

# A Novel Method for Handwritten Digit Recognition System

**Team ID : PNT2022TMID06143**

**Balaji B – 91761914007**

**Amarnath V– 91761914003**

**Manoj S – 91761914017**

**Saanakya SL – 91761914029**

**Sundarapandi S - 91761914041**

# TABLE OF CONTENTS

## 1. INTRODUCTION

1.1. Project Overview

1.2. Purpose

## 2. LITERATURE SURVEY

2.1. Existing problem

2.2. References

2.3. Problem Statement Definition

## 3. IDEATION & PROPOSED SOLUTION

3.1. Empathy Map Canvas

3.2. Ideation & Brainstorming

3.3. Proposed Solution

3.4. Problem Solution fit

## 4. REQUIREMENT ANALYSIS

4.1. Functional requirement

4.2. Non-Functional requirements

## 5. PROJECT DESIGN

5.1. Data Flow Diagrams

5.2. Solution & Technical Architecture

5.3. User Stories

## 6. PROJECT PLANNING & SCHEDULING

6.1. Sprint Planning & Estimation

6.2. Sprint Delivery Schedule

6.3. Reports from JIRA

## 7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1. Feature 1

7.2. Feature 2

7.3. Database Schema (if Applicable)

## 8. TESTING

8.1. Test Cases

8.2. User Acceptance Testing

## 9. RESULTS

9.1. Performance Metrics

## 10. ADVANTAGES & DISADVANTAGES

## 11. CONCLUSION

## 12. **FUTURE SCOPE**

## 13. **APPENDIX**

Source Code

GitHub & Project Demo Link

## **1. INTRODUCTION:**

Every day the world is searching new techniques in the field of computer science to upgrade the human limitations into machines to get more and more accurate and meaningful data. The way of machine learning and artificial intelligence has no negative slope it has only the slope having positive direction. This project deals with the very popular learning process called Neural Network. There are various ways by which one can achieve the goal to a desired output, but in Machine learning Neural network gives a way that machine learns the way to reach the output. To make machines more intelligent, the developers are diving into machine learning and deep learning techniques. A human learns to perform a task by practicing and repeating it again and again so that it memorizes how to perform the tasks. Then the neurons in his brain automatically trigger and they can quickly perform the task they have learned. Deep learning is also very similar to this. It uses different types of neural network architectures for different types of problems.

### **1.1 Project Overview:**

The handwritten digit recognition is the ability of computers to recognize human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different flavours. The handwritten digit recognition is the solution to this problem which uses the image of a digit and recognizes the digit present in the image.

This project has come through the concepts of statistical modelling, the computer vision and machine learning libraries which includes a lot of study about these concepts. We tried to lead these projects to the end of some updated techniques, upgradation and application of some new algorithms.

The project comes with the technique of OCR (Optical Character Recognition) which includes various research sides of computer science. The project is to take a picture of a character and process it up to recognize the image of that character like a human brain recognize the various digits. The project contains the deep idea of the Image Processing techniques and the big research area of machine learning and the building block of the machine learning called Neural Network. There are two different parts of the project 1) Training part 2) Testing part. Training part comes with the idea of to train a child by giving various sets of similar characters but not the totally same and to tell them the output of this is “this”. Like this idea one has to train the newly built neural network with so many characters. This part contains some new algorithm which is self-created and upgraded as the project need. The testing part contains the testing of a new dataset. This part always comes after the part of the training. At first one has to teach the child how to recognize the character. Then one has to take the test whether he has given right answer or not. If not, one has to train him harder by giving new dataset and new entries. Just like that one has to test the algorithm also.

The total world is working with the various problems of the machine learning. The goal of the machine learning is to factorize and to manipulate the real-life data and the real-life part of the human interaction or complex ideas or the problems in the real life.

The most curious of those is Handwritten Character Recognition because it is the building block of the human certified and the classification interaction between other humans. So, the goal was to create an appropriate algorithm that can give the output of the handwritten character by taking just a picture of that character. If one asks about Image processing then this problem can't be solved because there can be a lot of noises in that taken image which can't be controlled by human. The main thing is when human write a handwritten character or for our case digit, he has no single idea whether he has to draw it in the circulated pixels or just same as a standard image given. A machine can do that but not the human. So, by matching only the pixels one can't recognize that. The idea of machine learning lies on supervised data. Machine learning algorithm fully dependent on modelled data.

If someone models the Image directly, the model will get a lot of flatten values because that picture can be drawn with various RGB format or with various pixels which can't be modelled accurately due to noise. So, for this project one has to create a model by image processing and the machine learning. Both the techniques will be needed because these two techniques will enhance the technique of the machine learning and that can shape this project.

#### Need For The Software

The total project lies with a great computation speed and by an online server where run and compilation done quickly. All the packages were imported that were needed for the software online. We need the tools to be imported also. This project at first is in need of the software of python. The total code is written in python so it needs Python3. Python2 was not chosen because python3 has some additional upgrade over python2. The packages have been imported and the algorithm created which is done by installing the new packages from online in python3. Apart from that the total project is online compiled or ran and done by the software provided by the Google Colab free version. Apart from choosing Anaconda Navigator, Spyder or Jupiter Notebook, Google Colab or Colabotory have been chosen because this provide more speed and accurate compilation as is known. Creation of the machine learning algorithms deals with data and bigger size programs.

#### Better Functionalities

This project deals with the End users with some functionalities. The major functionality can be the Recognition of digit. User can write a handwritten digit and this project will recognize it accurately. Edge detection can be set in the process of image processing. ML algorithm can differentiate the various digits from another by recognizing it. The better functionality where the building block can be this project is Mathematical Model solver. One can take any picture of a mathematical problem and by this project one can recognize the digit inside it and then computer can compute that problem on its own. If a wrong answer comes, it can be checked through a step-by-step process by the computer and if it recognized the answer wrongly, it must be trained again. One has to train the model in various extents to recognize the various digits not only 0 to 9 but also more and more figure, like derivative integration and others.

The better functionality of this project can be license plate verification. Car license plate can be checked and one can set the record rightfully that which car is passing the gate and when by the recognition of characters.

### Feasibility Analysis

There are a lot of sides of feasibility. Let it be discussed one by one. Technical Feasibility The software which has been used in this project is fully open source and one can connect to it whenever he or she wish. The concept of python and open CV another side the research concept of image processing and machine learning is a very trending topic nowadays. Apart from that all the software running environment Google Colab is fully open source and easily can be accessed in the presence of internet. The user can also be a non-programmer and by clicking the run button he or she can set the digit in the webcam screen and can see the output.

### Economic Feasibility

This project is economically free because all the open-source software has been used that is why no money was charged or given. Only the study materials in the developer side or the designer are not available in free of cost. The software or the program is fully free of cost.

### Profitability

This project deals with two trending topics. Image Processing and Machine learning. The total machine learning concept has not used here but we have dealt with the building block of the machine learning called Neural Network. These two topics are an appropriate research topic and many scholars and teachers also are working with it every day to upgrade the techniques or the algorithms or to create some new algorithms.

The extension of the project can be used in large scale to detect the written character from images and to extract them in no time. Or in banking signature recognition or in license plate verification, Real time image chaining or filtering or object detection etc. This project is fully profitable because many sides of this project is in research nowadays and government is funding those researches.

## **1.2 Purpose:**

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

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

Handwritten Digit Recognition is the capability of a computer to fetch 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. For this, we have used support 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 the plots and maps that have been constructed using matplotlib for visualization.

A great attempt of research worker in machine learning and data mining has been contrived to achieve efficient approaches for approximation of recognition from data. In twenty first Century handwritten digit communication has its own standard and most of the times in daily life are being used as means of conversation and recording the information to be shared with individuals. One of the challenges in handwritten characters recognition wholly lies in the variation and distortion of handwritten character set because distinct community may use diverse style of handwriting, and control to draw the similar pattern of the characters of their recognized script. Identification of digit from where best discriminating features can be extracted is one of the major tasks in the area of digit recognition system. To locate such regions different kind of region sampling techniques are used in pattern recognition. The challenge in handwritten character recognition is mainly caused by the large variation of individual writing styles. Hence, robust feature extraction is very important to improve the performance of a handwritten character recognition system. Nowadays handwritten digit recognition has obtained lot of concentration in the area of pattern recognition system sowing to its application in diverse fields. In next days, character recognition system might serve as a cornerstone to initiate paperless surroundings by digitizing and processing existing paper documents. Handwritten digit dataset is vague in nature because there may not always be sharp and perfectly straight lines.

The main goal in digit recognition is feature extraction is to remove the redundancy from the data and gain a more effective embodiment of the word image through a set of numerical attributes. It deals with extracting most of the essential information from image raw data [6].

In addition, the curves are not necessarily smooth like the printed characters. Accordingly, an efficient handwritten recognition system can be developed by considering these limitations. It is quite exhausting that sometimes to identify hand written characters as it can be seen that most of the human beings can't even recognize their own written scripts. Hence, there exists constraint for a writer to write apparently for recognition of handwritten documents. Before revealing the method used in conducting this research, software engineering module is first presented. Pattern recognition along with Image processing plays compelling role in the area of handwritten character recognition.

The study describes numerous types of classification of feature extraction techniques like structural feature-based methods, statistical feature-based methods and global transformation techniques. Statistical approaches are established on planning of how data are selected. It utilizes the information of the statistical distribution of pixels in the image. The paper provided SVM based offline handwritten digit recognition system. The study provides the conversion of handwritten data into electronic data, nature of handwritten characters and the neural network approach to form machine competent of recognizing hand written characters. The study addresses a comprehensive criterion of handwritten digit recognition with various state of the art approaches, feature representations, and datasets. However, the relationship of training set size versus accuracy error and the dataset-independence of the trained models are analysed.



## 2. LITERATURE SURVEY

### 2.1 Existing problem

**A novel method for Handwritten Digit Recognition with Neural Networks - MALOTHU NAGU, N VIJAY SHANKAR, K. ANNAPURNA [1]**

Character recognition plays an important role in the modern world. It can solve more complex problems and makes humans' job easier. An example is handwritten character recognition. Pattern recognition system consists of two-stage process (Feature Extraction and Classification). Feature extraction is the measurement on a population of entities that will be classified. This assists the classification stage by looking for features that allows fairly easy to distinguish between the different classes.

There are Several Pattern Recognition Methods, they are: Bayesian decision theory, Nearest Neighbour rule, Linear Classification Discrimination

The Bayesian decision theory is a system that minimizes the classification error. This method is based on priority basis, it classifies using priority information about something that we would like to classify. We can use Baye's formula, which states the following:  $P(w_j | x) = p(x|w_j) \cdot P(w_j) / p(x)$ .

The Nearest Neighbour (NN) rule is used to classify handwritten characters. The distance measured between the two-character images is needed in order to use this rule. The goal of Linear Classification is to assign observations into the classes. This can be used to establish a classifier rule so that it can assign a new observation into a class. In another words, the rule deals with assigning a new point in a vector space to a class separated by a boundary. Linear classification provides a mathematical formula to predict a binary result. This result is a true or false (positive or negative) result. The following is an equation that can be stated as the discriminator:  $a_1 x_1 + a_2 x_2 + \dots + a_n x_n > x_0$

Shape describes a spatial region. Most shapes are a 2-D space. Shape recognition works on the similarity measure so that it can determine that two shapes correspond to each other. The recognition needs to respect the properties of imperfect perception, for example: noise, rotation, shearing, etc. One of the techniques used in shape recognition is elastic matching distance.

The difficult task is there are some handwritten digits that often run together or not fully connected. Numeral 5 is an example. But once these tasks have been carried out, the digits are available as individual items. But the digits are still in different sizes. Therefore, a normalization step has to be performed so we can have to have digits in equal sizes. After the digits are normalized, they are fed into the ANN. This is a feed-forward network with three hidden layers.

The input is a 16 x 16 array that corresponds to the size of a normalized pixel image. The first hidden layer contains 12 groups of units with 64 units per group. Each unit in the group is connected to a 5 x 5 square

in the input array and all 64 units in the group have the same 25 weight values. The second hidden layer consists of 12 groups of 16 units. This layer operates very similar to the first hidden layer, but now it seeks features in the first hidden layer. The third hidden layer consists of 30 units that are fully connected to the units in the previous layer. The output units are in turn fully connected to the third hidden layer.

### **Analytical Review of Pre-processing Techniques for Offline Handwritten Character Recognition - K. Gaurav and Bhatia P. K. [2]**

OCR has enabled scanned documents to become more than just image files, turning into fully searchable documents with text content that is recognized by computers. With the help of OCR, people no longer need to manually retype important documents when entering them into electronic databases. Instead, OCR extracts relevant information and enters it automatically. The result is accurate, efficient information processing in less time.

#### **Binarization (Thresholding)**

Document Image Binarization converts the image into bi-level form in such a way that foreground information is represented by black pixels and background by the white pixels

#### **Noise Reduction Techniques:**

The major objective of noise removal is to remove any unwanted bit-patterns, which do not have any significance in the output. Noise reduction techniques are filtering, morphological operations and noise modelling. Filters can be designed for smoothing, sharpening, thresholding, removing slightly textured background and contrast adjustment process. Various morphological operations can be designed to connect broken strokes, decomposed the connected strokes, smooth the contours, clip the unwanted points, thin the characters and extract boundaries. Smoothing can be done by filters [6]. Types of filters available are: Linear Filters (Averaging mask filter), Non-Linear Filters

It is found that major activities in a typical pre-processing system are image enhancements, noise removal, contrast adjustment, binarization, normalization, segmentation. All these techniques are necessary in a pre-processing system; no technique alone cannot be said as complete activity for pre-processing e.g., only using noise removal we cannot completely pre-process an image. Therefore, all image enhancement techniques have to be applied for accuracy in pre-processing system. Pre-processed image can be used in feature extraction phase and then in neural network phase. Even applying all these techniques, we cannot obtain the 100% accuracy in a pre-processing system.

#### **Limitations:**

OCR is unable to achieve a 100% recognition rate. Because of this, a system which permits fast and accurate recognition is a major requirement. The success of any OCR device to read accurately is the responsibility of the hardware manufacturer as well as depends on the quality of the items to be processed.

## **Devnagari numeral recognition by combining decision of multiple connectionist classifiers - Reena Bajaj, Lipika Dey, and S. Chaudhury [3]**

### **Classifier combination**

The classifiers used in this approach are based upon different representations of the input pattern. These representations, since they encode different types of property – style-dependent and style-invariant, cannot be combined into a single monolithic feature vector. Individual classifiers dealing with these representations output the class labels depending on the features used.

These class labels have been combined using a meta-pi network because it can devise a combination scheme on the basis of consistency and accuracy of the individual classifiers. Each output unit of the meta-pi net modulates output of individual classifiers. The network is called the meta-pi network owing to the multiplicative function that its output units perform. This function serves to combine the outputs of sub-networks (or “modules”), independently trained to classify numerals based on different features. Hence, there are three output nodes in the present meta-pi net. The network has been trained with the samples used for training the individual nets along with newer examples. Through the training process the meta-pi net learns to choose any one of the valid classifier outputs or a combination of these valid outputs to produce the correct global output.

The initial stage of the proposed architecture consists of connectionist modules for style-based categorisation of the input. Since style groups are characteristic of each character, for each character there exists a style categorisation module which acquires knowledge about style categories of the corresponding character through unsupervised learning. Output of this stage would indicate similarity of an unknown input with style categories of the different characters (including the correct one). An interesting feature of this stage is that the classifiers are not forced to classify distinct looking samples of one character into one monolithic class. The classifiers can self-organise themselves to categorise them into distinct style categories.

The novel feature of this work is the approach followed for identification and integration of style specific information in the recognition scheme. Use of multiple classifiers using the meta-pi network is another significant feature of this work. A complete hierarchical recognition architecture has been suggested in this work.

### **Rohan Sethi, Ila Kaushik [4]**

This paper is to demonstrate and represent the work which is related to hand-written digit recognition. The hand-written digit recognition is a very exigent task. In this recognition task, the numbers are not accurately written or scripted as they differ in shape or size; due to which the feature extraction and segmentation of hand-written numerical script is arduous. The vertical and horizontal projections methods are used for the purpose of segmentation in the proposed work. SVM is applied for recognition and classification, while convex hull algorithm is applied for feature extraction.

**L. Bottou, C. Cortes, J.S. Denker, H. Drucker [5]**

This paper compares the performance of several classifier algorithms on a standard database of handwritten digits. We consider not only raw accuracy, but also training time, recognition time, and memory requirements. When available, we report measurements of the fraction of patterns that must be rejected so that the remaining patterns have misclassification rates less than a given threshold.

**Eva Tuba, Nebojsa Bacanin [6]**

This paper describes an algorithm for handwritten digit recognition based on projections histograms. Classification is facilitated by carefully tuned 45 support vector machines (SVM) using One Against One strategy. Their proposed algorithm was tested on standard benchmark images from MNIST database and it achieved remarkable global accuracy of 99.05%, with possibilities for further improvement.

**U. Ravi Babu, Y. Venkateswarlu, Aneel Kumar Chintla [7]**

This paper presents a new approach to off-line handwritten digit recognition based on structural features which does not require thinning operation and size normalization technique. This paper uses four different types of structural features namely, number of holes, water reservoirs in four directions, maximum profile distances in four directions, and fill-hole density for the recognition of digits. The digit recognition system mainly depends on which kinds of features are used. The main objective of this paper is to provide efficient and reliable techniques for recognition of handwritten digits. A Euclidean minimum distance criterion is used to find minimum distances and k-nearest neighbour classifier is used to classify the digits. A MNIST database is used for both training and testing the system. 5000 images are used to test the proposed method a total 5000 numeral images are tested and get 96.94% recognition rate.

**Cheng-Lin Liu, K. Nakashima, H. Sako, H. Fujisawa [8]**

This paper presents the latest results of handwritten digit recognition on well-known image databases using the state of-the-art feature extraction and classification techniques. The tested databases are CENPARMI, CEDAR, and MNIST. On the test dataset of each database, 56 recognition accuracies are given by combining 7 classifiers with 8 feature vectors. All the classifiers and feature vectors give high accuracies. Among the features, the chain-code feature and gradient feature show advantages, and the profile structure feature shows efficiency as a complementary feature. In comparison of classifiers, the support vector classifier with RBF kernel gives the highest accuracy but is extremely expensive in storage and computation. Among the non-SV classifiers, the polynomial classifier performs best, followed by a learning quadratic discriminant function classifier. The results are competitive compared to previous ones and they provide a baseline for evaluation of future works.

**Chao Zhang, Zhiyao Zhou, Lan Lin [9]**

This paper proposes a new type of handwritten digit recognition system based on convolutional neural network (CNN). In order to improve the recognition performance, the network was trained with a large number

of standardized pictures to automatically learn the spatial characteristics of handwritten digits. For model training, according to the loss function, the convolutional neural network continuously updates the network parameters with the data set in MNIST, which contains 60,000 examples. For model tests, the system uses the camera to capture the pictures composed of the images generated by the test data set of MNIST and the samples written by different people, then continuously processes the captured graphics and refreshes the output every 0.5 seconds.

#### **M. Revow, C.K.I. Williams, G.E [10]**

Hinton describes a method of recognizing handwritten digits by fitting generative models that are built from deformable B-splines with Gaussian "ink generators" spaced along the length of the spline. The splines are adjusted using a novel elastic matching procedure based on the expectation maximization algorithm that maximizes the likelihood of the model generating the data. 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) the method involves a relatively small number of parameters and hence training is relatively easy and fast; and 4) unlike many other recognition schemes, it does not rely on some form of pre-normalization of input images, but can handle arbitrary scaling, translations and a limited degree of image rotation. The main disadvantage of the method is that it requires much more computation than more standard OCR techniques

#### **Shadman Sakib [11]**

This paper is to observe the variation of accuracies of CNN to classify handwritten digits using various numbers of hidden layers and epochs and to make the comparison between the accuracies. For this performance evaluation of CNN, we performed our experiment using the Modified National Institute of Standards and Technology (MNIST) dataset. Further, the network is trained using stochastic gradient descent and the back-propagation algorithm.

#### **V.N. Manjunath Aradhya, G. Hemantha Kumar [12]**

In this paper, we propose a novel system based on radon transform for handwritten digit recognition. We have used a radon function which represents an image as a collection of projections along various directions. The resultant feature vector by applying this method is the input for the classification stage. A nearest neighbour classifier is used for the subsequent recognition purpose. A test performed on the MNIST handwritten numeral database and on Kannada handwritten numerals demonstrate the effectiveness and feasibility of the proposed method.

**Savita Ahlawat, Amit Choudhary [13]**

This paper is to develop a hybrid model of a powerful Convolutional Neural Networks (CNN) and Support Vector Machine (SVM) for recognition of handwritten digits from MNIST dataset. The proposed hybrid model combines the key properties of both the classifiers. In the proposed hybrid model, CNN works as an automatic feature extractor and SVM works as a binary classifier. The MNIST dataset of handwritten digits is used for training and testing the algorithm adopted in the proposed model. The MNIST dataset consists of handwritten digits images which are diverse and highly distorted. The receptive field of CNN helps in automatically extracting the most distinguishable features from these handwritten digits. The experimental results demonstrate the effectiveness of the proposed framework by achieving a recognition accuracy of 99.28% over MNIST handwritten digits dataset.

**S. Bernard, S. Adam, L. Heutte [14]**

In the pattern recognition field, growing interest has been shown in recent years for multiple classifier systems and particularly for bagging, boosting and random sub-spaces. Those methods aim at inducing an ensemble of classifiers by producing diversity at different levels. Following this principle, Breiman introduced in 2001 another family of methods called random forest. Their work aims at studying those methods in a strictly pragmatic approach, in order to provide rules on parameter settings for practitioners. For that purpose, we have experimented with the forest-RI algorithm, considered as the random forest reference method, on the MNIST handwritten digits database. In this paper, we describe random forest principles and review some methods proposed in the literature. We present our experimental protocol and results. They finally draw some conclusions on random forest global behaviour according to their parameter tuning.

**L.S. Oliveira, R. Sabourin, F. Bortolozzi, C.Y [15]**

Suen discusses the use of genetic algorithms for feature selection for handwriting recognition. Its novelty lies in the use of multi-objective genetic algorithms where sensitivity analysis and neural networks are employed to allow the use of a representative database to evaluate fitness and the use of a validation database to identify the subsets of selected features that provide a good generalization. Comprehensive experiments on the NIST database confirm the effectiveness of the proposed strategy.

**P. Gallinari, M. Gilloux, A. Bellili [16]**

This paper presents an original hybrid MLP-SVM method for unconstrained handwritten digits recognition. Specialized support vector machines (SVMs) are introduced to significantly improve the multilayer perceptron (MLP) performances in local areas around the separation surfaces between each pair of digit classes, in the input pattern space. This hybrid architecture is based on the idea that the correct digit class almost systematically belongs to the two maximum MLP outputs and that some pairs of digit classes constitute the majority of MLP substitutions (errors). Specialized local SVMs are introduced to detect the correct class among these two classification hypotheses.

## 2.2 References

- [1] A novel method for Handwritten Digit Recognition with Neural Networks - MALOTHU NAGU, N VIJAY SHANKAR, K. ANNAPURNA
- [2] Analytical Review of Pre-processing Techniques for Offline Handwritten Character Recognition - K. Gaurav and Bhatia P. K.
- [3] Devnagari numeral recognition by combining decision of multiple connectionist classifiers - Reena Bajaj, Lipika Dey, and S. Chaudhury
- [4] Hand Written Digit Recognition using Machine Learning - Rohan Sethi, Ila Kaushik (2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT))
- [5] Learning algorithms for classification: A comparison on handwritten digit recognition - L. Bottou, C. Cortes, J.S. Denker, H. Drucker
- [6] An algorithm for handwritten digit recognition using projection histograms and SVM classifier - Eva Tuba, Nebojsa Bacanin (2015 23rd Telecommunications Forum Telfor (TELFOR))
- [7] Handwritten Digit Recognition Using Structural, Statistical Features and K-nearest Neighbour Classifier - U. Ravi Babu, Y. Venkateswarlu, Aneel Kumar Chintha
- [8] Handwritten digit recognition: benchmarking of state-of-the-art techniques - Cheng-Lin Liu, K. Nakashima, H. Sako, H. Fujisawa
- [9] Handwritten Digit Recognition Based on Convolutional Neural Network - Chao Zhang, Zhiyao Zhou, Lan Lin
- [10] Hand-printed digit recognition using deformable models - M. Revow, C.K.I. Williams, G.E
- [11] Study and Observation of the Variations of Accuracies for Handwritten Digits Recognition with Various Hidden Layers and Epochs using Convolutional Neural Network -Shadman Sakib
- [12] Two-Dimensional Matrix Principal Component Analysis Useful for Character Recognition - V.N. Manjunath Aradhya, G. Hemantha Kumar
- [13] Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN) - Savita Ahlawat, Amit Choudhary
- [14] Dynamic Random Forests -S. Bernard, S. Adam, L. Heutte
- [15] A Methodology for Feature Selection Using Multi-Objective Genetic Algorithms for Handwritten Digit String Recognition - L.S. Oliveira, R. Sabourin, F. Bortolozzi, C.Y
- [16] An MLP-SVM combination architecture for offline handwritten digit recognition -P. Gallinari, M. Gilloux, A. Bellili.

## 2.3 Problem Statement Definition

I am	Rajesh who works as a cashier in a Bank
I'm trying to	quickly enter the account details which are written by account holders
but	it takes long time
because	I don't have any application for automatic digits recognition
Which makes me feel	Frustrated because it increases my workload



Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Cashier	Enter the account details which are written by account holders	it takes long time	I don't have any application for automatic digits recognition	Frustrated because it increases my workload.
PS-2	Postman	Separate the letter according to the PIN CODE	it takes long time	I don't have any application for automatic digits recognition	Frustrated because its time consuming.

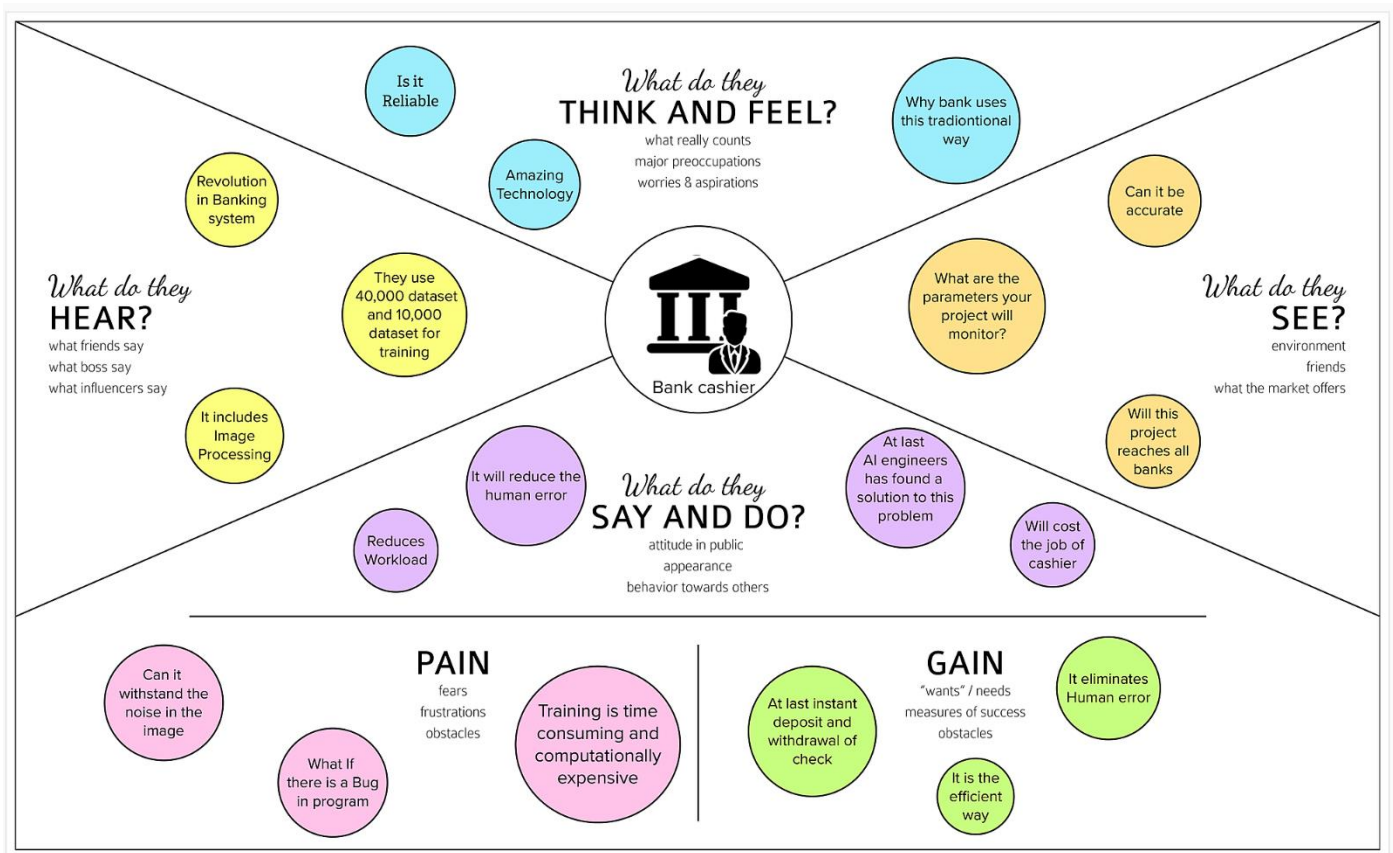


### 3. IDEATION & PROPOSED SOLUTION

#### 3.1 Empathy Map Canvas

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.

It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.



#### 3.2 Ideation & Brainstorming

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich number of creative solutions.

Team Gathering, Collaboration and Select the Problem Statement:



Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

- 10 minutes to prepare
- 1 hour to collaborate
- 2-8 people recommended

Share template feedback



Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A

Team gathering

Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B

Set the goal

Think about the problem you'll be focusing on solving in the brainstorming session.

C

Learn how to use the facilitation tools

Use the Facilitation Superpowers to run a happy and productive session.

Open article

1

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

How might we implement a new Hand Written Digit Recognition System?



Key rules of brainstorming

To run an smooth and productive session



Stay in topic.



Encourage wild ideas.



Defer judgment.



Listen to others.



Go for volume.



If possible, be visual.

# Brainstorm, Idea Listing and Grouping:

2

## Brainstorm

Write down any ideas that come to mind that address your problem statement.

🕒 10 minutes

### TIP

You can add a sticky note and fill the pencil sketch to sketch icon to start drawing!

### Balaji B

- Can be implemented using Java
- Use of keras library
- MINST dataset can be used
- Using Bango library to develop the web Application
- use of decision Naive Bayes algorithm
- Use of Random Forest to develop the web Application

### Amarnath V

- Use Digits dataset
- Use of Theano Library
- Use of Support Vector Machine library image
- Recognise each character using
- Can be implemented using python
- Creating API for further use

### Manoj S

- Generative models to produce more quality data
- Using TensorFlow Library
- Custom Artificial neural network to detect images
- Training Model from Scratch
- Using Docker for the web app
- Creating a New custom Dataset

### Saanakkya SL

- Using Convolutional neural network to perform digit recognition
- Online Recognition
- Develop a GUI Web app
- using Flask library to develop the web Application
- Can be implemented using C++
- Use of Pytorch library

### Sundarapandi S

- Use US postal Service dataset
- Develop a GUI software package
- Shape analysis of the input image and detection of character
- Online Recognition
- Random forest algorithm can be used
- Can be implemented using R programming

3

## Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub groups.

🕒 20 minutes

### Datasets

- MINST dataset can be used
- Use Digits dataset
- Creating a New custom Dataset
- Use US postal Service dataset

### Algorithms

- Use of Convolution Naive Bayes algorithm
- Using Convolutional Neural Network to perform digit recognition
- Custom Artificial neural network to detect images
- Random Forest algorithm can be used
- Use of Support Vector Machine to classify images

### Implementation Libraries

- Use of keras library
- Use of Pytorch library
- Use of Theano library
- Using TensorFlow Library

### Implementation Languages

- Can be implemented using C++
- Can be implemented using R programming
- Can be implemented using Java
- Can be implemented using Python

### Final Application

- Creating API for further use
- Develop a GUI Web app
- Develop a GUI software package
- Using Kubernetes and Docker for the web app

### Web app frameworks

- Using Django framework to develop the web Application
- Using Flask library to develop the web Application

### TIP

Add custom labels to sticky notes to make it easier to find, review, organize, and categorize into different sub groups within your mind.

### Idea Prioritization:

4

## Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

 20 minutes



### 3.3 Proposed Solution

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Rajesh who is a cashier needs a way to quickly enter the account details which are written by account holders so that account holders don't have to wait long.
2.	Idea / Solution description	<p>Using MNIST dataset and Convolutional Neural Network to Perform digits recognition.</p> <p>All of those digits can be converted into electronic words in a text document format, and this data only needs a fraction of the physical storage space of the physical copies.</p>
3.	Novelty / Uniqueness	Unlike OCR which recognize all the character, it can accurately recognize the digits.
4.	Social Impact / Customer Satisfaction	It saves time and work load in sectors which uses this technology.
5.	Business Model (Revenue Model)	<p>In banking sector, numerical detail in cheque can be easily recognized.</p> <p>Pin code details are obtainable in Postal System.</p>
6.	Scalability of the Solution	Able to distinguish numbers in noisy environment. No restriction on number of digits to be recognized.

### 3.4 Problem Solution fit

Define CS, fit into CC	<b>1. CUSTOMER SEGMENT(S)</b> Who is your customer? <div>CS</div> <p>The Customers who deal with handwritten digits like Banking sectors, schools, colleges, railways, firms, etc.</p>	<b>6. CUSTOMER CONSTRAINTS</b> What constraints prevent your customers from taking action or limit their choices of solutions? <div>CC</div> <p>They believe that the alternatives will result in errors and faults and will be inconvenient.</p>	<b>5. AVAILABLE SOLUTIONS</b> Which solutions are available to the customers when they face the problem? <div>AS</div> <p>There are no widely used software's to detect handwriting; instead, they check with other people to affirm what number it is.</p>	Explore AS, differentiate
Focus on J&P, tap into BE, understand RC	<b>2. JOBS-TO-BE-DONE / PROBLEMS</b> Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides. <div>J&amp;P</div> <p>Handwritten digits can be difficult to understand and interpret at times. It may cause errors when dealing with rough handwriting.</p>	<b>9. PROBLEM ROOT CAUSE</b> What is the real reason that this problem exists? What is the back story behind the need to do this job? <div>RC</div> <p>We face numerous challenges in handwritten number recognition. because of different people's jotting styles and the lack of Optic character recognition This investigation offers an in-depth comparison of various machine literacy and deep literacy</p>	<b>7. BEHAVIOUR</b> What does your customer do to address the problem and get the job done? <div>BE</div> <p>Finding the best software for detecting accurate digits in a more efficient manner</p>	Focus on J&P, tap into BE, understand
Identify strong TR & EM	<b>3. TRIGGERS</b> What triggers customers to act? <div>TR</div> <p>To wait for manual confirmation of digits.</p>	<b>10. YOUR SOLUTION</b> If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality. If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour. <div>SL</div> <p>A solution to this problem is the Handwritten digit recognition system, which uses a picture of a digit and recognizes the digit present in the image. Convolutional Neural Network model built with PyTorch and applied to the MNIST dataset to recognizes handwritten digits.</p>	<b>8. CHANNELS OF BEHAVIOUR</b> 8.1 ONLINE What kind of actions do customers take online? Extract online channels from #7 <div>CH</div> <p>8.2 OFFLINE          What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development</p>	Identify strong TR & EM
	<b>4. EMOTIONS: BEFORE / AFTER</b> How do customers feel when they face a problem or a job and afterwards? <div>EM</div> <p>Feels frustrated and sad when numbers are not entered.</p>		<p>Using software that is available on the internet. Obtaining assistance from those nearby in order to recognize the digits written by their customers.</p>	

## 4. REQUIREMENT ANALYSIS

### 4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story/ Sub-Task)
FR-1	User Registration	Registration through Gmail
FR-2	User Confirmation	Confirmation via Email
FR-3	Uploading the image	Uploading the handwritten digit image in the format provided
FR-4	Using a web browser	User requires a desktop or mobile browser
FR-5	Image Data	Handwritten digit recognition is the ability of a computer to recognize the human handwritten digits from different sources like images, papers, touch screens, etc, and classify them into 10 predefined classes (0-9). This has been a topic of boundless research in the field of deep learning. In the realm of deep learning, this has been the subject of countless studies
FR-6	Digit Classifier Model	To train a convolutional network to predict the digit from an image, use the MNIST database of handwritten digits. get the training and validation data first.
FR-7	MNIST dataset	The MNIST dataset is an acronym that stands for the Modified National Institute of Standards and Technology dataset. It is a collection of 60,000 tiny square grayscale photographs, each measuring 28 by 28, comprising handwritten single digits between 0 and 9.
FR-8	Cloud Computing	Cloud Computing is defined as a virtual platform that allows you to store and access your data over the internet without any limitations.

### 3.2 Non-Functional requirements

FR No.	Non-Functional Requirement	Description
NFR-1	<b>Usability</b>	Handwritten character recognition is one of the practically important issues in pattern recognition applications. The applications of digit recognition include postal mail sorting, bank check processing, form data entry, etc. One of the very significant problems in pattern recognition applications is the recognition of handwritten characters. Applications for digit recognition include filling out forms, processing bank checks, and sorting mail.
NFR-2	<b>Security</b>	All the data provided by user are secured.
NFR-3	<b>Reliability</b>	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 and also, this process is 99% accurate
NFR-4	<b>Performance</b>	The neural network uses the examples to automatically infer rules for recognizing handwritten digits. Furthermore, by increasing the number of training examples, the network can learn more about handwriting, and so improve its accuracy. There are a number of ways and algorithms to recognize handwritten digits, including Deep Learning/CNN, SVM, Gaussian Naive Bayes, KNN, Decision Trees, Random Forests, etc.
NFR-5	<b>Availability</b>	Since we use cloud, the availability of this software is all over the world only Internet facility is needed.
NFR-6	<b>Scalability</b>	This process has lot of future technology, and the usage is abundant.

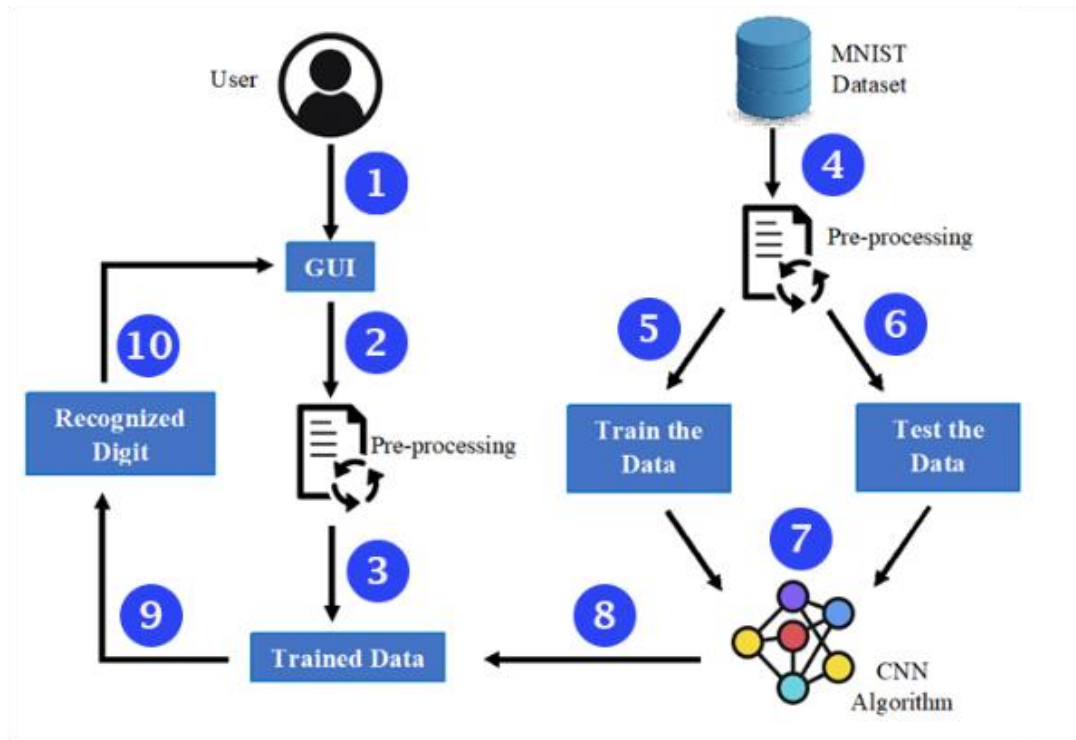


## 5. PROJECT DESIGN

### 5.1 Data Flow Diagrams

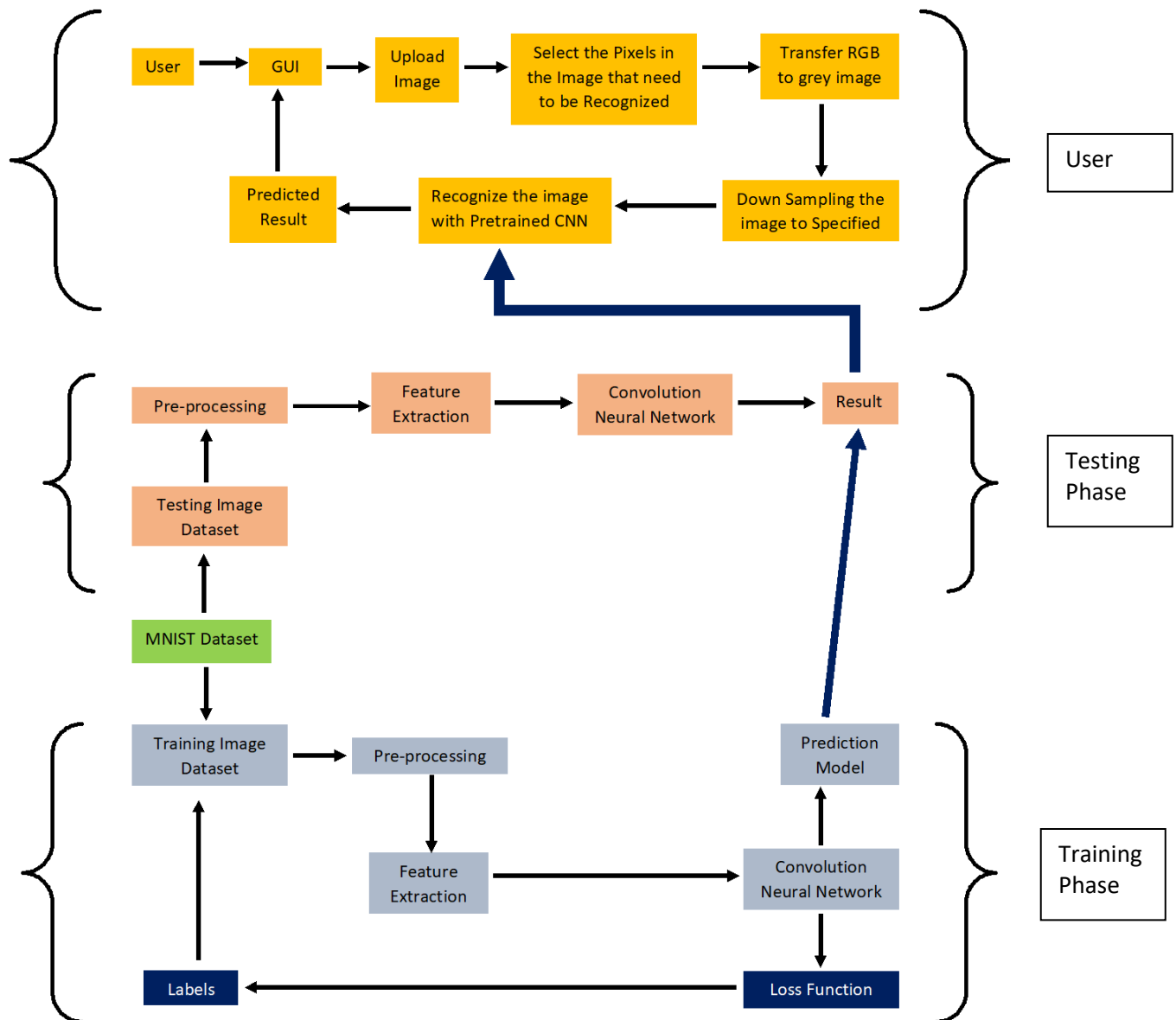
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

#### Simplified DFD:



1. Get image from user.
2. Pre-processing the given image by specifying the pixel to be recognized and convert it to black and white image.
3. Pre-processing image is compared with trained data.
4. MNIST dataset sent to pre-processing.
5. Pre-processed data classified to training data.
6. Pre-processed data classified to testing data.
7. Both Training and Testing data are sent to CNN Algorithm.
8. Trained data is obtained from the CNN Model.
9. Image from user is recognized using trained data.
10. Recognized data is sent to the user.

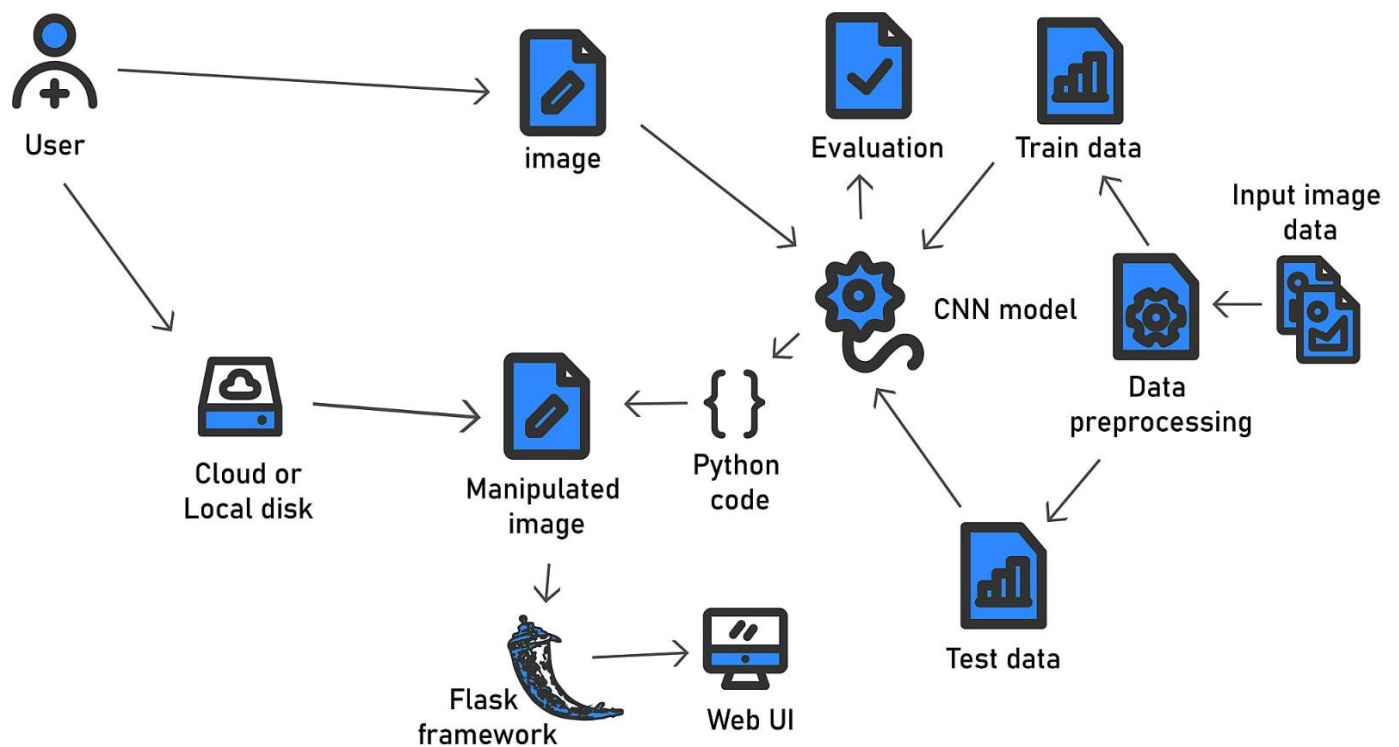
## Data Flow Diagram:



## 5.2 Solution & Technical Architecture

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behaviour, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.



### 5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Cashier in Bank Sector)	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	Medium	Sprint-2
	Uploading Image	USN-2	As a user, I can upload the images.	Uploading images (.jpg, .png, etc.,)	High	Sprint-3
	Selecting the part to be recognized	USN-3	User will select the part that need to be recognized.	By Scrolling of mouse	Low	Sprint-3
	Image Processing	USN-4	The background is eliminated from captured image and converted into binary image.	Accurate prediction is done only on the binary image.	Medium	Sprint-3
	Prediction	USN-5	The binary image is predicted as Numerical Digits (1,2,3), by using CNN model.	Accurate prediction is done using CNN model	High	Sprint-1
	Perform Action	USN-6	Performing action by training the MNIST Dataset.		High	Sprint-1
	Viewing Result	USN-7	Viewing the Recognized digital data of digits.	The resultant Digit is displayed in the web UI.	High	Sprint-4

## 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	Download the Dataset	10	High	Manoj S, Amarnath V
Sprint-1		USN-2	Image Pre-processing	10	High	Sundarapandi S, Saanakkya SL, Balaji B
Sprint-1		USN-3	Import and Configure the Image Data Generator Library and Class	10	High	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V
Sprint-1		USN-4	Apply Image Data Generator Functionality to Train-Set and Test-Set	10	High	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V
Sprint-2	Model Building	USN-5	Import the Model Building Libraries and Initializing the Model	10	High	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V
Sprint-2		USN-6	Adding CNN Layers and Dense Layers	10	High	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V
Sprint-2		USN-7	Configure the Learning Process	10	High	Mariraj K, Lokesh S
Sprint-2		USN-8	Train the Model, Save the Model and Test the Model	10	High	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V

Sprin -2		USN-9	Image processing of given image	10	High	Sundarapandi S, Saanakky SL, Balaji B, Manoj S, Amarnath V
Sprint-3	Application Building	USN-10	Create Web Application using HTML, CSS, JavaScript	10	Medium	Sundarapandi S, Saanakky SL, Balaji B, Manoj S, Amarnath V
Sprint-3		USN-11	Build Python code	10	High	Sundarapandi S, Saanakky SL, Balaji B, Manoj S, Amarnath V
Sprint-3		USN-12	Run the Application	10	High	Sundarapandi S, Saanakky SL, Balaji B, Manoj S, Amarnath V
Sprint-4	Train The Model on IBM	USN-13	Register for IBM Cloud	10	High	Sundarapandi S, Saanakky SL, Balaji B, Manoj S, Amarnath V
Sprint-4		USN-14	Train the Model and Test the Model and its Overall Performance	10	High	Sundarapandi S, Saanakky SL, Balaji B, Manoj S, Amarnath V

## 6.2 Sprint Delivery Schedule

SI. No	Milestone	Activities	Team members
1	Data collection	Download the Dataset	Sundarapandi S, Saanakkya SL
2	Data collection	Image Pre-processing	Saanakkya SL, Balaji B
3	Data collection	Import the Image Data Generator Library	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V
4	Data collection	Configure Image Data Generator Class	Manoj S, Amarnath V
5	Data collection	Apply Image Data Generator Functionality to Trainset and Test set	Saanakkya SL, Balaji B, Manoj S
6	Model Building	Import the Model Building Libraries	Sundarapandi S, Saanakkya SL, Balaji B
7	Model Building	Initializing the Model	Balaji B, Manoj S, Amarnath V
8	Model Building	Adding CNN Layers	Saanakkya SL, Balaji B, Manoj S, Amarnath V
9	Model Building	Adding Dense Layers	Sundarapandi S, Saanakkya SL, Manoj S, Amarnath V
10	Model Building	Configure the Learning Process	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V h V
11	Model Building	Train The Model	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V
12	Model Building	Save the Model	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V
13	Model Building	Test Model	Sundarapandi S,

			Saanakkya SL, Balaji B, Manoj S, Amarnath V
14	Application Building	Create HTML Pages	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V
15	Application Building	Build Python code	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V
16	Application Building	Run the Application	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V
17	Train The Model on IBM	Register for IBM Cloud	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V
18	Train The Model on IBM	Train Model on IBM	Sundarapandi S, Saanakkya SL, Balaji B, Manoj S, Amarnath V

## 7. CODING & SOLUTIONING

### 7.1 Feature 1

Along with the predicted value our model gives the probability of the other digits.

```

import os
import random
import string
from pathlib import Path
import numpy as np
from tensorflow.keras.models import load_model
from PIL import Image, ImageOps

def recognize(image):
    model=load_model(Path("./model/model.h5"))

    img = Image.open(image).convert("L")
    img_name = image.filename
    img = ImageOps.grayscale(img)

```



```

img = ImageOps.invert(img)
img = img.resize((28, 28))

img2arr = np.array(img)
img2arr = img2arr / 255.0
img2arr = img2arr.reshape(1, 28, 28, 1)

results = model.predict(img2arr)
best = np.argmax(results,axis = 1)[0]

pred = list(map(lambda x: round(x*100, 2), results[0]))

values = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
others = [(i,pred[i]) for i in range(0,10)]
best = (pred.index(max(pred)),max(pred))
#best = others.pop(best)

return best,others

```

## 7.2 Feature 2

```

# loading the MNIST dataset

mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

input_shape = (28, 28, 1)

x_train=x_train.reshape(x_train.shape[0], x_train.shape[1], x_train.shape[2], 1)
x_train=x_train / 255.0
x_test = x_test.reshape(x_test.shape[0], x_test.shape[1], x_test.shape[2], 1)
x_test=x_test/255.0

```

## 8. TESTING

### 8.1 Test Cases

S.No	Feature type	Component	Test scenario	Steps to execute	Expected Result	Actual Result	Status
1	Functional	Handwritten image of 9	Check whether the model detects 9	Find the uploaded handwritten image	9	9	Pass
2	Functional	Handwritten image of 4	Check whether the model detects 4	Find the uploaded handwritten image	4	4	Pass
3	Functional	Handwritten image of 0	Check whether the model detects 9	Find the uploaded handwritten image	0	0	Pass

### 8.2 User Acceptance Testing

Purpose of document :

The purpose of this document is to briefly explain the test coverage and open issues of the project - Emerging Methods for Early Detection of Forest Fires at the time of the release to User Acceptance Testing (UAT).

Defect analysis :

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved.

Stage	Severity 1(High)	Severity 2	Severity 3	Severity 4(Low)	Subtotal
Design	0	1	0	2	3
Coding and solutioning	2	1	2	4	9
Executing	1	1	0	1	3
Total	3	3	2	7	15
Fixed bugs	3	3	2	7	15
Not fixed	0	0	0	0	0

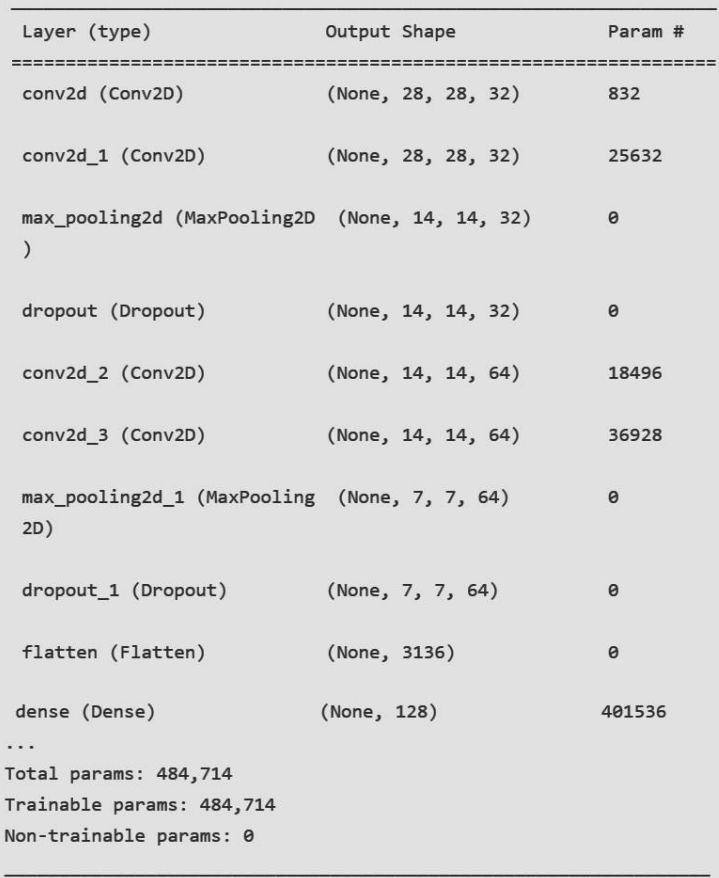
## Test Case Analysis :

This report shows the number of test cases that have passed, failed, and untested

Section	Total test cases	Not tested	Fail	Pass
Pre-processing the images	1	0	0	1
Prediction	2	0	0	2

## 9. RESULTS

### 9.1. Performance Metric

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Total params: 484,714  Trainable params: 484,714  Non-trainable params: 0	

2.	Accuracy	<p>Training Accuracy - 0.9870</p> <p>Validation Accuracy – 0.9923</p>	<pre> Epoch 1/10 844/844 [=====] - 250s 295ms/step - loss: 0.2157 - acc: 0.9340 - val_loss: 0.0394 - val_acc: 0.9893 Epoch 2/10 844/844 [=====] - 248s 294ms/step - loss: 0.0761 - acc: 0.9787 - val_loss: 0.0448 - val_acc: 0.9890 Epoch 3/10 844/844 [=====] - 245s 290ms/step - loss: 0.0589 - acc: 0.9835 - val_loss: 0.0316 - val_acc: 0.9917 Epoch 4/10 844/844 [=====] - 247s 292ms/step - loss: 0.0516 - acc: 0.9856 - val_loss: 0.0366 - val_acc: 0.9903 Epoch 5/10 844/844 [=====] - 243s 288ms/step - loss: 0.0469 - acc: 0.9869 - val_loss: 0.0261 - val_acc: 0.9942 Epoch 6/10 844/844 [=====] - 243s 288ms/step - loss: 0.0450 - acc: 0.9876 - val_loss: 0.0289 - val_acc: 0.9933 Epoch 7/10 844/844 [=====] - 242s 287ms/step - loss: 0.0468 - acc: 0.9874 - val_loss: 0.0369 - val_acc: 0.9918 Epoch 8/10 844/844 [=====] - 244s 290ms/step - loss: 0.0460 - acc: 0.9877 - val_loss: 0.0302 - val_acc: 0.9927 Epoch 9/10 844/844 [=====] - 244s 289ms/step - loss: 0.0481 - acc: 0.9872 - val_loss: 0.0305 - val_acc: 0.9932 Epoch 10/10 844/844 [=====] - 246s 291ms/step - loss: 0.0511 - acc: 0.9870 - val_loss: 0.0319 - val_acc: 0.9923 </pre>
----	----------	---	---

## 10. ADVANTAGES AND DISADVANTAGES

### Advantages:

- The system not only produces a classification of the digit but also a rich description of the instantiation parameters which can yield information such as the writing style.
- The generative models can perform recognition driven segmentation.
- The method involves a relatively small number of parameters and hence training is relatively easy and fast.
- Unlike many other recognition schemes, it does not rely on some form of pre-normalization of input images, but can handle arbitrary scaling, translations and a limited degree of image rotation.
- Detection of vehicle number – nowadays it is very difficult to identify the vehicle number when the vehicle is damaged or the number plate gets corroded, so by the use of handwritten digit recognition we easily identify the numbers present in those number plates.
- Banks for reading cheques – The bank workers will be very careful while reading the number carefully in the cheques with the help of handwritten digit recognition the software will automatically check the numbers present in the cheques and depending upon that it automatically does its work and update the process
- Post offices for arranging letter – arranging the post letters according to its numbers is mandatory nowadays, even though if we give full focus, there will be some chance to miss or rearrange the post letters, the handwritten digit recognition makes it easier.

**Disadvantages:**

- It requires much more computation than more standard OCR techniques.
- It is not done in real time as a person writes and therefore not appropriate for immediate text input.

**11. CONCLUSION:**

The main objective of this investigation is to find a representation of isolated handwritten digits that allow their effective recognition. In this paper used different machine learning algorithm for recognition of handwritten numerals. In any recognition process, the important problem is to address the feature extraction and correct classification approaches. The proposed algorithm tries to address both the factors and well in terms of accuracy and time complexity. The overall highest accuracy 90.37% is achieved in the recognition process by Multilayer Perceptron. 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.

**12. FUTURE SCOPE:**

The future development of the applications based on algorithms of deep and machine learning is practically boundless. 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, we can use these algorithms in hospitals application for detailed medical diagnosis, treatment and monitoring the patients, we can use it in surveillances system to keep tracks of the suspicious activity under the system, in fingerprint and retinal scanners, database filtering applications, Equipment checking for national forces and many more problems of both major and minor category. The advancement in this field can help us create an environment of safety, awareness and comfort by using these algorithms in day-to-day application and high-level application (i.e., corporate level or Government level). Application-based on artificial intelligence and deep learning is the future of the technological world because of their absolute accuracy and advantages over many major problems.

- Online handwriting recognition on computer tablets.
- Recognize zip codes on mail for postal mail sorting.
- Processing bank check amounts.
- Numeric entries in forms filled up by hand.

- Facial image Recognition is used in car board system depending on information of the mentality of the driver can be provided to the system to initiate his/her and the customer safety.
- Development of a facial emotion recognition system implementing the computer visions and enhancing the advanced feature extraction and classification.
- This system can be used in digital in security systems which can identify a person in any form of expression he presents himself.

### 13. APPENDIX

#### **Python:**

Python is an interpreted, high-level, general purpose programming language created by Guido Van Rossum and first released in 1991, Python's design philosophy emphasizes code Readability with its notable use of significant White space. Its language constructs and object oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming

#### **Keras :**

Keras is a powerful and easy-to-use free open source Python library for developing and evaluating deep learning models.

It wraps the efficient numerical computation libraries Theano and TensorFlow and allows you to define and train neural network models in just a few lines of code. It uses libraries such as Python, C#, C++ or standalone machine learning toolkits. Theano and TensorFlow are very powerful libraries but difficult to understand for creating neural networks. Keras is based on minimal structure that provides a clean and easy way to create deep learning models based on TensorFlow or Theano. Keras is designed to quickly define deep learning models. Well, Keras is an optimal choice for deep learning applications.

Steps for creating a keras model:

- 1)First we must define a network model.
- 2)Compile it, which transforms the simple sequence of layers into a complex group of matrix operations.
- 3)Train or fit the network.

To import:

```
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout
```

## **TensorFlow:**

TensorFlow is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow. TensorFlow tutorial is designed for both beginners and professionals. Our tutorial provides all the basic and advanced concept of machine learning and deep learning concept such as deep neural network, image processing and sentiment analysis.

TensorFlow is one of the famous deep learning frameworks, developed by Google Team. It is a free and open-source software library and designed in Python programming language, this tutorial is designed in such a way that we can easily implements deep learning project on TensorFlow in an easy and efficient way. Unlike other numerical libraries intended for use in Deep Learning like Theano, TensorFlow was designed for use both in research and development and in production systems. It can run on single CPU systems, GPUs as well as mobile devices and largescale distributed systems of hundreds of machines.

## **Numpy:**

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, Fourier transform, and matrices. Numpy which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. This tutorial explains the basics of NumPy such as its architecture and environment. It also discusses the various array functions, types of indexing, etc. It is an open source project and you can use it freely.

NumPy stands for Numerical Python. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important.

## Source Code

### Train the Model On IBM:

```
# importing Required Libraries

import tensorflow as tf
import seaborn as sns
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
from keras.utils.vis_utils import plot_model

# loading the MNIST dataset

mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

sns.countplot(y_train)

input_shape = (28, 28, 1)

x_train=x_train.reshape(x_train.shape[0], x_train.shape[1], x_train.shape[2], 1)
x_train=x_train / 255.0
x_test = x_test.reshape(x_test.shape[0], x_test.shape[1], x_test.shape[2], 1)
x_test=x_test/255.0

y_train = tf.one_hot(y_train.astype(np.int32), depth=10)
y_test = tf.one_hot(y_test.astype(np.int32), depth=10)

plt.imshow(x_train[100][:,:,0])
print(y_train[100])

X_train_plot = x_train.reshape(-1, 28, 28)

def Show_example_digits(mono = 'gray'):
    fig = plt.figure(figsize = (16, 16))
    for idx in range(15):
        plt.subplot(5, 5,idx+1)
        plt.imshow(X_train_plot[idx], cmap = mono)
        plt.title("Digit {}".format(y_train[idx]))

    plt.tight_layout()
Show_example_digits()

# determine the shape of the input images
inp_shape = x_train.shape[1:]
print(inp_shape)

batch_size = 64
num_classes = 10
```



```
epochs = 10
```

```
# defining the model
```

```
model = tf.keras.models.Sequential([tf.keras.layers.Conv2D(32, (5,5), padding='same', activation='relu',
input_shape=input_shape),
    tf.keras.layers.Conv2D(32, (5,5), padding='same', activation='relu'),
    tf.keras.layers.MaxPool2D(),
    tf.keras.layers.Dropout(0.25),
    tf.keras.layers.Conv2D(64, (3,3), padding='same', activation='relu'),
    tf.keras.layers.Conv2D(64, (3,3), padding='same', activation='relu'),
    tf.keras.layers.MaxPool2D(strides=(2,2)),
    tf.keras.layers.Dropout(0.25),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(num_classes, activation='softmax')
])
```

```
model.compile(optimizer=tf.keras.optimizers.RMSprop(epsilon=1e-08), loss='categorical_crossentropy',
metrics=['acc'])
```

```
# text Description of model
```

```
model.summary()
```

```
history = model.fit(x_train, y_train,
    batch_size=batch_size,
    epochs=epochs,
    validation_split=0.1)
```

```
# plotting the learning curves
```

```
fig, ax = plt.subplots(1,1)
ax.plot(history.history['loss'], color='b', label="Training Loss")
ax.plot(history.history['val_loss'], color='r', label="Validation Loss")
legend = ax.legend(loc='best', shadow=True)
```

```
# evaluate the model
```

```
loss, accuracy = model.evaluate(x_test, y_test, verbose=0)
print(f'Accuracy: {accuracy*100}')
```

```
y_pred = model.predict(x_test)
```

```
def draw_output(idx_nums):
```

```
    plt.figure(figsize = (20, 20))
    plt.xticks( range(10) )
    x = np.ceil(np.sqrt(len(idx_nums)))
    cnt = 1
    for ph in idx_nums:
        plt.subplot(x, x, cnt)
        curr_photo = y_test[ph]

        plt.xlim(0, 10)
        plt.title("Digit: {0}\n idx: {1} ".format(np.argmax(y_test[ph]), ph), fontsize = 10)
        plt.bar(range(10), y_pred[ph])
```

```

cnt += 1

cnt_error = []
for idx, (a, b) in enumerate(zip(y_test, y_pred)):
    if np.argmax(a) == np.argmax(b): continue
    cnt_error.append( (np.argmax(a)) )

cnt_error = np.unique(cnt_error, return_counts = True)
sns.set_style("darkgrid")
plt.figure(figsize = (15, 7))
bar_plot = sns.barplot(cnt_error[0], cnt_error[1], palette="muted")
plt.show()

cnt_ind = 1
list_idx = []
X_val_plot = x_test.reshape( x_test.shape[:-1] )
fig = plt.figure(figsize=(14, 14))

for idx, (a, b) in enumerate(zip(y_test, y_pred)):
    if np.argmax(a) == np.argmax(b): continue
    if (np.argmax(a) == 2 or np.argmax(a) == 9):
        plt.subplot(5, 5, cnt_ind)
        plt.imshow(X_val_plot[idx], cmap='gray', interpolation='none')
        plt.title('y_true={0}\ny_pred={1}\n ind = {2}'.format(np.argmax(a), np.argmax(b), idx))
        plt.tight_layout()
        list_idx.append(idx)
        cnt_ind += 1

image = x_train[0]
# lets display the image which we want to predict
plt.imshow(np.squeeze(image), cmap='gray')

image.shape[0],image.shape[1],image.shape[2]

# make a prediction
# reshaping the image for model input
image= image.reshape(1,input_shape[0],input_shape[1],input_shape[2])
# predicting the label of image
yhat = model.predict([image])
print('Predicted: {}'.format(np.argmax(yhat)))

# Predict the values from the testing dataset
Y_pred = model.predict(x_test)
# Convert predictions classes to one hot vectors
Y_pred_classes = np.argmax(Y_pred,axis = 1)
# Convert testing observations to one hot vectors
Y_true = np.argmax(y_test,axis = 1)
# compute the confusion matrix
confusion_mtx = tf.math.confusion_matrix(Y_true, Y_pred_classes)

plt.figure(figsize=(10, 8))
sns.heatmap(confusion_mtx, annot=True, fmt='g')

```

```

model.save('hand_written_digits_CNN.h5')

model = tf.keras.models.load_model('hand_written_digits_CNN.h5')

image = x_test[100]
# lets display the image which we want to predict
plt.imshow(np.squeeze(image), cmap='gray')

# make a prediction
# reshaping the image for model input
image= image.reshape(1,input_shape[0],input_shape[1],input_shape[2])
# predicting the label of image
yhat = model.predict([image])
print('Predicted: {}'.format(np.argmax(yhat)))

!tar -zcvf model.tgz hand_written_digits_CNN.h5

!pip install watson-machine-learning-client

!pip install ibm_watson_machine_learning

from ibm_watson_machine_learning import APIClient
wml_credentials = {
    "url": "https://jp-tok.ml.cloud.ibm.com",
    "apikey": "Y0hOkxEIr9-Qwjc7rRJqcbOPqPn2GdjCddwHsedqsc8N"
}
client = APIClient(wml_credentials)
client

client.spaces.get_details()

space_id = 'f5b3e32f-adf0-45f4-918c-88ac9ee62536'

client.set.default_space(space_id)

client.software_specifications.list()

software_space_uid=client.software_specifications.get_uid_by_name('tensorflow_rt22.1-py3.9')
software_space_uid

model_details = client.repository.store_model(model='model.tgz',meta_props={
    client.repository.ModelMetaNames.NAME : 'A Novel Method for Handwritten Digit Recognition System',
    client.repository.ModelMetaNames.TYPE:'tensorflow_2.7',
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_space_uid
})
model_details

model_id = client.repository.get_model_id(model_details)
model_id

client.repository.download(model_id,'model.tar.gb')

```

## app.py

```
from flask import Flask,render_template,request
from PIL import Image, ImageOps
import os
import random
import string
from pathlib import Path
import numpy as np

def random_name_generator(n):
    return "".join(random.choices(string.ascii_uppercase + string.digits, k=n))

app=Flask(__name__)

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

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

@app.route('/submit',methods=['POST'])
def submit():
    if request.method=='POST':
        image = request.files['image']
        img_name = image.filename
        file_path = os.path.join('static/data/', img_name)
        image.save(file_path)
        img = Image.open(image).convert("L")
        img = img.resize((255, 255))
        img.save(os.path.join('static/thumb/', "255X255_"+img_name))
        best = (9, 73.19)
        # others = [(0, 9.15),(1, 0.35000000000000003), (2, 0.4), (3, 0.0), (4, 109.9), (5, 4.1499999999999995),
        (6, 3.5), (7, 3.4000000000000004), (8, 3.15), (9, 365.95)]
        others = [(0, 1.83), (1, 0.07), (2, 0.08), (3, 0.0), (4, 21.98), (5, 0.83), (6, 0.7), (7, 0.68), (8, 0.63),(9, 73.19)]

        return render_template("submit.html", best=best, others=others, img_name=img_name)

if __name__=="__main__":
    app.run()
```

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

## **CNN.py**

```
import os  
import random  
import string  
from pathlib import Path  
import numpy as np  
from tensorflow.keras.models import load_model  
from PIL import Image, ImageOps  
  
def recognize(image):  
    model=load_model(Path("./model/model.h5"))  
  
    img = Image.open(image).convert("L")  
    img_name = image.filename  
    # img_name = random_name_generator(10) + '.jpg'  
  
    # if not os.path.exists(f".static/data/"):   
    #     os.mkdir(os.path.join('./static', 'data'))  
    #img.save(Path(f".static/data/{img_name}"))  
  
    img = ImageOps.grayscale(img)  
    img = ImageOps.invert(img)  
    img = img.resize((28, 28))  
  
    img2arr = np.array(img)  
    img2arr = img2arr / 255.0  
    img2arr = img2arr.reshape(1, 28, 28, 1)  
  
    results = model.predict(img2arr)  
    best = np.argmax(results,axis = 1)[0]  
  
    pred = list(map(lambda x: round(x*100, 2), results[0]))  
  
    values = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]  
    others = [(i,pred[i]) for i in range(0,10)]  
    best = (pred.index(max(pred)),max(pred))  
    #best = others.pop(best)  
  
    return best,others
```

## **GitHub & Project Demo Link**

### **GitHub:**

**<https://github.com/IBM-EPBL/IBM-Project-16783-1659622378>**

### **Project Demo Link:**

**[https://drive.google.com/file/d/19Wdrld4kUsIVZFqsK8oZ74yOf2AzGItS/view?usp=share link](https://drive.google.com/file/d/19Wdrld4kUsIVZFqsK8oZ74yOf2AzGItS/view?usp=share_link)**