

PROJECT REPORT

A NOVEL METHOD FOR HANDWRITTEN DIGIT RECOGNITION

PSG COLLEGE OF TECHNOLOGY



TEAM ID: PNT2022TMID13092

TEAM MEMBERS:

19I302-AARSHANA E WINY

19I334-P C KRUTHIKKHA

19I338-RAMYA P

19I351 -SOUNDARYA S

1. INTRODUCTION

1.1 Project Overview

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. Handwriting recognition system is the most basic and an important step towards this huge and interesting area of Computer Vision.

1.2 Purpose

To build an AI model that can input the of the handwritten digits and take the required and essential features from the taken input and analyze the AI model and predict the handwritten digit. Handwritten digit recognition becomes a vital scope and it is appealing to many researchers because of its use in a variety of machine learning and computer vision applications. It is a hard task for the machine to predict the handwritten digits because handwritten digits are not perfect and can be made with many different flavors. 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.

2. LITERATURE SURVEY

2.1 Existing problem

[1] Eva Tuba, Nebojsa Bacanin .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.

[2] Cheng-Lin Liu,K. Nakashima,H. Sako,H. Fujisawa.This paper presents the latest results of handwritten digit recognition on well-known image databases using the stateof-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.

[3] U. Ravi Babu,Y. Venkateswarlu,Aneel Kumar Chintha.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 neighbor 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.

[4] Chao Zhang,Zhiyao Zhou,Lan Lin.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.

[5] Rohan Sethi,Ila Kaushik.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.

[6] Malathy. S,C N Vanitha,Nirdhum Narayan,Rajesh Kumar,Gokul. R. In the paperwork, the speech output feature is integrated along with the text output. Convolutional Neural Network model is applied in the image classification. The dataset used to train and test is the MNIST dataset. There are various applications of handwritten digit recognition in real time. It is applied

in detection of vehicle number, reading of bank cheques, the arrangement of letters in the post office.

[7] Shadman Sakib. 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.

[8] Savita Ahlawat,Amit Choudhary. 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.

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

[10] S. Bernard, S. Adam, L. Heutte. 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 behavior according to their parameter tuning.

[11] Hong Yan,Minyue Fu,Bailing Zhang,M.A. The adaptive-subspace self-organizing map (ASSOM) proposed by Kohonen is a recent development in self-organizing map (SOM) computation.In this paper, we propose a method to realize ASSOM using a neural learning algorithm in nonlinear autoencoder networks. Our method has the advantage of numerical stability. We have applied our ASSOM model to build a modular classification system for handwritten digit recognition. Ten ASSOM modules are used to capture different features in the ten classes of digits. When a test digit is presented to all the modules, each module provides a reconstructed pattern and the system outputs a class label by comparing the ten reconstruction errors. Our experiments show promising results. For relatively small size modules, the classification accuracy reaches 99.3% on the training set and over 97% on the testing set.

[12] V.N. Manjunath Aradhya,G. Hemantha Kumar,S. 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 neighbor 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.

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

[14] P. Gallinari,M. Gilloux,A. Bellili.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. The hybrid MLP-SVM recognizer

achieves a recognition rate of 98.01%, for real mail zip code digits recognition task, a performance better than several classifiers reported in recent research.

[15] M. Revow, C.K.I. Williams, G.E. 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 scalings, translations and a limited degree of image rotation. We have demonstrated that our method of fitting models to images does not get trapped in poor local minima. The main disadvantage of the method is that it requires much more computation than more standard OCR techniques.

2.2 References

- [1] E. Tuba and N. Bacanin, "An algorithm for handwritten digit recognition using projection histograms and SVM classifier," 2015 23rd Telecommunications Forum Telfor (TELFOR), 2015, pp. 464-467, doi: 10.1109/TELFOR.2015.7377507.
- [2] Cheng-Lin Liu, K. Nakashima, H. Sako and H. Fujisawa, "Handwritten digit recognition using state-of-the-art techniques," Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition, 2002, pp. 320-325, doi: 10.1109/IWFHR.2002.1030930.
- [3] Babu, U. & Venkateswarlu, Y. & Chintha, Aneel. (2014). Handwritten Digit Recognition Using K-Nearest Neighbour Classifier. Proceedings - 2014 World Congress on Computing and Communication Technologies, WCCCT 2014. 60-65. 10.1109/WCCCT.2014.7.
- [4] C. Zhang, Z. Zhou and L. Lin, "Handwritten Digit Recognition Based on Convolutional Neural Network," 2020 Chinese Automation Congress (CAC), 2020, pp. 7384-7388, doi: 10.1109/CAC51589.2020.9326781.
- [5] R. Sethi and I. Kaushik, "Hand Written Digit Recognition using Machine Learning," 2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT), 2020, pp. 49-54, doi: 10.1109/CSNT48778.2020.9115746.

- [6] M. S, C. N. Vanitha, N. Narayan, R. Kumar and G. R, "An Enhanced Handwritten Digit Recognition Using Convolutional Neural Network," 2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA), 2021, pp. 1724-1727, doi: 10.1109/ICIRCA51532.2021.9544669.
- [7] Arif, Rezoana Bente & Siddique, Md. Abu & Khan, Mohammad Mahmudur Rahman & Oishe, Mahjabin. (2018). Study and Observation of the Variations of Accuracies for Handwritten Digits Recognition with Various Hidden Layers and Epochs using Convolutional Neural Network. 10.1109/CEEICT.2018.8628078.
- [8] ISSN 1877-0509, <https://doi.org/10.1016/j.procs.2020.03.309>.
(<https://www.sciencedirect.com/science/article/pii/S1877050920307754>)
- [9] L. Bottou et al., "Comparison of classifier methods: a case study in handwritten digit recognition," Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 3 - Conference C: Signal Processing (Cat. No.94CH3440-5), 1994, pp. 77-82 vol.2, doi: 10.1109/ICPR.1994.576879.
- [10] S. Bernard, S. Adam and L. Heutte, "Using Random Forests for Handwritten Digit Recognition," Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), 2007, pp. 1043-1047, doi: 10.1109/ICDAR.2007.4377074.
- [11] Bailing Zhang, Minyue Fu, Hong Yan and M. A. Jabri, "Handwritten digit recognition by adaptive-subspace self-organizing map (ASSOM)," in IEEE Transactions on Neural Networks, vol. 10, no. 4, pp. 939-945, July 1999, doi: 10.1109/72.774267.
- [12] V. N. M. Aradhya, G. H. Kumar and S. Nousath, "Robust Unconstrained Handwritten Digit Recognition using Radon Transform," 2007 International Conference on Signal Processing, Communications and Networking, 2007, pp. 626-629, doi: 10.1109/ICSCN.2007.350685.
- [13] L. S. Oliveira, R. Sabourin, F. Bortolozzi, and C. Y. Suen. 2002. Feature Selection Using Multi-Objective Genetic Algorithms for Handwritten Digit Recognition. In Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02) Volume 1 - Volume 1 (ICPR '02). IEEE Computer Society, USA, 10568.

[14] A. Bellili, M. Gilloux and P. Gallinari, "An hybrid MLP-SVM handwritten digit recognizer," Proceedings of Sixth International Conference on Document Analysis and Recognition, 2001, pp. 28-32, doi: 10.1109/ICDAR.2001.953749.

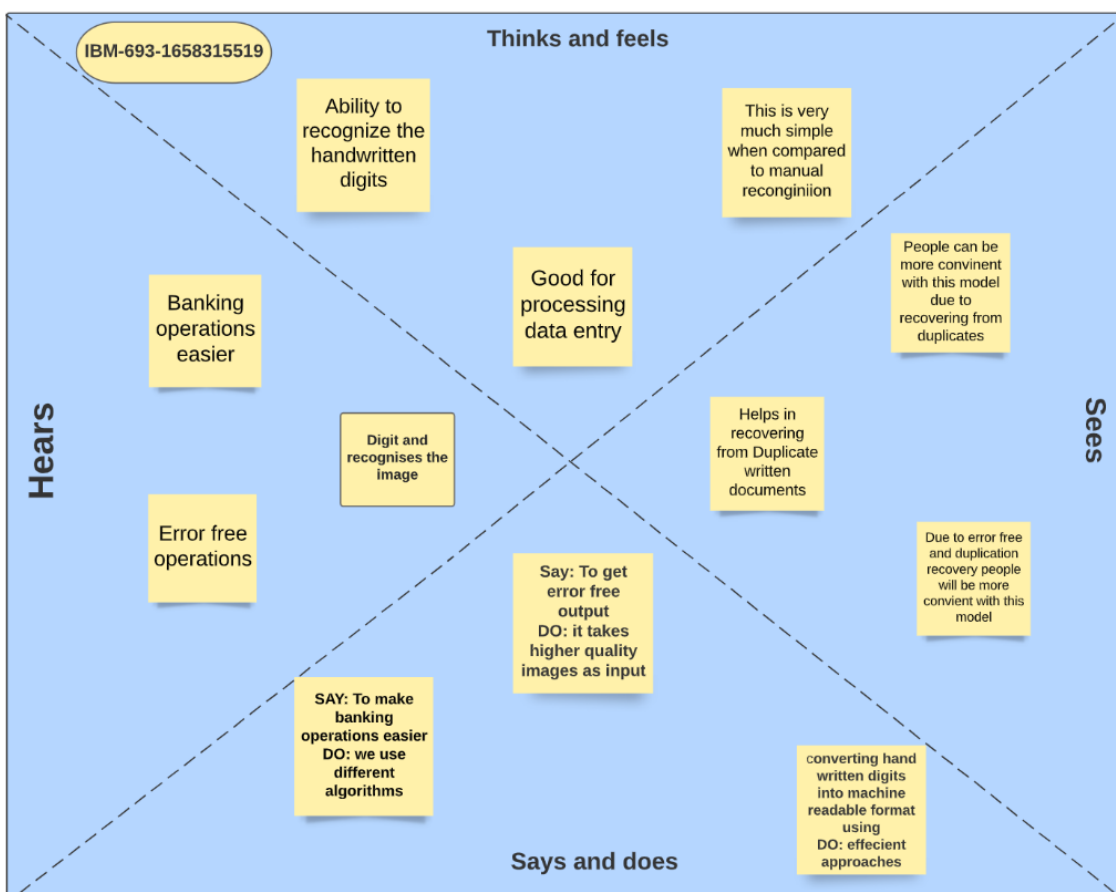
[15] M. Revow, C. K. I. Williams and G. E. Hinton, "Using generative models for handwritten digit recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, no. 6, pp. 592-606, June 1996, doi: 10.1109/34.506410.

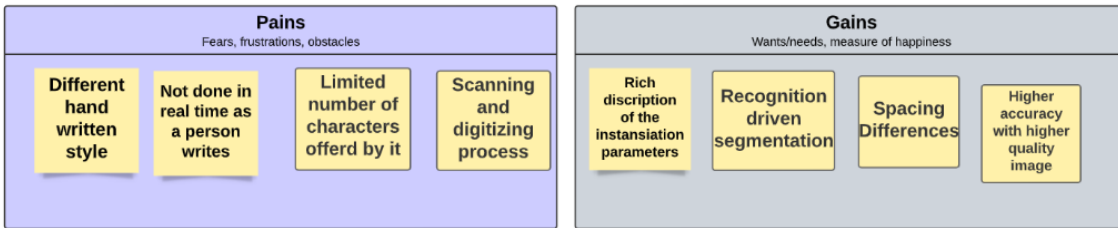
2.3 Problem Statement Definition

The problem statement is that the capability of the computer to identify and understand handwritten digits automatically where the handwritten digits images are given as input to an AI model, the required and essential features from the input are taken, analyzed and the AI model will predict the handwritten digit

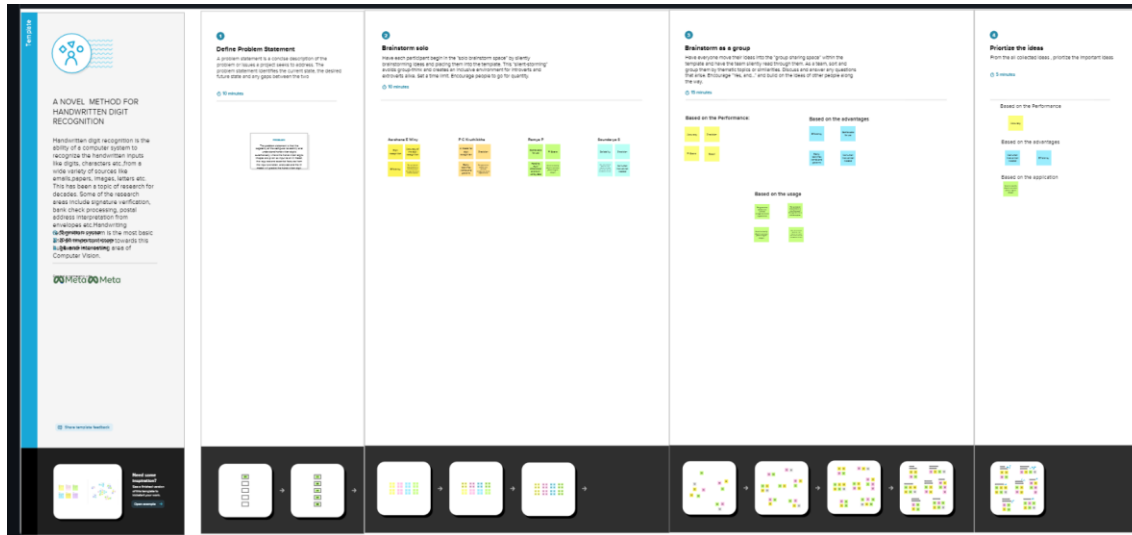
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas





3.2 Ideation & Brainstorming



3.3 Proposed Solution

To solve this problem, we are going to use Convolutional Neural Network. The Convolutional Neural Network (CNN or ConvNet) is a subtype of the Neural Networks that is mainly used for applications in image and speech recognition. Its built-in convolutional layer reduces the high dimensionality of images without losing its information. That is why CNNs are especially suited for this use case. Preprocessing involves first converting the data type from unsigned integers to floats, then dividing the pixel values by the maximum value. Deep learning models outperform machine learning models in most cases, so we are going to use the deep learning model CNN for our project. The main social impact of this work is to reduce the errors that occur in banking sectors due to incorrect recognition of handwritten digits that are written in cheques and other credit cards. Thus, automating this digit recognition will greatly improve the goodwill of the organization and customer satisfaction. Every one of us have different styles of writing and perception. Manually recognizing the handwritten digits are error prone due to various factors. So, if this digit recognition is done manually in business organizations, even if a single error occurs, it may cause severe damage to the organization. So here, we have proposed a solution to automate the digit recognition process. A deep learning model is trained with images of different styles, sizes, orientation and then the model is based to predict based on previous learning. We can extend this project into providing solutions to various other problems like

solving handwritten mathematical equation by making some changes with the training data and final code. Organizations such as banks, revenue departments, accounting sectors are facing issues in recognizing written digits such as in cheques. This can be handled by our handwritten digit recognition project as they expand into different business domains without impacting performance. Our proposed solution is thus scalable and can fit into different domains and solve different problems

3.4 Problem Solution fit

Project Title: A Novel Method for Handwritten Digit Recognition

Project Design Phase-I - Solution Fit

Team ID: PNT2022TMID13092

| | | | | |
|------------------------|--|---|---|---------------------------|
| Define CS, fit into CC | <div>1. CUSTOMER SEGMENT(S)<div>CS</div><p>Who is your customer?</p><ol style="list-style-type: none">1)Banks- processing digits in the cheque2)Automatic parking-license plate readers3)Post office-code recognition written in envelope4)Equation solver-recognize digits to solve mathematical equation5)Accounting sectors- recognition of digits in forms, cheques</div> | <div>6. CUSTOMER CONSTRAINTS<div>CC</div><p>What constraints prevent your customers from taking action or limit their choices of solutions?</p><ol style="list-style-type: none">1)Requirement of high accuracy-correct prediction needed for processing in banks2)Requirement of minimal error- errors due to variation in style, orientation, size of writing3)Minimal processing time- algorithm should take less time to execute4)Minimal storage space- algorithm should occupy less storage space</div> | <div>5. AVAILABLE SOLUTIONS<div>AS</div><p>Which solutions are available to the customers when they face the problem- or need to get the job done? What have they tried in the past? What pros & cons do these solutions have?</p><p>Past solution: Manual handwritten digit recognition</p><p>Pros:</p><ul style="list-style-type: none">Human intuition in recognizing the digits is goodFaster<p>Cons:</p><ul style="list-style-type: none">Old people and people with errors in eye sight find it difficult to predict correctly</div> | Explore AS, differentiate |
| | <div>2. JOBS-TO-BE-DONE / PROBLEMS<div>J&P</div><p>Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one, explore different sides.</p><ul style="list-style-type: none">Prepare dataset to train the modelDataset should contain handwriting from different peopleIt should have digits written in different orientation, style, size, thickness, etc.The model should make prediction with good accuracyModel should take less time to executeModel should occupy less storage space</div> | <div>9. PROBLEM ROOT CAUSE<div>RC</div><p>i)What is the real reason that this problem exists?</p><p>ii)What is the back story behind the need to do this job?</p><p>Reason:</p><ul style="list-style-type: none">Certain digits look similar, difficult to recognize themIrregularities in handwriting<p>Need:</p><ul style="list-style-type: none">Automation in banking, post offices, license plate Recognition, accounting and financial sectors to save timeIntegration with other technologies like IOT</div> | <div>7. BEHAVIOUR<div>BE</div><p>What does your customer do to address the problem and get the job done?</p><ul style="list-style-type: none">Analyse benefits of using automated handwritten digit recognizerFind the applications of the automatic handwritten digit recogniserAnalyse whether it will overcome the difficulties encountered in the past system</div> | |

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

Focus on AS, differentiate

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Functional Requirements:

Following are the functional requirements of the proposed solution.

| FR No. | Sub Requirement (Story / Sub-Task) |
|--------|---|
| FR-1 | Image Data: Handwritten digit recognition refers to a computer's capacity to identify human handwritten digits from a variety of sources, such as photographs, documents, touch screens, etc., and categorise them into ten established classifications (0-9). In the realm of deep learning, this has been the subject of countless studies. |
| FR-2 | Website: Web hosting makes the code, graphics, and other items that make up a website accessible online. A server hosts every website you've ever visited. The type of hosting determines how much space is allotted to a website on a server. Shared, dedicated, VPS, and reseller hosting are the four basic varieties. |
| FR-3 | 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-4 | Cloud: The cloud offers a range of IT services, including virtual storage, networking, servers, databases, and applications. In plain English, cloud computing is described as a virtual platform that enables unlimited storage and access to your data over the internet. |
| FR-5 | Modified National Institute of Standards and Technology dataset: The abbreviation MNIST stands for the MNIST 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. |

4.2 Non-Functional requirements

Non-functional Requirements:

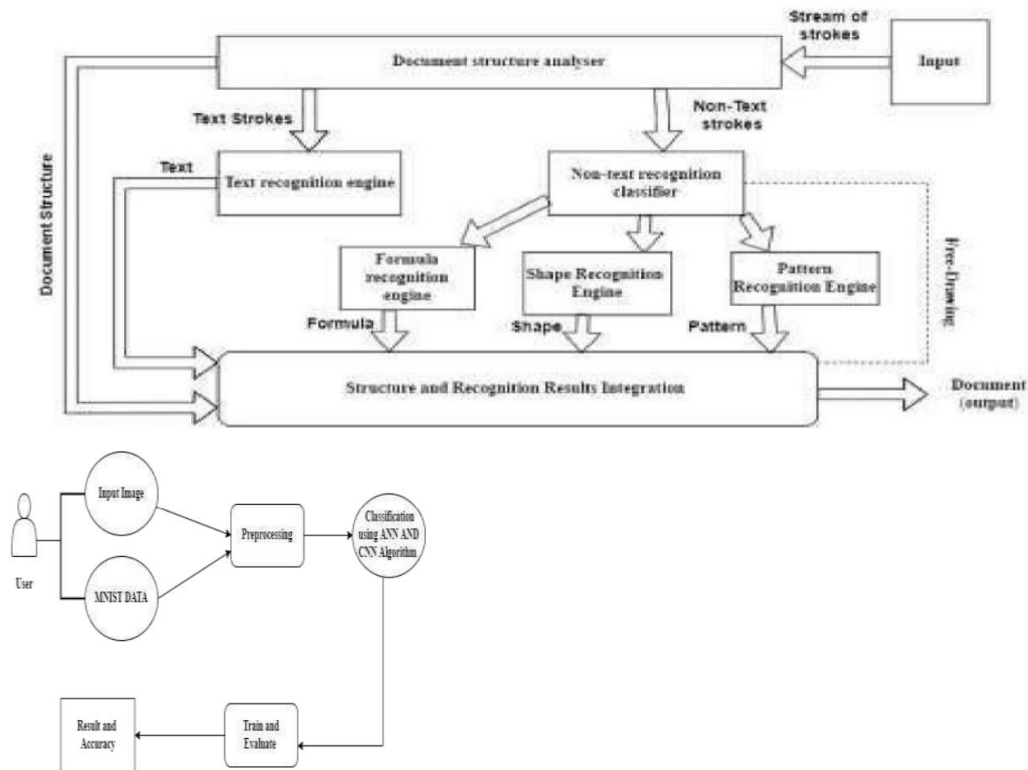
Following are the non-functional requirements of the proposed solution.

| FR No. | Non-Functional Requirement | Description |
|--------|----------------------------|---|
| NFR-1 | Usability | 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 | Security | The system generates a thorough description of the instantiation parameters, which might reveal information like the writing style, in addition to a categorization of the digit. The generative models are capable of segmentation driven by recognition. The procedure uses a relatively. |
| NFR-3 | Reliability | The samples are used by the neural network to automatically deduce rules for reading handwritten digits. Furthermore, the network may learn more about handwriting and hence enhance its accuracy by increasing the quantity of training instances. |

| | | |
|-------|--------------------|--|
| | | Numerous techniques and algorithms, such as Deep Learning/CNN, SVM, Gaussian Naive Bayes, KNN, Decision Trees, Random Forests, etc., can be used to recognise handwritten numbers. |
| NFR-4 | Performance | With typed text in high-quality photos, optical character recognition (OCR) technology offers accuracy rates of greater than 99%. However, variances in spacing, abnormalities in handwriting, and the variety of human writing styles result in less precise character identification. |
| NFR-6 | Scalability | The task of handwritten digit recognition, using a classifier, has great importance and use such as – 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 (for example - tax forms) and so on |

5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture

Dataset description:

The MNIST Handwritten Digit Recognition Dataset includes 60,000 training and 10,000 testing handwritten digit images. Each image has a height of 28 pixels and a width of 28 pixels, for a total of 784 (28*28) pixels. Each pixel is connected with a single pixel value. It indicates how bright or dark that pixel is (larger numbers indicates darker pixel). This pixel value is an integer between 0 and 255. MNIST is a dataset which is widely used for handwritten digit recognition. The dataset consists of 60,000 training images and 10,000 test images. The artificial neural networks can all most mimic the human brain and are a key ingredient in image processing field. Handwritten digit recognition using MNIST dataset is a major project made with the help of Neural Network. It basically detects the scanned images of handwritten digits. We've taken it a step further, and our handwritten digit recognition technology not only recognizes scanned images of handwritten numbers, but also allows you to write digits on the screen and have them recognized using an integrated GUI.

Procedure:

1. Install the latest TensorFlow library.
2. Prepare the dataset for the model.
3. Develop Single Layer Perceptron model for classifying the handwritten digits.

4. Plot the change in accuracy per epochs.
5. Evaluate the model on the testing data.
6. Analyze the model summary.
7. Add hidden layer to the model to make it Multi-Layer Perceptron.
8. Add Dropout to prevent overfitting and check its effect on accuracy.
9. Increasing the number of Hidden Layer neuron and check its effect on accuracy.
10. Use different optimizers and check its effect on accuracy.
11. Increase the hidden layers and check its effect on accuracy.
12. Manipulate the batch size and epochs and check its effect on accuracy.

Approach:

We will approach this project by using a three-layered Neural Network.

- The input layer: It distributes the features of our examples to the next layer for calculation of activations of the next layer.
- The hidden layer: They are made of hidden units called activations providing nonlinear ties for the network. A number of hidden layers can vary according to our requirements.
- The output layer: The nodes here are called output units. It provides us with the final prediction of the Neural Network on the basis of which final predictions can be made. A neural network is a model based on how the brain functions. It is made up of several layers with numerous activations, which mirror neurons in our brain. A neural network attempts to learn a set of parameters from a set of data, which may aid in recognising underlying links. Because neural networks can adapt to changing input, they can produce the best possible results without having to rethink the output criteria.

Methodology:

We created a Neural Network with one hidden layer and 100 activation units (excluding bias units). Data is loaded from a.mat file, then features (X) and labels (Y) are extracted. Then, to avoid overflow during computation, features are divided by 255 and rescaled into a range of [0,1]. The data is divided into 60,000 training instances and 10,000 testing examples. Feedforward is used with the training set to calculate the hypothesis, followed by backpropagation to reduce the error between the layers. To solve the issue of overfitting, the regularisation parameter lambda is adjusted to 0.1. The optimizer is run 70 times to get the best fit model.

Forward Propagation Architecture:

It is a brief description of how the CNN module will extract features and categorize the image based on them. The network's input layer, hidden layers, and output layer are depicted in the

design. The feature extraction phase of the network involves multiple layers, including convolution and resampling. Explanation of given system:

- The first layer of the architecture is the User layer. User layer will comprise of the people who interacts with the app and for the required results.
- The next three layers is the frontend architecture of the application. The application will be developed using which is the open-source platform for HTML, CSS and JavaScript. The application is deployed in the localhost which is shown on the browser. Through the app, the user will be able to upload pictures of the handwritten digits and convert it into the digitalized form.
- The one in between the database and view layer is the business layer which is the logical calculations on the basis of the request from the client side. It also has the service interface. • The backend layer consists of two datasets: Training Data and Test Data. The MNIST database has been used for that which is already divided into training set of 60,000 examples and test of 10,000 examples.
- The training algorithm used is Convolution Neural Network. This will prepare the trained model which will be used to classify the digits present in the test data. Thus, we can classify the digits present in the images as: Class 0,1,2,3,4,5,6,7,8,9.

Working:

- Neural Networks receive an input and transform it through a series of hidden layers.
- Each hidden layer is made up of a set of neurons, where each neuron is fully connected to all neurons in the previous layer.
- Neurons in a single layer function completely independently.
- The last fully connected layer is called the "output layer".

Convolution Layer:

The Convolutional layer is the core building block of a CNN. The layer's parameters consist of a set of learnable filters (or kernels), which have a small receptive field, but extend through the full depth of the input volume. During the forward pass, each filter is convolved across the width and height of the input volume, computing the dot product between the entries of the filter and the input and producing a 2- dimensional activation map of that filter. As a result, the network learns filters that activate when they see some specific type of feature at some spatial position in the input.

Feature Extraction:

All neurons in a feature share the same weights .In this way all neurons detect the same feature at different positions in the input image. Reduce the number of free parameters.

Subsampling Layer:

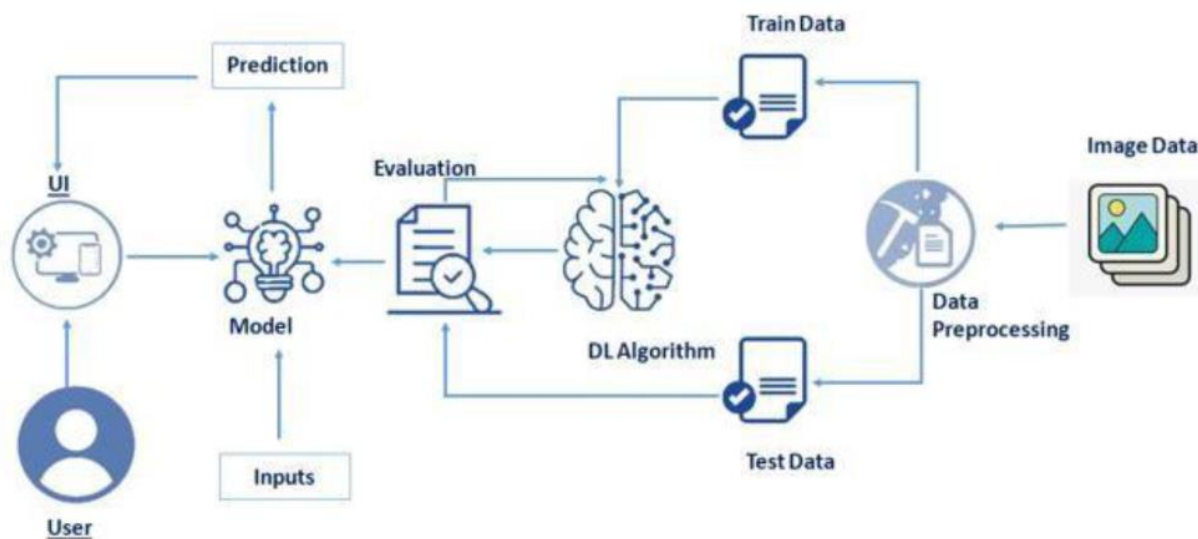
Subsampling, or down sampling, refers to reducing the overall size of a signal. The subsampling layers reduce the spatial resolution of each feature map. Reduce the effect of noises and shift or distortion invariance is achieved.

Pooling layer:

It is common to periodically insert a Pooling layer in-between successive Conv layer in a Convnet architecture. Its function is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation.

TensorFlow:

TensorFlow is an open-source machine learning library for research and production. TensorFlow offers APIs for beginners and experts to develop for desktop, mobile, web, and cloud. See the sections below to get started. By scanning the numerical digit and convert into png format using python3 command in terminal we can get text output and sound output.



5.3 User Stories

| User Type | Functional Requirement (Epic) | User Story Number | User Story / Task | Acceptance criteria | Priority | Release |
|------------------------|-------------------------------|-------------------|--|---|----------|----------|
| Customer (Mobile user) | Home | USN-1 | As a user, I can view the guide and awareness to use this application. | I can view the awareness to use this application and its limitations. | Low | Sprint-1 |

| | | | | | | |
|--|-----------|-------|---|--|--------|----------|
| | | USN-2 | As a user, I'm allowed to view the guided video to use the interface of this application. | I can gain knowledge to use this application by a practical method. | Low | Sprint-1 |
| | | USN-3 | As a user, I can read the instructions to use this application. | I can read instructions also to use it in a userfriendly method. | Low | Sprint-2 |
| | Recognize | USN-4 | As a user, In this prediction page I get to choose the image. | I can choose the image from our local system and predict the output. | High | Sprint-2 |
| | Predict | USN-6 | As a user, I'm Allowed to upload and choose the image to be uploaded | I can upload and choose the image from the system storage and also in any virtual storage. | Medium | Sprint-3 |
| | | USN-7 | 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. | High | Sprint-4 |
| | | USN-8 | As a user, I can access the MNIST data set | I can access the MNIST data set to produce the accurate result. | Medium | Sprint-3 |

| Customer (Web user) | Home | USN-9 | As a user, I can view the guide to use the web app. | I can view the awareness of this application and its limitations. | Low | Sprint-1 |
|------------------------|-------------------------------|-------------------|---|---|----------|----------|
| User Type | Functional Requirement (Epic) | User Story Number | User Story / Task | Acceptance criteria | Priority | Release |
| Customer (Mobile user) | Home | USN-1 | As a user, I can view the guide and awareness to use this application. | I can view the awareness to use this application and its limitations. | Low | Sprint-1 |
| | | | | | | |
| | | USN-2 | As a user, I'm allowed to view the guided video to use the interface of this application. | I can gain knowledge to use this application by a practical method. | Low | Sprint-1 |
| | | USN-3 | As a user, I can read the instructions to use this application. | I can read instructions also to use it in a userfriendly method. | Low | Sprint-2 |
| | Recognize | USN-10 | As a user, I can use the web application virtually anywhere. | I can use the application portably anywhere. | High | Sprint-1 |

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 | Registration | USN-1 | As a user, I can register by entering my phone number, password, and confirming my password | 8 | High | Aarshana |
| Sprint-1 | | USN-2 | As a user, I will receive confirmation otp once I have registered for the application | 2 | High | Ramya |
| Sprint-1 | | USN-3 | As a user, I can also register by entering email, password, and confirming my password | 8 | Low | Soundarya |
| Sprint-2 | | USN-4 | As a user, I can register for the application through Gmail account | 2 | Medium | Kruthikkha |
| Sprint-2 | | USN-5 | As a user, I can verify my account using the confirmation email received once I have registered for the application | 2 | Medium | Ramya |
| Sprint-1 | Login | USN-6 | As a user, I can log into the application by entering mobile number & password | 2 | High | Aarshana |
| Sprint-3 | | USN-7 | As a user, I can log into the application by entering email & password | 8 | Medium | Kruthikkha |
| Sprint-3 | | USN-8 | As a user, I can log into the application through the linked Gmail account | 2 | Low | Soundarya |
| Sprint-2 | Dashboard | USN-9 | As a user, I can drag and drop images of the digits to recognize the digits | 8 | High | Aarshana |
| Sprint-2 | Uploading | USN-10 | As a user, I can upload the images of digits that is stored in the device in order to recognize the digits | 8 | High | Ramya |
| Sprint-3 | | USN-11 | As a user, I can link my google drive and upload the images of the digits directly from the drive in order to recognize the digits | 8 | Low | Soundarya |
| Sprint-3 | | USN-12 | As a user, I can use the application effectively based on the instructions given | 2 | High | Kruthikkha |
| Sprint-4 | | USN-13 | As a user, I can upload the image once I click the upload button | 2 | High | Aarshana |
| Sprint-4 | | USN-14 | As a user, I can get confirmation message before the image gets uploaded | 2 | High | Soundarya |
| Sprint-4 | Recognition | USN-15 | As a user, I can view the recognized digit, once I click the recognize button | 8 | High | Ramya |
| Sprint-4 | | USN-16 | As a user, I can tell whether recognized digit is correct or not | 8 | High | Kruthikkha |

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 | 07 Nov 2022 |
| Sprint-4 | 20 | 6 Days | 14 Nov 2022 | 19 Nov 2022 | 20 | 14 Nov 2022 |

6.3 Reports from JIRA

Projects / Handwritten Digit Recognition

Backlog


...



Epic ▾

Label ▾

Type ▾





 Insights

▼ HDR Sprint 1 24 Oct – 29 Oct (4 issues)

20 0 0

Start sprint

...

- ☒ HDR-18 As a user, I can register by entering my phone number, password, and confirming my password 8 TO DO ▾ 
- ☒ HDR-2 As a user, I will receive confirmation otp once I have registered for the application As a us... 2 TO DO ▾  ...
- ☒ HDR-3 As a user, I can log into the application by entering mobile number & password 8 TO DO ▾ 
- ☒ HDR-4 As a user, I can also register by entering email, password, and confirming my password 2 TO DO ▾ 

+ Create issue





▼

▼ HDR Sprint 2 22 Oct – 29 Oct (4 issues)

20 0 0

Complete sprint

...

- ☒ HDR-5 As a user, I can register for the application through Gmail account 2 TO DO ▾ 
- ☒ HDR-6 As a user, I can verify my account using the confirmation email received once I have registered f... 2 TO DO ▾ 
- ☒ HDR-7 As a user, I can drag and drop images of the digits to recognize the digits 8 TO DO ▾ 
- ☒ HDR-8 As a user, I can upload the images of digits that is stored in the device in order to recognize the... 8 TO DO ▾ 





+ Create issue

▼ HDR Sprint 3 7 Nov – 12 Nov (4 issues)

20 0 0

Start sprint

...

- ☒ HDR-9 As a user, I can log into the application by entering email & password 8 TO DO ▾ 
- ☒ HDR-10 As a user, I can log into the application through the linked Gmail account 2 TO DO ▾ 
- ☒ HDR-11 As a user, I can link my google drive and upload the images of the digits directly from the drive... 8 TO DO ▾ 
- ☒ HDR-12 As a user, I can use the application effectively based on the instructions given 2 TO DO ▾ 





+ Create issue

▼ HDR Sprint 4 14 Nov – 19 Nov (4 issues)

20 0 0

Start sprint

...

- ☒ HDR-13 As a user, I can upload the image once I click the upload button 2 TO DO ▾ 
- ☒ HDR-14 As a user, I can get confirmation message before the image gets uploaded 2 TO DO ▾ 
- ☒ HDR-15 As a user, I can view the recognized digit, once I click the recognize button 8 TO DO ▾ 
- ☒ HDR-16 As a user, I can tell whether recognized digit is correct or not 8 TO DO ▾ 

+ Create issue

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

7.1 Feature 1

```
[ ] model=Sequential()  
    model.add(Conv2D(64,(3,3),input_shape=(28,28,1),activation='relu'))  
    model.add(Conv2D(32,(3,3),activation='relu'))
```

```
[ ] model.add(Flatten())  
    model.add(Dense(no_of_classes,activation='softmax'))
```

- Mandatory Conv2D parameter is the numbers of filters that convolutional layers will learn from.
- It is an integer value and also determines the number of output filters in the convolution.
- Here there are total of 32 filters and then we use Max Pooling to reduce the spatial dimensions of the output volume.
- As far as choosing the appropriate value for no. of filters, it is always recommended to use powers of 2 as the values.
- This parameter determines the dimensions of the kernel. Common dimensions include 1×1, 3×3, 5×5, and 7×7 which can be passed as (1, 1), (3, 3), (5, 5), or (7, 7) tuples.
- It is an integer or tuple/list of 2 integers, specifying the height and width of the 2D convolution window.
- This parameter must be an odd integer.

7.2 Feature 2

Evaluation is a process during development of the model to check whether the model is best fit for the given problem and corresponding data. Keras model provides a function, evaluate which does the evaluation of the model. It has three main arguments,

- Test data
- Test data label
- verbose - true or false

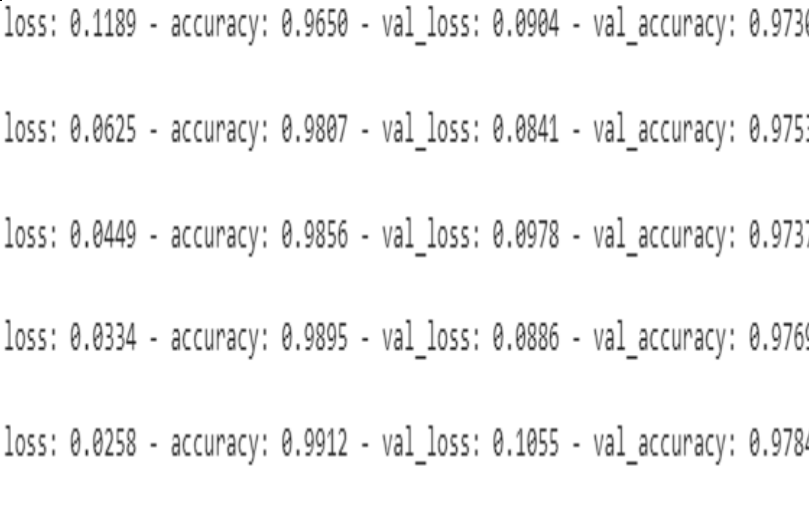
```
[ ] metrics=model.evaluate(x_test,y_test,verbose=0)  
    print("Test loss and Accuracy: ")  
    print(metrics)
```

```
Test loss and Accuracy:  
[0.1054532378911972, 0.9783999919891357]
```

8 TESTING:

8.1 Testcases:

Model Performance Testing:

| S.No. | Parameter | Values | Screenshot |
|-------|-----------|---|--|
| 1. | Metrics | Results of training the model: Epoch 5/5 loss: 0.0258 accuracy: 0.9912 val_loss: 0.1055 val_accuracy: 0.9784 |  <pre>loss: 0.1189 - accuracy: 0.9650 - val_loss: 0.0904 - val_accuracy: 0.9736 loss: 0.0625 - accuracy: 0.9807 - val_loss: 0.0841 - val_accuracy: 0.9753 loss: 0.0449 - accuracy: 0.9856 - val_loss: 0.0978 - val_accuracy: 0.9737 loss: 0.0334 - accuracy: 0.9895 - val_loss: 0.0886 - val_accuracy: 0.9769 loss: 0.0258 - accuracy: 0.9912 - val_loss: 0.1055 - val_accuracy: 0.9784</pre> |

| | | | |
|--|-------------------------------------|---|--|
| | Test loss and Accuracy of the model | Test loss and Accuracy: [0.1054532378911972, 0.9783999919891357] | <pre>metrics=model.evaluate(x_test,y_test,verbose=0) print("Test loss and Accuracy: ") print(metrics)</pre> Test loss and Accuracy: [0.1054532378911972, 0.9783999919891357] |
|--|-------------------------------------|---|--|

8.2 User Acceptance Testing

Defect Analysis

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

| 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

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

| Section | Total Cases | Not Tested | Fail | Pass |
|--------------------|-------------|------------|------|------|
| Print Engine | 6 | 0 | 0 | 6 |
| Client Application | 10 | 0 | 0 | 10 |
| Security | 1 | 0 | 0 | 1 |
| Outsource Shipping | 2 | 0 | 0 | 2 |
| Version Control | 2 | 0 | 0 | 2 |

9 RESULTS

9.1 Performance Testing

The Performance metrics that is used in this project are: test loss and accuracy

Test loss: Test Loss is the penalty for a bad prediction. That is, loss is a number indicating how bad the model's prediction was on a single example. If the model's prediction is perfect, the loss is zero; otherwise, the loss is greater.

Accuracy: Accuracy is defined as the number of classifications a model correctly predicts divided by the total number of predictions made. It's a way of assessing the performance of a model.

```
metrics=model.evaluate(x_test,y_test,verbose=0)
print("Test loss and Accuracy: ")
print(metrics)
```

```
Test loss and Accuracy:
[0.1054532378911972, 0.9783999919891357]
```

10 ADVANTAGES & 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 scalings, translations and a limited degree of image rotation.
- Accuracy of the digit recognition
- Easily identifies trends and patterns
- Comfortable for use

- Handling multidimensional and multi variety data
- Most basic and an important step towards this huge and interesting area of Computer Vision.
- No human intervention needed
- High Reliability

Disadvantages:

- It is not done in real time as a person writes and therefore not appropriate for immediate text input.
- Characters look very similar, making it hard for a computer to recognise accurately.
- Joined-up handwriting is another challenge for computers
- No creativity. A big disadvantage is that it cannot learn to think outside the box.

11 CONCLUSION

The task of handwriting recognition is the transcription of handwritten data into a digital format. The goal is to process handwritten data electronically with the same or nearly the same accuracy as humans. Handwritten digit recognition has immense applications

in the field of medical, banking, student management, and taxation process etc. Many classifiers like KNN, SVM, CNN are used to identify the digit from the handwritten image. as per the review, CNN is providing better performance than others.

12 FUTURE SCOPE

The task of handwritten digit recognition, using a classifier, has great importance and use such as – 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 (for example - tax forms) and so on.

13 APPENDIX

Source Code

✓ Importing the required libraries

```
import numpy as np
import tensorflow
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten, Conv2D
from tensorflow.keras.models import load_model
from keras.optimizers import Adam
from keras.utils import np_utils

import matplotlib.pyplot as plt
import seaborn as sns
from PIL import Image
```

✓ Loading the data

```
(x_train,y_train),(x_test,y_test)=mnist.load_data()

print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

✓ Analyzing the data

```
print(x_train[0])  
print(y_train[0])  
plt.imshow(x_train[0])
```

✓ Reshaping the data

```
x_train=x_train.reshape(60000,28,28,1).astype('float32')  
x_test=x_test.reshape(10000,28,28,1).astype('float32')
```

✓ Applying onehot encoding

```
no_of_classes=10  
y_train=np_utils.to_categorical(y_train,no_of_classes)  
y_test=np_utils.to_categorical(y_test,no_of_classes)  
print(y_train[0])
```

✓ MODEL BUILDING

1)Add CNN layers

```
model=Sequential()  
model.add(Conv2D(64,(3,3),input_shape=(28,28,1),activation='relu'))  
model.add(Conv2D(32,(3,3),activation='relu'))
```

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

2)Compiling the model

```
model.compile(loss='categorical_crossentropy',optimizer='Adam',metrics='accuracy')
```

3)Train the model

```
model.fit(x_train,y_train,validation_data=(x_test,y_test),epochs=5,batch_size=32)
```

4)Observing the metrics

```
metrics=model.evaluate(x_test,y_test,verbose=0)
print("Test loss and Accuracy: ")
print(metrics)
```

5)Test the model

```
prediction=model.predict(x_test[:4])
print(prediction)
```

```
print(np.argmax(prediction,axis=1))
print(y_test[:4])
```

6)Save the model

```
from google.colab import drive
drive.mount('/content/drive')
cd /content/drive/MyDrive/lbm
model.save('mnistCNN.h5')
ls
```

7)Test with the saved model

```
model=load_model('mnistCNN.h5')
for index in range(4):
    img=Image.open('/content/sample_data/data/'+str(index)+".PNG").convert("L")
    img=img.resize((28,28))
    im2arr=np.array(img)
    im2arr=im2arr.reshape(1,28,28,1)
    y_pred=model.predict(im2arr)
    print(y_pred)
```

app.py:

```
import numpy as np
import os
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from flask import Flask,render_template,request
from PIL import Image

app=Flask(__name__)
```

```

model=load_model("digit.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']
        basepath=os.path.dirname(__file__)
        filepath=os.path.join(basepath,'uploads',f.filename)
        f.save(filepath)
        img = Image.open(filepath).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)

        index=['Zero','One','Two','Three','Four','Five','Six','Seven','Eight','Ni
ne']

        text="The Classified Digit is : " +str(index[num[0]])

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

```

index.html:

```

<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Handwritten Digit Classification</title>
    <link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css"
rel="stylesheet">

```

```

<script
src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
<script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
<script
src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
<link href="{ url_for('static', filename='css/main.css') }"
rel="stylesheet">
<style>

.bg-dark {
    background-color: #112099!important;
}
#result {
    color: #ffffff;
}
body
{
    background-image:
url("data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAASEAAACvCAMAAACFDpg1AAAAkFBMVE
V04uz///+t6u5I40r///38/Px/f3/+f7W1tb9/PrT09P6+vrz8/Nvb291dXXu7u+FhYX15uWQkJChoaG
tra3b3NybnJvCwsKULJTIyciGhIfGxMeLiYyzs7Pg40BsbGxurr0xsrIOERArLCxiY2NGR0ZaW1lTUVR5
d3opJypycHO6ubuRj5N2dHeBgoE+Pj7QneZTAAACx0lEQVR4nO3Wa30bOBSAYZ+NJTACcZMM0AFvWrtxQ
zf7///dCkNu05182ClTz+d9wBpdDTqWwJsNAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGP+xm199A7+7m80NPrD58w98aKM1EnwgREi2v/o
mfmsHQt7cIRfoSQTnd7qVyu9QFMrdNzbuly+55wNxt/sjqzGvLcuPLV65mWkNrIiI1MsYhE440UkYi
FU1npFQqKjQqKd3UL+RCMjXJdeNPbWFAnoe6NDK5kjRko2lQGvKR5GnokMp0gTXNa0jWS6b7r50wk80sK
+Pd66Wbbsn4KnR9N89QCmdaZXJf2EGqRo5V5TPtxH8SaXu5/7z1Vt83+uSnK0wX+zHJ9zNYe5cp6W3nsi
wulHPnoj390/9cGqdjbMvyd66XBdN24xbiW01aJvs9V5bI6bcqjKx4sYi6bIyTcTqJM1MZAu7ltLYbdu
Zk07Lp5WfE10Eo1XJoZd67Fod1fEh8i4/64fbg1xuo+bW38nlWy/+611YYRcftbFD5ZBfyjsv/mHabY9S
7y/xsT96Gd1hkMdveRWdxsdYLjIVL0b+Wvn+o7WfQyKmqVStNpUiU6TOvzoxrdiXN2M8Z3YqldJoXMR7
U3bDbpQWRXJMtZK1sbHqLzX1yH05bF60ldaXCMh6cusKLtW1n80rczZJkTIjdbGdTGcEpFVQ5iu7fzRt2
GX2aKph7aJ510WV6Yy+/g6Mi/OdWKP5Wir+G+ry8KGyLUSiqZ9KKait1m6coDWj9C/T+C/Tkt9PzS8Hle
1doTU9Zje3eHtHU4zV8m1NDeE0qVeCtc2s4x9joha/heo19K7b1vTT1hD86SW2ao3U3yd+dJxDtTbdfIc
oTmA8hLZuah+QoDkH2RYP2W+Re91AAAAAE1FTkSuQmCC");
    background-size: cover;
}

</style>
</head>

<body>

<nav class="navbar navbar-dark bg-dark">
    <div class="container">

```

```

        <a class="navbar-brand" href="#">Handwritten Digit Classification
using CNN</a>
    </div>
</nav>
<div class="container">
    <div id="content" style="margin-top:2em">
        <div class="container">
            <div class="row">
                <div class="col-sm-6 bd">
                    <h3>Handwritten Digit Classification: </h3>
                    <br>
                    <p>Handwritten digit recognition is the process to provide the
ability to machines to recognize human handwritten digits. It is not an easy task
for the machine because handwritten digits are not perfect, vary from person-to-
person, and can be made with many different flavor.</p>
                    
                </div>
                <div class="col-sm-6">
                    <div>
                        <h4>Upload Image Here To Identify the Digit</h4>
                        <form action = "http://localhost:5000/" id="upload-file"
method="post" enctype="multipart/form-data">
                            <label for="imageUpload" class="upload-label">
                                Choose...
                            </label>
                            <input type="file" name="image" id="imageUpload" accept=".png,
.jpg, .jpeg">
                        </form>

                        <div class="image-section" style="display:none;">
                            <div class="img-preview">
                                <div id="imagePreview">
                                    </div>
                                </div>
                                <div>
                                    <button type="button" class="btn btn-info btn-lg " id="btn-
predict">Predict!</button>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>

```

```

        <div class="loader" style="display:none;"></div>

        <h3>
            <span id="result"> </span>
        </h3>

    </div>
</div>

    </div>
</div>
</div>
</div>
</body>

<footer>
    <script src="{{ url_for('static', filename='js/main.js') }}"
type="text/javascript"></script>
</footer>

</html>

```

Main.css:

```

.img-preview {
    width: 256px;
    height: 256px;
    position: relative;
    border: 5px solid #F8F8F8;
    box-shadow: 0px 2px 4px 0px rgba(0, 0, 0, 0.1);
    margin-top: 1em;
    margin-bottom: 1em;
}

.img-preview>div {
    width: 100%;
    height: 100%;
    background-size: 256px 256px;
    background-repeat: no-repeat;
    background-position: center;
}

input[type="file"] {

```

```

    display: none;
}

.upload-label{
    display: inline-block;
    padding: 12px 30px;
    background: lavender;
    color: #fff;
    font-size: 1em;
    transition: all .4s;
    cursor: pointer;
}

.upload-label:hover{
    background: #902ac0;
    color: #c19dbe;
}

.loader {
    border: 8px solid #f3f3f3; /* Light grey */
    border-top: 8px solid #df0be3; /* Blue */
    border-radius: 50%;
    width: 50px;
    height: 50px;
    animation: spin 1s linear infinite;
}

@keyframes spin {
    0% { transform: rotate(0deg); }
    100% { transform: rotate(360deg); }
}

```

GitHub & Project Demo Link:

Demo:

https://drive.google.com/file/d/1r3po09Xknr9T_8ZPiwycGppmwkyQRTwx/view?usp=sharing

Github:

<https://github.com/IBM-EPBL/IBM-Project-693-1658315519>

