:25%;
     left:50%;
     transform:translate(-50%,-50%);
    }

    #num{
  text-align: center;
  font-size: 30px;
  margin: 0 auto;
  padding: 3% 5%;
  padding-top: 8%;
  color: white;
    }

</style>

<body>

    <div id="rectangle">
        <h1 id="num">Predicted Number is {{num}}</h1>
    </div>

</body>
</html>
```

Style.css

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

#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: rgb(0, 32, 112);
   background-color: skyblue;
   padding-top: 1%;
   padding-bottom: 1%;
   font-weight: bold;
   font-family: 'Bookman', 'URW Bookman L', serif;
}

#team_id{
   text-align: center;
   font-size: 50px;
   padding-right: 3%;
}

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

#prediction_heading{
  font-family: 'Josefin Sans', sans-serif;
  margin-top: 7.5%;
}

#result{
  font-size: 5rem;
}

#title{
  padding: 1.5% 15%;
  margin: 0 auto;
  text-align: center;
}

.btn {
    font-size: 15px;
    padding: 10px;
    /* -webkit-appearance: none; */
```

```
      background: #eee;
      border: 1px solid #888;
      margin-top: 20px;
      margin-bottom: 20px;
  }

  .buttons_div{
    margin-bottom: 30px;
    margin-right: 80px;
  }

  .heading{
    font-family:"American Typewriter", serif;
    font-weight: 700;
    font-size: 2rem;
    display: inline;
  }

  .leftside{
    text-align: center;
    margin: 0 auto;
    margin-top: 2%;
    /* padding-left: 10%; */
  }

  #frame{
    margin-right: 10%;
  }

  .predicted_answer{
    text-align: center;
    margin: 0 auto;
    padding: 3% 5%;
    padding-top: 0;
    /* padding-left: 10%; */
  }

  h1{
    text-align: center;
    color: aliceblue;
    padding: 100px 50px 65px 100px;
}

  @media (min-width: 720px) {
    .leftside{
      padding-left: 10%;
    }
  }
```

GitHub Link

https://github.com/IBM-EPBL/IBM-Project-37447-1660309468.git

Project Demo Link

https://drive.google.com/drive/folders/1eQtlydPGXAgcOSlNCyeUc06CsD8sRRJw?usp=sharing