HANDWRITTEN DIGIT RECOGNITION USING DEEP LEARNING

Major project report submitted
in partial fulfillment of the requirementfor award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

PAVAN KUMAR PELLURU	(20UECS0735)	**(17153)**
G NARAYANA SWAMY NAIDU	(20UECS0293)	**(17168)**
B HARSHA VARDHAN REDDY	(20UECS0095)	**(18328)**

Under the guidance of
Dr. M Sankar,M.Tech,Ph.D.,
PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

HANDWRITTEN DIGIT RECOGNITION USING DEEP LEARNING

Major project report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

PAVAN KUMAR PELLURU	(20UECS0735)	**(17153)**
G NARAYANA SWAMY NAIDU	(20UECS0293)	**(17168)**
B HARSHA VARDHAN REDDY	(20UECS0095)	**(18328)**

Under the guidance of
Dr. M Sankar,M.Tech,Ph.D.,
PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled "HANDWRITTEN DIGIT RECOGNITION USING DEEP LEARNING" by "PAVAN KUMAR PELLURU (20UECS0735), G NARAYANA SWAMY NAIDU (20UECS0293), B HARSHA VARDHAN REDDY (20UECS0095)" has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

Signature of Professor In-charge
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)
PAVAN KUMAR PELLURU
Date: / /

(Signature)
G NARAYANA SWAMY NAIDU
Date: / /

(Signature)
B HARSHA VARDHAN REDDY
Date: / /

APPROVAL SHEET

This project report entitled “HANDWRITTEN DIGIT RECOGNITION USING DEEP LEARNING” by “PAVAN KUMAR PELLURU(20UECS0735), G NARAYANA SWAMY NAIDU (20UECS0293), B HARSHA VARDHAN REDDY (20UECS0095)” is approved for the degree of B.Tech in Computer Science & Engineering.

Examiners **Supervisor**

Dr. M Sankar,M.Tech,Ph.D.,

Date: / /

Place:

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science & Engineering,Dr.M.S. MURALI DHAR, M.E., Ph.D.,** for providing immense support in all our endeavors.

We also take this opportunity to express a deep sense of gratitude to our Internal Supervisor **Dr. M SANKAR,M.Tech,Ph.D.,** for his cordial support, valuable information and guidance, he helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. ASHOK KUMAR, M.Tech., Ms. C. SHYAMALA KUMARI, M.E.,** for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

PAVAN KUMAR PELLURU (20UECS0735)
G NARAYANA SWAMY NAIDU (20UECS0293)
B HARSHA VARDHAN REDDY (20UECS0095)

ABSTRACT

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans. In deep learning, Convolutional Neural Network (CNN) is at the center of spectacular advances that mixes Artificial Neural Network (ANN) and up to date deep learning strategies. It has been used broadly in pattern recognition, sentence classification, speech recognition, face recognition, text categorization, document analysis, scene, and digit recognition. Humans can see and visually sense the world around them by using their eyes and brains. Computer vision works on enabling computers to see and process images in the same way that human vision does by using several machine learning and deep learning algorithms. Digit Recognition (DR) is the process of converting images of digit into digital format.The existing systems uses various algorithms for implementing digit recognition systems which consist of Proximal Support Vector Machine(PSVM), Multilayer Perceptron(MLP), Support Vector Machine (SVM) but by using these machine learning algorithms one can only get accuracy of about 80% but for many other applications such as banking industry applications require better accuraccy. This project lies within the ability to develop an efficient algorithm that can recognize the digits which are scanned and sent as input by the user with better accuracy. This experiment is performed using the Modified National Institute of Standards and Technology (MNIST) dataset and Convolutional Neural Network algorithm. The goal of the project is to create a model that will be able to identify and determine the digit from its image with better accuracy by using Convolutional Neural Network can get 92% accuracy.

Keywords: Artificial Neural Network, Convolutional Neural Network, Deep Learning, Digit Recognition, MNIST Dataset, Support Vector Machine.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

ANN	Artificial Neural Network
API	Application Programming Interface
CNN	Convolutional Neural Network
FCL	Fully Connected Layer
GUI	Graphical User Interface
HMM	Hidden Markov Models
MNIST	Modified National Institute of Standards and Technology
OCR	Optical Character Recognition
PIL	Python Image Library
PSO	Particle Swarm Optimization
SVM	Support Vector Machine
TF	Tensor Flow
UML	Unified Modeling Language

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

Developers are using different machine learning and deep learning techniques to make machines more intelligent. In deep learning, Convolutional Neural Network is being used in many fields like object detection, face recognition, spam detection, image classification. Digit recognition has not only professional and commercial applications, but also has practical application in our daily life and can be of great help to the visually impaired. It also helps us to solve complex problems easily thus making our lives easier. Digit recognition is the ability of a machine to recognize human handwritten digits. It is a hard task for a machine because digits are not perfect. So, the solution to this problem is to uses the image of a digit and recognizes the digit present in the image. This project uses applied Convolutional Neural Network algorithm for training on the MNIST dataset a library written in python. Digit Recognition system is the working of a machine to train itself or recognizing the digits from different sources like emails, bank cheque, papers, images, etc. and in different real world scenarios for online digit recognition on computer tablets or system, recognize number plates of, numeric entries in forms filled up by hand and so on.

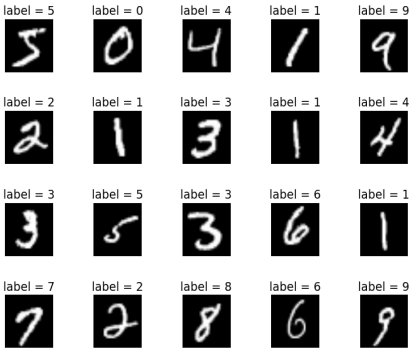

Figure 1.1: **Test Data**

The following Figure 1.1 contains test samples of digits which are present in MNIST dataset.

1.2 Aim of the Project

Handwritten digit recognition using deep learning aims to equip computers with the ability to accurately interpret and recognize handwritten numbers. This task encounters significant challenges due to the inherent variability in human handwriting, characterized by differences in style, size, and orientation.To address this challenge, deep learning techniques, particularly Convolutional Neural Networks (CNN), are employed to learn hierarchical representations of input images. These networks are trained on extensive datasets of handwritten digits, wherein their parameters are adjusted iteratively to minimize classification errors. Through continuous learning from examples, the model gradually enhances its ability to generalize and accurately recognize new handwritten digits. The successfull implementation of this approach facilitates automated digit recognition across various applications, including postal services, bank transactions, and digitized document processing.

1.3 Project Domain

1.3.1 Deep Learning

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. For instance object recognition, image and sound classification, object detection, image segmentation, etc. The digit recognition is the ability of computers to recognize human digits. It is a hard task for the machine because digits are not perfect and can be made with many different flavors. The digit recognition is the solution to this problem which uses the image of a digit and recognizes the digit present in the image.

1.3.2 Convolutional Neural Networks

Convolutional Neural Networks are deep artificial neural networks. Convolutional Neural Networks is used to classify images (e.g., name what they see), cluster them by similarity (photo search) and perform object recognition within scenes. It is used to identify faces, individuals, street signs, tumors, platypuses and many other aspects of visual data. The convolutional layer is the core building block of a Convolutional Neural Networks.

1.4 Scope of the Project

The scope of using Convolutional Neural Networks for digit recognition is quite broad. It includes both the development and training of the Convolutional Neural Networks model, as well as the evaluation and optimization of its performance. The task of digit recognition using a classifier has great importance and use such as online digit 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 and so on. There are different challenges faced while attempting to solve this problem. The digits are not always of the same size, thickness or orientation and position relative to the margins. The goal is to implement a pattern classification method to recognize the digits provided by the user. The general problem faced in this digit classification was the similarity between the digits like 1 and 7, 5 and 6, 3 and 8,etc.

Also people write the same digits in many different ways. Finally the uniqueness and variety in the handwriting of different individuals also influences the formation and appearance of the digits. Digit recognition regarding the applying of machine learning algorithms supported image preprocessing and have extraction. To boot, the requirements are not alone to reinforce this recognition performance, but together to hunt the simplest trustiness inside the applications of written digits. The aim of this project is to implement a classification algorithm to acknowledge written digits (0-9). It has been shown in pattern recognition that no single classifier performs the foremost effective for all pattern classification problems consistently.

Chapter 2

LITERATURE REVIEW

[1] A. B. Siddique, et al., [2019], have proposed in their paper a system for offline cursive digit recognition is described which is based on Hidden Markov Models (HMM) using discrete and hybrid modelling techniques. Digit recognition experiments using a discrete and two different hybrid approaches, which consist of a discrete and semicontinuous structures, are compared. It is found that the recognition rate performance can be improved of a hybrid modelling technique for HMMs, which depends on a neural vector quantizer, compared to discrete and hybrid HMMs, based on tired mixture structure, which may be caused by a relative small data set.

[2] C.C.Park,et al., [2019], have presented in their paper the various preprocessing techniques involved in the digit recognition with different kind of images ranges from a simple digit form based documents and documents containing colored and complex background and varied intensities.In this, different pre-processing techniques like skew detection and correction, image enhancement techniques of contrast stretching, binarization, noise removal techniques, normalization and segmentation, morphological processing techniques are discussed.

[3] Dala, et al., [2020], have presented in their study three different kinds of features, namely, the density features, moment features and descriptive component features for classification of Devanagari Numerals. They proposed multi classifier connectionist architecture for increasing the recognition reliability and they obtained 89.6%accuracy for Devanagari numerals.

[4] J. Pradeep, et al., [2020], have discussed Diagonal feature extraction proposed for offline character recognition. It is based on ANN model. Two approaches using 54 features and 69 features are chosen to build this Neural Network recognition system. To compare the recognition efficiency of the proposed

diagonal method of feature extraction, the neural network recognition system is trained using the horizontal and vertical feature extraction methods.

[5] Mayank jain , et al., [2021], have presented in their study the recognition of Hindi and English numerals by representing them in the form of exponential membership functions which serve as a fuzzy model. The recognition is carried out by modifying the exponential membership functions fitted to the fuzzy sets. These fuzzy sets are derived from features consisting of normalized distances obtained using the Box approach. The membership function is modified by two structural parameters that are estimated by optimizing the entropy subject to the attainment of membership function to unity. The overall recognition rate is found to be 85% for Hindi numerals and 92% percent for English numerals.

[6] M. M. A. Ghosh , et al., [2021], have presented in their study the recognition of Hindi and English numerals by representing them in the form of exponential membership functions which serve as a fuzzy model. The recognition is carried out by modifying the exponential membership functions fitted to the fuzzy sets. These fuzzy sets are derived from features consisting of normalized distances obtained using the Box approach. The membership function is modified by two structural parameters that are estimated by optimizing the entropy subject to the attainment of membership function to unity. The overall recognition rate is found to be 87% for Hindi numerals and 90% percent for English numerals.

[7] R. B. Arif, et al., [2021], have proposed SVM based offline digit recognition. Authors claim that SVM outperforms the Multilayer perceptron classifier. Experiment is carried out on MNIST standard dataset. Advantage of Multilayer perceptron is that it is able to segment non-linearly separable classes. However, Multilayer perceptron can easily fall into a region of local minimum, where the training will stop assuming it has achieved an optimal point in the error surface.

[8] R. Alhajj , et al., [2021], have discussed fuzzy membership function based approach for character recognition. Character images are normalized to 20 X 10 pixels. Average image (fused image) is formed from 10 images of each character. Bonding box around character is determined by using vertical and horizontal

projection of character. After cropping image to bounding box, it is resized to 10 X 10 pixels size. After that, thing is performed and thinned image is placed in one by one raw of 100 X 100 canvas.

[9] Renata F. P , et al., [2021], have proposed a modified quadratic classifier based scheme to recognize the offline numerals of six popular Indian scripts is proposed. Multilayer perceptron has been used for recognizing English characters.

[10] T. Anitha Pal, et al., [2021], have presented in their study the recognition of Hindi and English numerals by representing them in the form of exponential membership functions which serve as a fuzzy model. The recognition is carried out by modifying the exponential membership functions fitted to the fuzzy sets. The overall recognition rate is found to be 85% for Hindi numerals and 91% percent for English numerals.

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

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 digit 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 digit 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 the 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 modeled 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 modeled 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.

3.2 Proposed System

The project comes with the Technique of Digit recognition that is to take the image of digit from the online or offline records and recognise the digit .Digit recognition problem is a promising problem in handwriting recognition problem. It is also one of the challenging problems in computer vision and machine learning. It is so called challenging task as developing an accurate automated recognition of digits is difficult. The applications of digit recognition includes bank check processing and online form data submitted by users by using digital devices. The variation in digti among people makes it difficult to train the computers to recognize digits. This work employs machine learning techniques to address the digit recognition problem. To analyse various classification techniques and finding the best technique for digit recognition, the work focuses on utilizing a tool that would embrace multiple types of classification models with diverse performance metrics. The proposed model contains the four stages in order to classify and detect the digits:

- Pre - processing,
- Segmentation,
- Feature Extraction,
- Classification and Recognition.

Deep Learning has emerged as a central tool for self perception problems like understanding images, a voice from humans, robots exploring the world. The aim is to implement the concept of Convolutional Neural Network digit recognition. Understanding Convolutional Neural Network and applying it to the handwritten digit recognition system is the target of the proposed model.

3.3 Feasibility Study

From a technical perspective, since this project makes heavy use of numerical computations, using Octave is a wise choice as it will make the program more efficient.This software will also provide us with some libraries to read and manipulate the images that will make the implementation process easier. As for the

dataset to use in the testing of the project, the MNIST Database has been selected. it contains thousands of handwritten digits that have been used in the development of programs with a similar aim. This dataset is open for public use with no charges.

It is also very convenient for the project and will help us reduce the time by using directly as a test set without having to make one ourselves. Since all the tools to be used in this project are free of charge and very easy to use, it can be concluded that this project is very feasible in terms of financial resources, effort, and time.

3.3.1 Economic Feasibility

Automation of manual processes: Digit recognition using Convolutional Neural Network can automate processes that were previously done manually, such as reading and interpreting handwritten forms or documents. This can save time and reduce labor costs.

Cost effective solution: Using Convolutional Neural Network for handwritten digit recognition can be a cost effective solution compared to traditional methods such as hiring people to manually input data. Once the model has been trained, it can be used to recognize digits in large quantities of data without requiring significant ongoing costs.

Potential for new applications: Digit recognition using Convolutional Neural Network can enable the development of new applications and services, such as handwriting recognition software for mobile devices or online form processing services.

3.3.2 Technical Feasibility

The frontend and backend of the code have been compiled using visual studios which can be installed on any operating system, and is fully open source and can be accessed easiely 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. on another side the research concept of image processing and deep learning is a very trending topic nowadays and all the required libraries in python are available at free of cost. Amazon uses convolutional neural network for

hand- writing recognition to process and analyze scanned documents such as invoices and shipping labels. Google uses convolutional neural network for handwritten digit recognition in its Google Cloud Vision API, which provides image recognition services for various applications such as document processing, facial recognition, and object detection. IBM uses convolutional neural network for digit recognition in its Watson Visual Recognition service, which provides image analysis and recognition services for various industries including healthcare, retail, and finance.

3.3.3 Social Feasibility

Digit Recognition has been used to recognizing the digits from different sources like emails, bank cheque, postal codes, etc which can reduce human errors. The reason behind this is because this field can be further developed for malicious intentions, perhaps in the military field to be able to target certain people, and many other similar possibilities. That is why if image processing is not well managed and used with a wise mentality and good intentions, it could represent a threat to societies.

3.4 System Specification

This project needs the help of hardware and software requirements to be fit in the computer or laptop. The user and the toolkits and hardware and software requirements are required also.

3.4.1 Hardware Specification

- Processor: Intel(R) i3 or more, 2.00 GHz
- RAM: 4 GB
- Storage: 500 GB
- Internet connectivity: Yes
- Webcam connectivity: Yes

3.4.2 Software Specification

- Operating System – Windows 7 or more, ubuntu 16 or above, fedora 20 or above.
- Python – Python 2.7 or above
- OpenCV – OpenCV 3.2.0 or above
- Numpy – Python 2.7 3.5
- Tensorflow – Tensorflow 2.1.6 or above
- Pil – Pil 1.1.5 or above

3.4.3 Standards and Policies

Anaconda Prompt

Anaconda prompt is a type of command line interface which explicitly deals with the ML(MachineLearning) modules. and navigator is available in all the Windows,Linux and MacOS. The anaconda prompt has many number of IDE's which make the coding easier. The UI can also be implemented in python.

Standard Used: ISO/IEC 27001

Visual studio

The Visual Studio IDE is a creative launching pad that we use to edit, debug, and build code, and then publish an app. Over and above the standard editor and debugger that most IDEs provide, Visual Studio includes compilers, code completion tools, graphical designers, and many more features to enhance the software development process.

Standard Used: ISO/IEC 27001

Jupyter

It's like an open source web application that allows us to share and create the documents which contains the live code, equations, visualizations and narrative text. It can be used for data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning.

Standard Used: ISO/IEC 27001

Chapter 4

METHODOLOGY

4.1 General Architecture

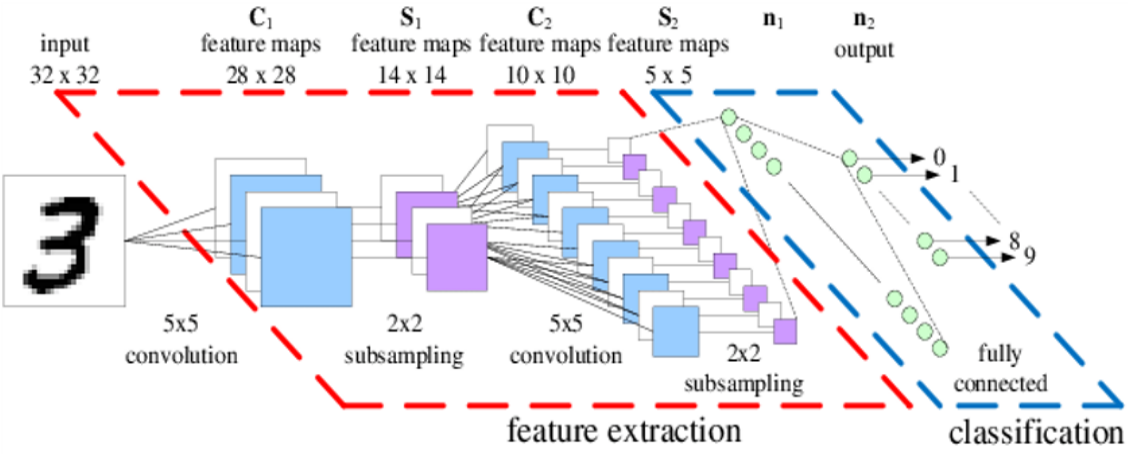

Figure 4.1: **General Architecture of Digit Recognition**

The Figure 4.1 describes the architecture diagram of Digit Recognition using Convolutional Neural Network. It shows us the steps involved in recognition the given Input Digit by using Convolutional Neural Network.

Pre-Processing: The role of the pre-processing step is it performs various tasks on the input image. It basically upgrades the image by making it reasonable for segmentation. The fundamental motivation behind pre-processing is to take off a fascinating example from the background. For the most part, noise filtering, smoothing and standardization are to be done in this stage. The pre-processing additionally characterizes a smaller portrayal of the example. Binarization changes over a gray scale image into a binary image.

Segmentation: Once the pre-processing of the input images is completed, subimages of individual digits are formed from the sequence of images.

Pre-processed 12 digit images are segmented into a subimage of individual digits, which are assigned a number to each digit. Each individual digit is resized into pixels. In this step an edge detection technique is being used for segmentation of dataset images.

Convolutional Layer: This layer is the first layer that is used to extract the various features from the input images. In this layer, the mathematical operation of convolution is performed between the input image and a filter of a particular size MxM. By sliding the filter over the input image, the dot product is taken between the filter and the parts of the input image with respect to the size of the filter (MxM).

Pooling Layer: In most cases, a Convolutional Layer is followed by a Pooling Layer. The primary aim of this layer is to decrease the size of the convolved feature map to reduce the computational costs. This is performed by decreasing the connections between layers and independently operates on each feature map. Depending upon method used, there are several types of Pooling operations. In Max Pooling, the largest element is taken from feature map. Average Pooling calculates the average of the elements in a predefined sized Image section. The total sum of the elements in the predefined section is computed in Sum Pooling.

Fully Connected Layer: The Fully Connected (FC) layer consists of the weights and biases along with the neurons and is used to connect the neurons between two different layers. These layers are usually placed before the output layer and form the last few layers of a convolutional neural network Architecture.

Activation Functions: An activation function in a neural network defines how the weighted sum of the input is transformed into an output from a node or nodes in a layer of the network. Sometimes the activation function is called a "transfer function." If the output range of the activation function is limited, then it may be called a "squashing function." Many activation functions are nonlinear and may be referred to as the "nonlinearity" in the layer or the network design. The choice of activation function has a large impact on the capability and performance of the neural network, and different activation functions may be used in different parts of the model.

4.2 Design Phase

4.2.1 Data Flow Diagram

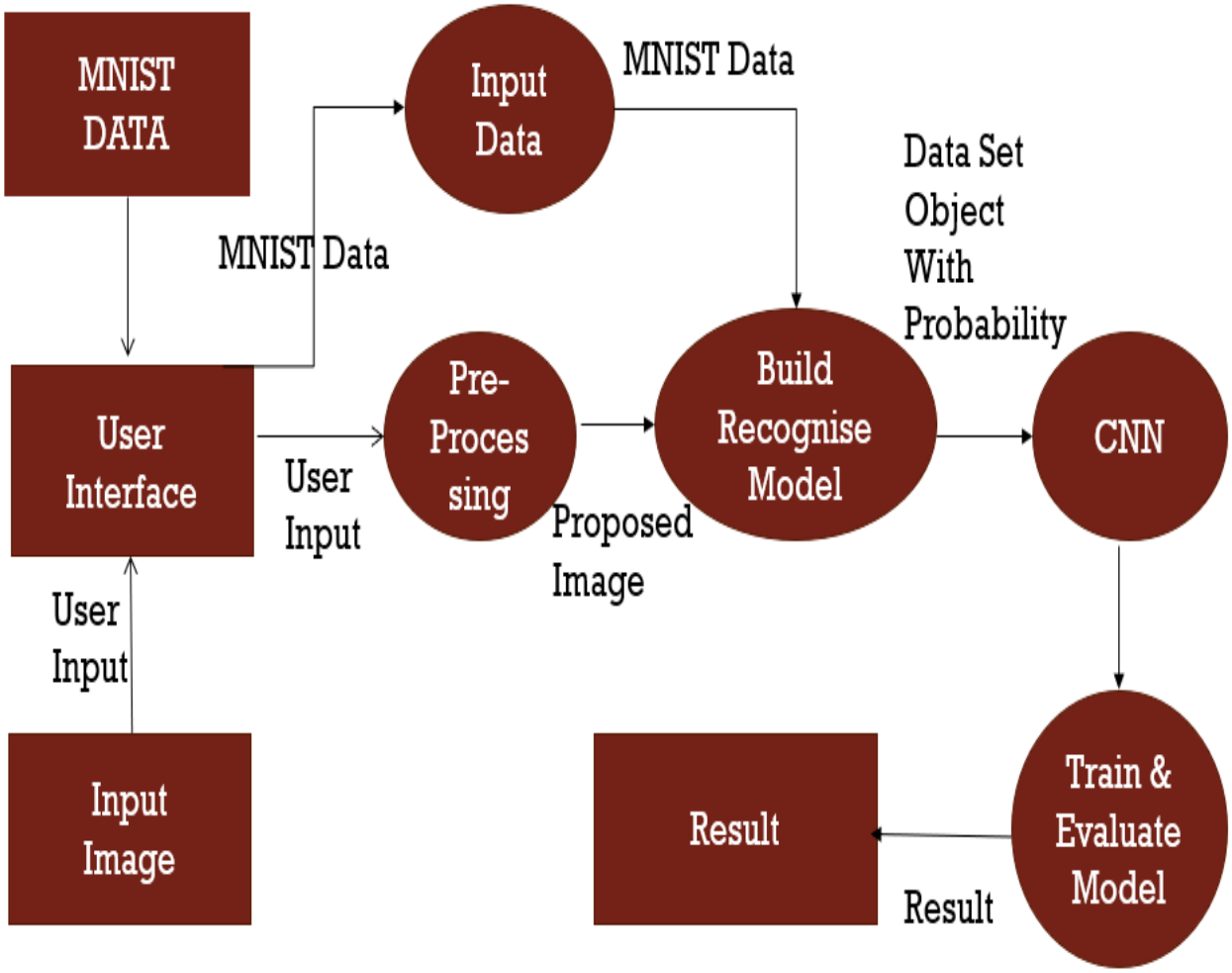

Figure 4.2: **Data Flow Diagram**

The following Figure 4.2 describes the Data flow diagram of the proposed system model. There are two ways to provide input to the system. The user can either upload the image of the digit he wants to detect or the data from the MNIST dataset. The input images are pre-processed. After pre-processing the image is sent to convolutional neural network filter where it converts the three dimensional image into one dimensional matrix after in pooloing layer using max pooling the size of the data is reduced nearly to half of its orginal size. The convoluted image is compared with the already present test images and Using the different classifiers the recognized digits are recognized' accuracy is compared and the result is obtained.

4.2.2 Use Case Diagram

Figure 4.3: **Use Case Diagram**

The Figure 4.3 describes about the use case diagram. A use case diagram at its simplest is a representation of a user interaction with the system that shows the relationship between the user and the different use cases in which the user is involved The actor can be human or other external system that provides input digit . The input given by actor is sent to digit recognition system which contains various use cases, The user can upload the image of the digit he wants to detect. The given input images are pre-processed . The pre-processing additionally characterizes a smaller portrayal of the example. Binarization changes over a gray scale image into a binary image. Once the pre-processing of the input images is completed, sub images of individual digits are formed from the sequence of images. Pre-processed digit images are segmented into a sub image of individual digits. The given input pass undergoes all the uses case then finally digit is recognised.

4.2.3 Class Diagram

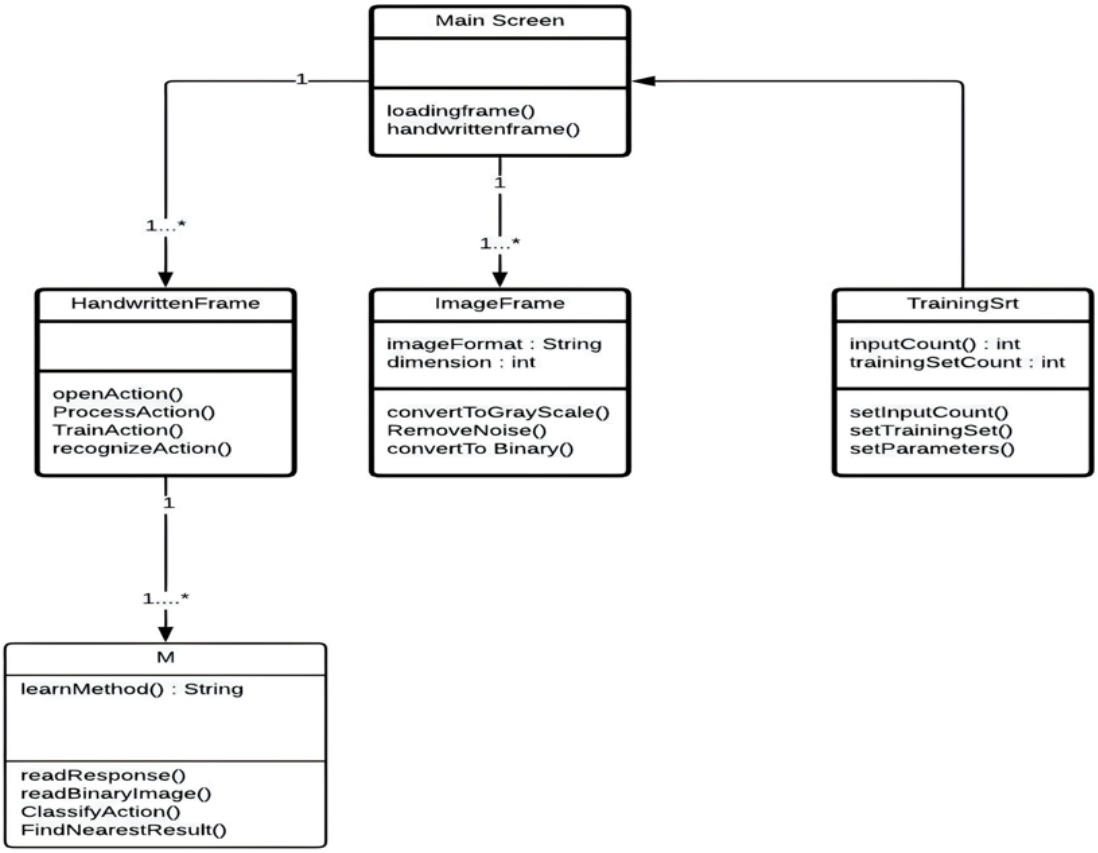

Figure 4.4: **Class Diagram**

The Figure 4.4 describes the Class diagram of model class structure and contents using design elements such as classes(handwritten frame,image frame,main screen), packages and objects. Class diagram describes three perspectives when designing a system Conceptual, Specification, Implemenation. Classes are composed of three things: name, attributes and operations. The handwritten frame class contains operations like trainAction(), recogniseAction() and mainframe class contains the operations to load and recognise data . Class diagrams also display relations such as containment, inheritance, associations etc. The association relationship is most common relationship in a class diagram. The association shows the relationship between instances of classes. The purpose of class diagram is to model the static view of an application. Class diagrams are the only diagrams which can be directly mapped with object oriented languages and thus widely used at the time of construction. The data from image class is loaded using load frame operation present in mainframe class and using the method actions present in the M class the digit is recognized.

4.2.4 Sequence Diagram

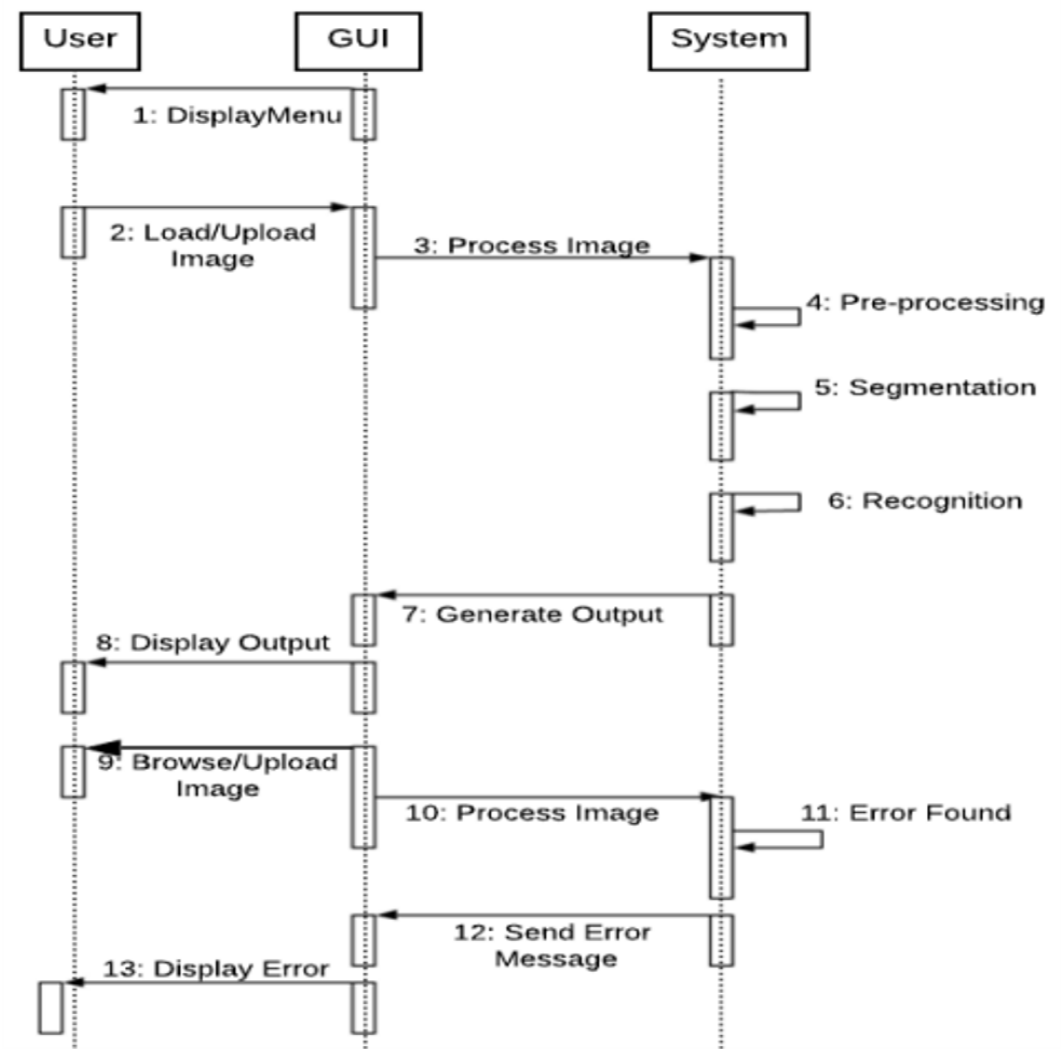

Figure 4.5: **Sequence Diagram**

The Figure 4.5 describes the sequence diagram of the system model. The figure describes the sequence of steps to be taken while performing execution. The convolutional neural network model works in the following sequence. User uploads a particular image of any digit which he wants to recognize. The image will be processed by the system. On running the system code the output is generated that shows which is the digit uploaded by the user and also displays the accuracy rate predicted by the model. On uploading image with different resolutions other than the one mentioned in the code, the output generated shows error, and displays an error message to the user.

4.2.5 Activity Diagram

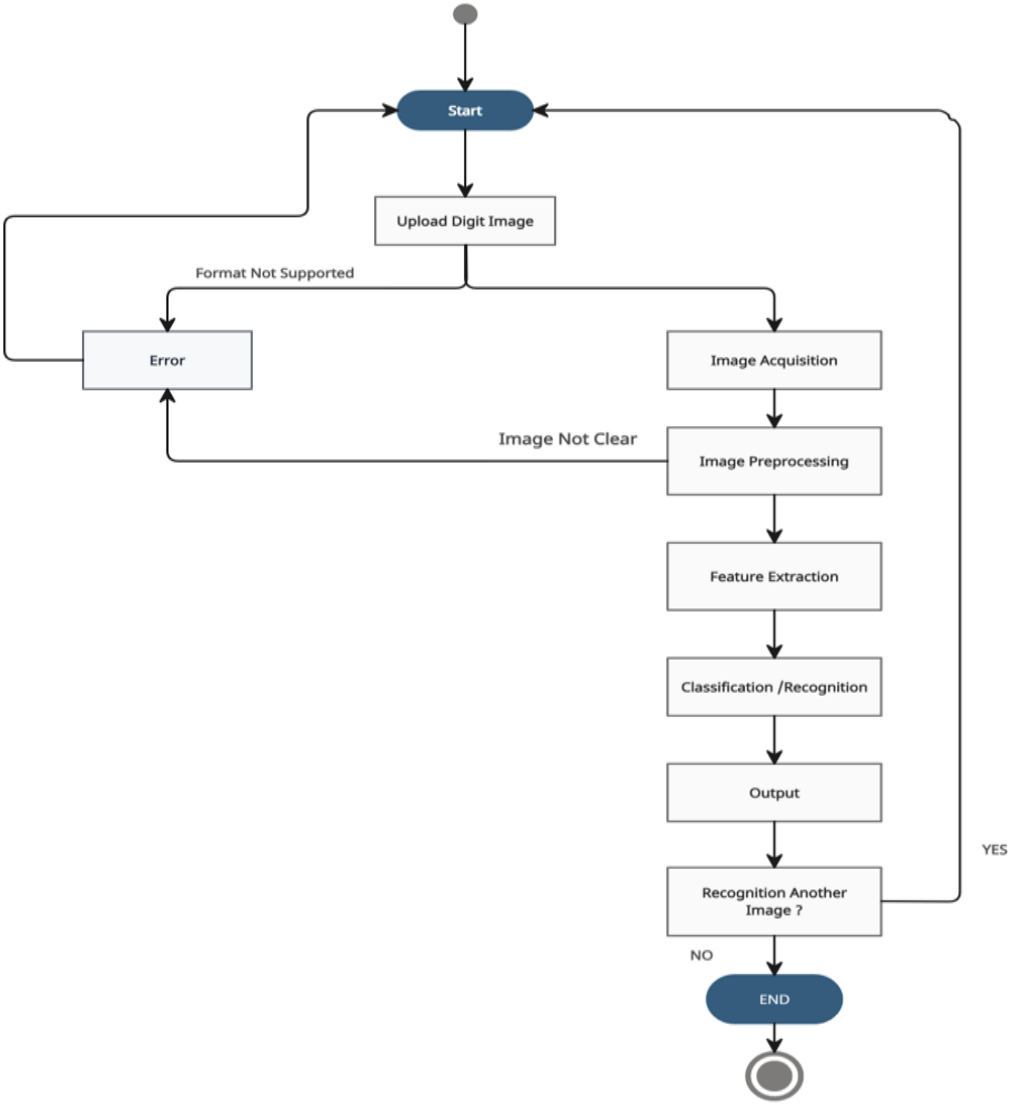

Figure 4.6: **Activity Diagram**

As shown in the Figure 4.6 activity diagrams describe how activities are coordinated to provide a service which can be at different levels of abstraction. Typically, an event needs to be achieved by some operations, particularly where the operation is intended to achieve a number of different things that require coordination, or how the events in a single use case relate to one another, in particular, use cases where activities may overlap and require coordination. It is also suitable for modeling how a collection of use cases coordinate to represent business workflows.

4.3 Algorithm & Pseudo Code

4.3.1 Convolutional Neural Network

Step-1: Import necessary libraries such as TensorFlow and Keras.
Step-2: Collecting the MNIST data and then pre-process the data.
Step-3: Then use convolutional layers to separate the data into the feature data and target variable. Then split the data into training and testing.
Step-4: Then use pooling layers to reduce the spatial dimensions of the extracted features.
Step-5: After using the pooling layer then flatten the output from the convolutional layers into a one-dimensional vector.
Step-6: Then connect the flattened vector to one or more fully connected layers.
Step-7: After connecting the layers then use the generate class predictions function using a softmax activation function.
Step-8: Then train the model using backpropagation and gradient descent to minimize the loss function.
Step-9: After training the model with the training dataset, the testing dataset is utilized to evaluate its performance.
Step-10: A specific subset of the overall MNIST dataset serves as the testing dataset, based on which the accuracy of the proposed model is computed.
Step-11: After opening the website the user is proposed with two options.
Step-12: Choose file by clicking this button one can give input image to system.
Step-13: When the prediction button is clicked , the model will give predicted digit based on given input by the user.

4.3.2 Pseudo Code

```
pd.read csv(train.csv)as matrix()
 c i f Decision  Tree Classifier()
# training dataset
 t r a i n data  set[0:21000,1:]
trainlabel=dataset [0:21000,0]
cffit(train,trainlabel
#testing data
 t e s t i n g dataset [21000:1: ]
actual label=dataset(21000:,0)
d=testing[0]
d.shape=(28,28)
plt.imshow(d,cmap=plt.cm.gray)print(clf.predict([testing[0]])
```

```
plt.title( Sample  Digit Recognized )
plt.show ( )
```

4.4 Module Description

4.4.1 Data Collection

At the beginning of the project, all the required modules for training the model are imported. Importing the dataset and commencing work on it is straightforward because the Keras library already includes many datasets, including MNIST. The MNIST load data function is called to retrieve the training data along with its labels, as well as the testing data with its labels.

MNIST dataset consists of 60,000 training samples and 10,000 testing samples. Each sample is a grayscale image with a resolution of 28x28 pixels, representing a single digit from 0 to 9. MNIST serves as a benchmark for many machine learning algorithms, particularly in the field of computer vision, as it provides a standardized dataset for testing and comparing the performance of different models.

4.4.2 Data Pre Processing

Data preprocessing plays a crucial role in preparing datasets for training deep learning models in handwritten digit recognition. It begins with gathering a dataset of handwritten digit images, such as those from MNIST. After exploring the dataset to understand its size and characteristics, flawed images are removed. Subsequently, pixel values are normalized for consistency, and data augmentation techniques, such as rotations or flips, are applied to enhance model performance. The dataset is then split into training, validation, and testing sets to facilitate effective model training and evaluation. Following a systematic pipeline ensures consistency, while efficient data loading accelerates the training process. In summary, proper preprocessing lays the foundation for successful model training and accurate digit recognition.

4.4.3 Model Training

Model training for handwritten digit recognition using deep learning is crucial for building an effective recognition system. It involves selecting a suitable neural network architecture like Convolutional Neural Networks (CNNs) known for their excellence in image classification. Initializing network parameters, choosing a loss function such as categorical cross-entropy, and selecting optimization algorithms like SGD or Adam are vital steps. Hyperparameter tuning, adjusting learning rates and batch sizes, enhances model performance. Throughout training, parameters are iteratively updated to minimize the loss function. Monitoring training progress detects overfitting, while early stopping prevents it. Finally, the trained model is assessed on a separate testing dataset using metrics like accuracy and precision to ensure its accuracy and generalization ability. This holistic approach ensures the development of a robust handwritten digit recognition system.

4.4.4 Testing the Model

To initiate the training of the model, the model.fit() function of Keras is called. This function requires parameters such as the training data, validation data, epochs, and batch size. Training the model typically takes some time. Upon successful completion of model training, the weights and model definition can be saved in the 'mnist.h5' file. Optimizers are algorithms or methods used to adjust the attributes of the neural network, such as weights and learning rate, to minimize losses. There are various types of optimizers, each with its own approach to handling weights and bias inputs. In this project, the backpropagation algorithm is modified with the Adam optimizer instead of the gradient descent optimizer. Initially, gradient descent was considered, but the Adam or Stochastic gradient descent optimizer was found to be more effective, especially considering the varying inputs from the different pixels.

4.5 Steps to execute/run/implement the project

4.5.1 Import required Libraries

- At the project beginning, all the modules needed for training model are imported.
- One can easily import the dataset and start working on that because the Keras library already contains many datasets and MNIST is one of them.

- The mnist load data() function is called to get training data with its labels and also the testing data with its labels.

4.5.2 Training Model

- After completing data preprocessing, the CNN model is created which consists of various convolutional and pooling layers alongside a 3x3 sized kernel.
- The model will then be trained on the basis of training and validation data with the help of several python libraries such as TensorFlow, Pillow, OpenCV, Tkinter, Numpy that were preloaded to perform these specific tasks.

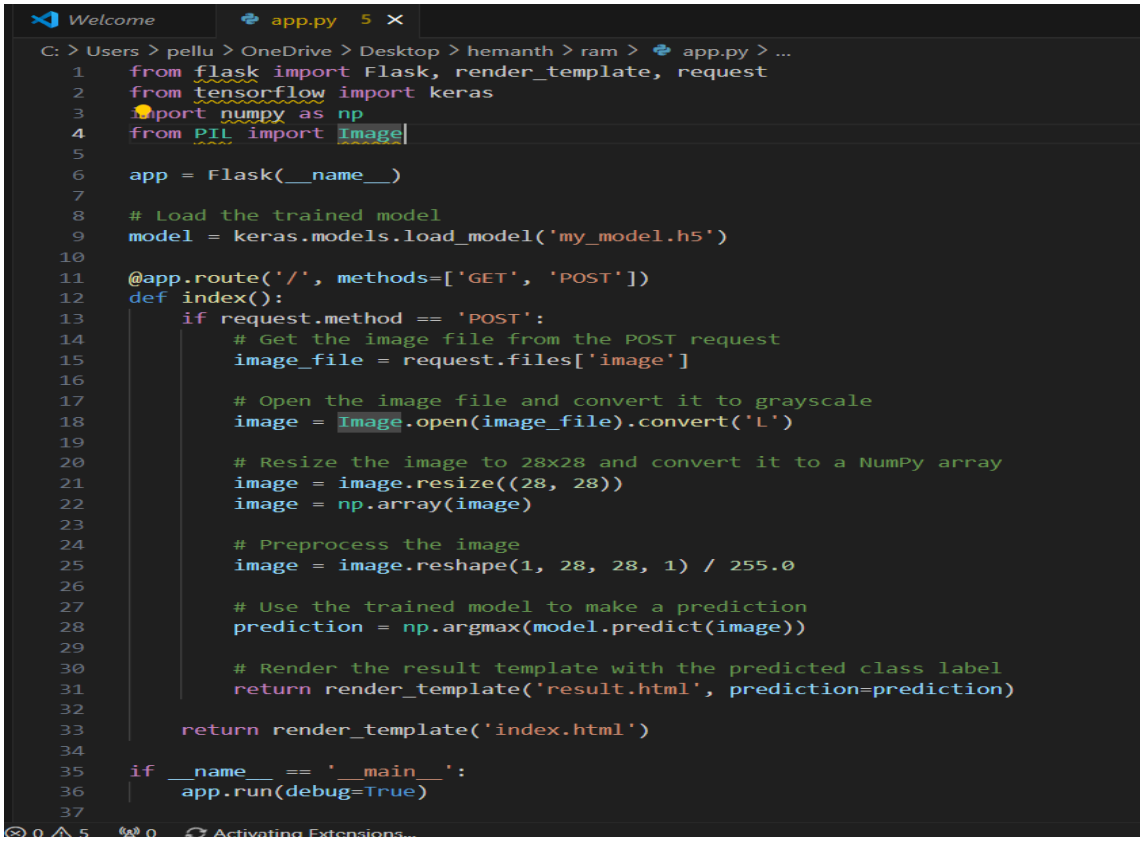

Figure 4.7: **Training Data**

4.5.3 Testing Accuracy

- After the model is trained using the training dataset, the testing dataset is used to evaluate how well model works.

- A particular part of the overall MNIST dataset is used as the testing dataset on the basis of which the accuracy is computed for the proposed model.
- By using CNN algorithm one can get accuracy upto 99% .

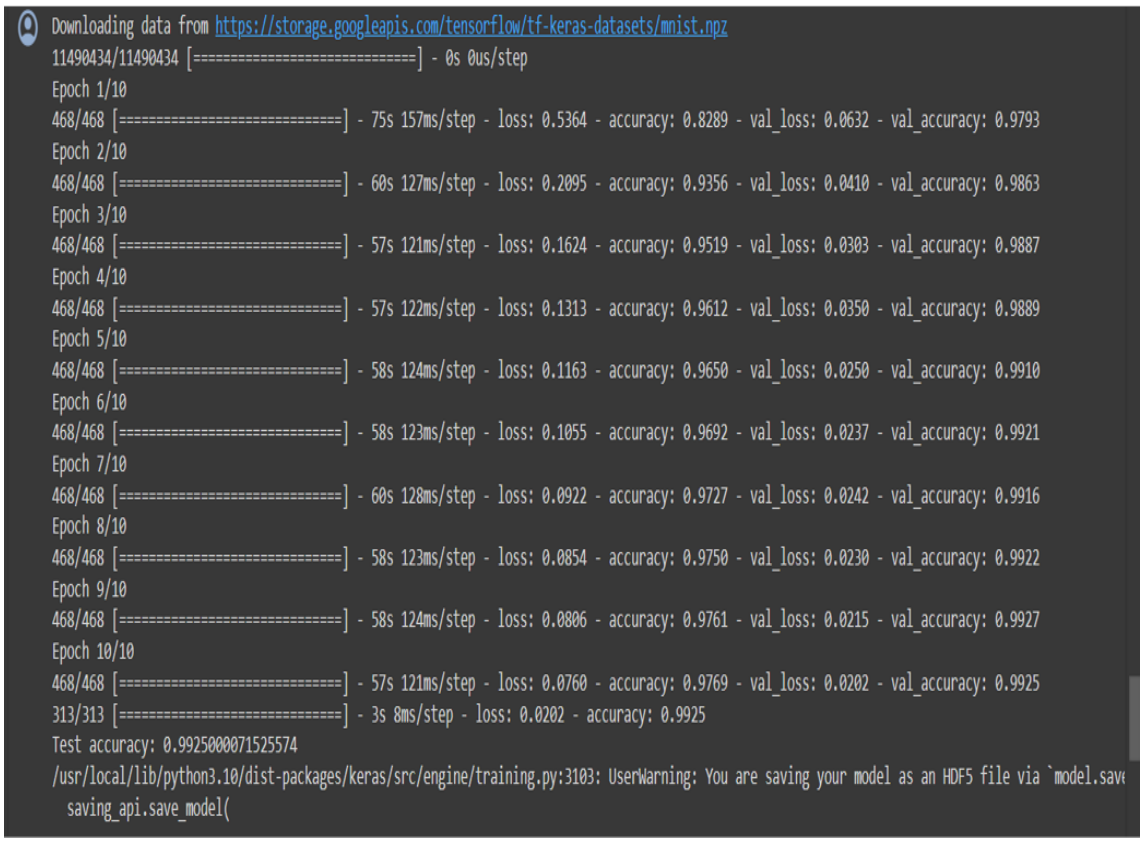

Figure 4.8: **Testing Accuracy**

As shown in Figure 4.8, this describes the training of a convolutional neural network (CNN) model on the MNIST dataset to recognize handwritten digits. After training, the model is saved to a file named 'my_model.h5'.

4.5.4 Digit Recognition

- After opening the app the user is proposed with two options.
- Choose file by clicking this button one can give input image to the system.
- Predict Number: This option uses the CNN model to predict the string of digits given by the user.

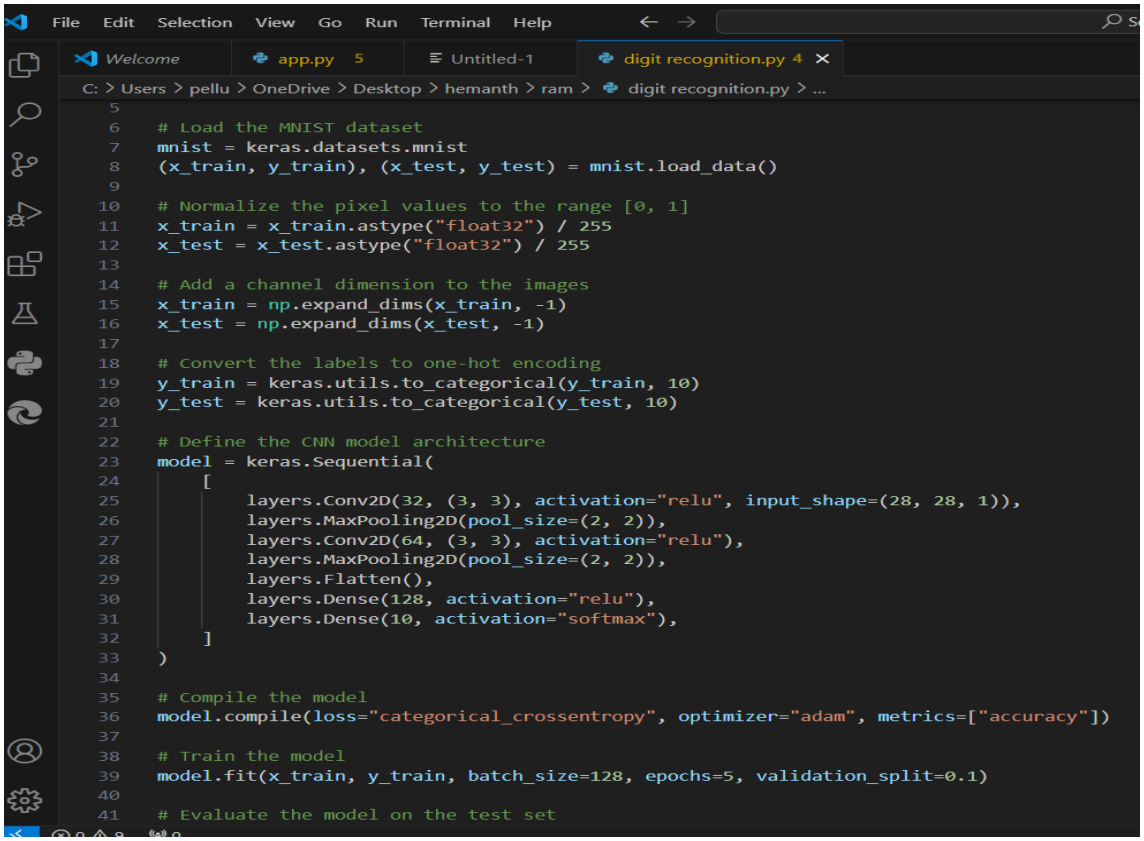

Figure 4.9: **Digit Recognition**

As shown in the Figure 4.9 describes how a Flask web application is created to enable users to upload images of handwritten digits and receive predictions from the trained model. When a user uploads an image, it is pre processed, passed through the trained model, and the predicted digit is displayed on the web interface.

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Input Design

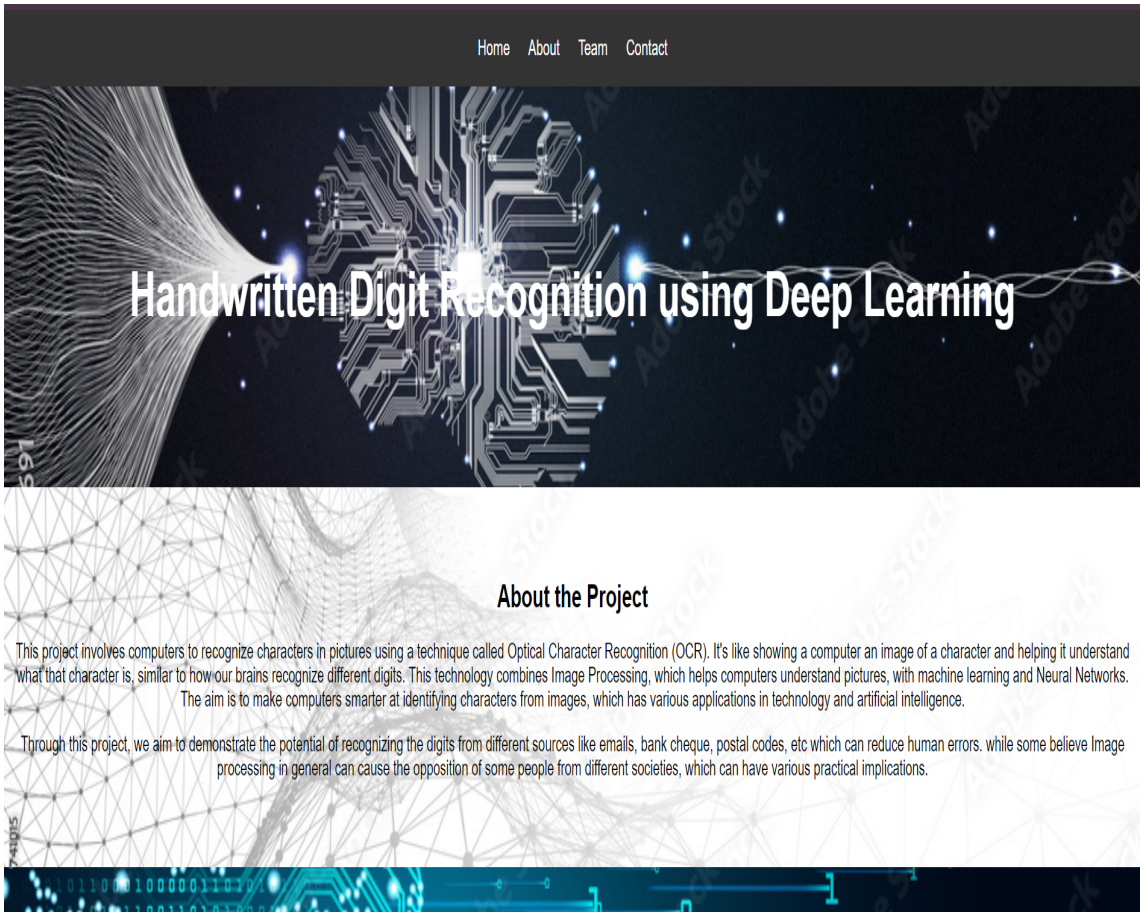

Figure 5.1: **Home Page**

As shown in the Figure 5.1 shows the Home Page of digit recognition system in which one can give input digit to recognise. There are two ways to provide input to the system. The user can either upload the image of the digit he wants to detect. After providing the Digit, one should click predict button to get the result.

5.1.2 Sample Input

The Following are some sample grey scale images which are are given as input.

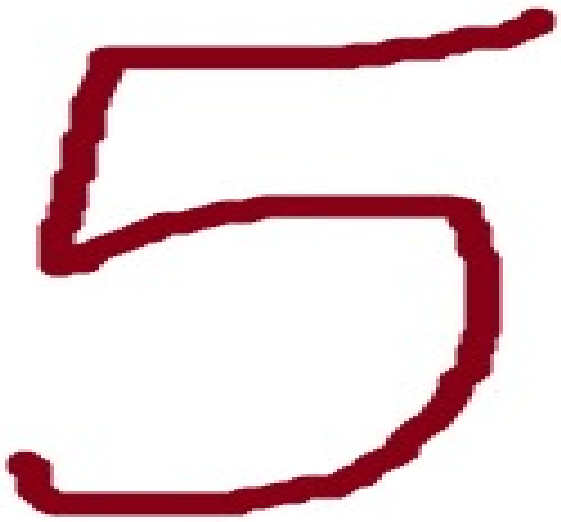

Figure 5.2: **Digit Input Image**

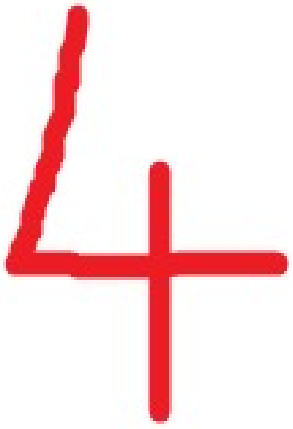

Figure 5.3: **MNIST Test Input**

The user can give input either by drawing it in any drawing app or from any available datasets. The given input images are converted into one dimensional matrix using CNN algorithm.

5.1.3 Output Design

The given input images are pre-processed. After preprocessing the image is sent to CNN filter where it converts the three dimensionnal image into one dimensional matrix after in pooloing layer using max pooling the size of the data is reduced nearly to half of its orginal size. The convoluted image is compared with the already present test images and based on the accuracy the image is recognised.

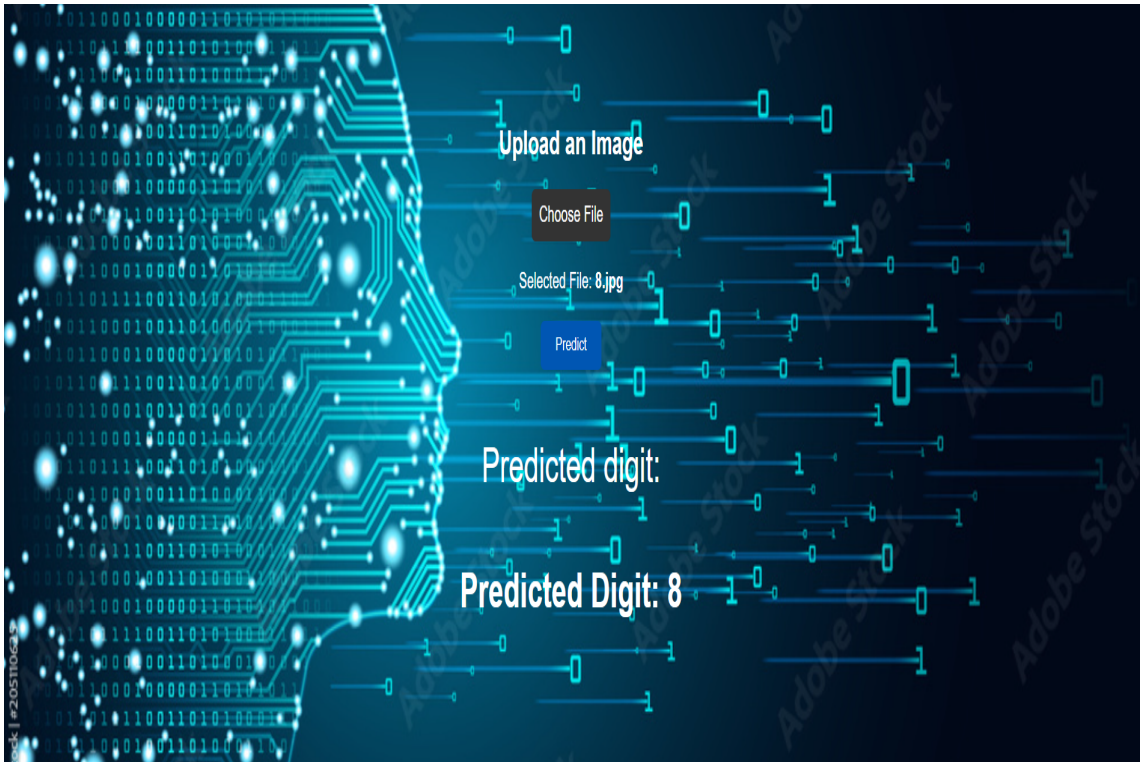

Figure 5.4: **Prediction Page**

5.2 Testing

Testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is defect free. It involves the execution of a software component or system component to evaluate one or more properties of interest. Software testing also helps to identify errors, gaps, or missing requirements in contrary to the actual requirements

5.3 Types of Testing

5.3.1 Unit Testing

Input

```
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten, Conv2D, MaxPooling2D, Dropout
from tensorflow.keras.datasets import mnist
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Load the MNIST dataset
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Normalize the pixel values to the range [0, 1]
x_train = x_train.astype("float32") / 255
x_test = x_test.astype("float32") / 255

# Reshape the data to have a single channel
x_train = np.expand_dims(x_train, -1)
x_test = np.expand_dims(x_test, -1)

# Convert labels to one-hot encoding
y_train = tf.keras.utils.to_categorical(y_train, 10)
y_test = tf.keras.utils.to_categorical(y_test, 10)
```

In the handwritten digit recognition project, unit testing plays a critical role in ensuring the correctness and robustness of individual components. Unit tests are created to validate the functionality of key aspects such as data preprocessing, model training, and prediction. For example, tests are written to verify that data preprocessing functions correctly normalize pixel values and augment the dataset. Additionally, tests are created to ensure that the model architecture is accurately defined, the loss function is appropriate, and the optimizer functions correctly during model training. Furthermore, unit tests can validate the accuracy of digit predictions made by the trained model on sample input images. By conducting unit testing, any errors or inconsistencies can be identified and addressed early in the development process, thereby enhancing the reliability and performance of the handwritten digit recognition system.

Test result

SUCCESS

5.3.2 Integration Testing

Input

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Handwritten Digit Recognition using Deep Learning</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            box-sizing: border-box;
            background-color: #f2f2f2;
        }
        header {
            background-color: #333;
            color: #fff;
            padding: 20px 0;
            text-align: center;
        }
        nav ul {
            list-style-type: none;
            margin: 0;
            padding: 0; }
        nav ul li {
            display: inline;
            margin: 0 10px; }
        nav ul li a {
            color: #fff;
            text-decoration: none; }
```

Integration testing in the handwritten digit recognition project ensures that all parts of the system work seamlessly together. It involves testing how different components, such as data preprocessing, model training, and deployment with Flask, interact with each other. For example, integration tests verify that the trained model is correctly loaded and used within the Flask application. They also check if Flask routes handle image uploads, preprocess images, make predictions using the

model, and display the results accurately. Integration tests ensure that the entire system functions smoothly and reliably, allowing for accurate digit recognition in real-world applications.

Test result

SUCCESS

5.3.3 System Testing

Input

```
from flask import Flask, render_template, request
from tensorflow import keras
import numpy as np
from PIL import Image

app = Flask(_name_)

# Load the trained model
model = keras.models.load_model('my_model.h5')

@app.route('/', methods=['GET', 'POST'])
def index():
    if request.method == 'POST':
        # Get the image file from the POST request
        image_file = request.files['image']

        # Open the image file and convert it to grayscale
        image = Image.open(image_file).convert('L')

        # Resize the image to 28x28 and convert it to a NumPy array
        image = image.resize((28, 28))
        image = np.array(image)

        # Preprocess the image (normalize and reshape)
        image = image.reshape(1, 28, 28, 1) / 255.0

        # Use the trained model to make a prediction
        prediction = model.predict(image)

        # Get the predicted digit (index of the highest probability)
        predicted_digit = np.argmax(prediction)

        # Render the result template with the predicted class label
        return render_template('RESULT2.html', prediction=predicted_digit)

```

```
36      return render_template('INDEX2.html')
37
38  if __name__ == '__main__':
39      app.run(debug=True)
```

System testing in the handwritten digit recognition project involves evaluating the entire system to ensure it works as intended in real-world situations. This testing phase examines the system's performance, functionality, reliability, and user-friendliness. It checks if the digit recognition process,from image upload to displaying predictions,functions smoothly. Additionally,system testing assesses how well the system handles different scenarios, such as varying image qualities or user interactions. By conducting systematic system testing, the digit recognition system can be ensured to meet user needs and operate effectively across different environments.

Test Result

SUCCESS

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

Key rationale toward Optical Character Recognition (OCR) from image includes features extraction technique supported by a classification algorithm for recognition of characters based on the features. Previously, several algorithms for feature classifications and extraction have been utilized for the purpose of character recognition. But, with the advent of convolutional neural networks in deep learning, no separate algorithms are required for this purpose. However, in the area of computer vision, deep learning is one of the outstanding performers for both feature extraction and classification. However, convolutional neural networks architecture consists of many nonlinear hidden layers with a enormous number of connections and parameters. Therefore, to train the network with very less amount of samples is a very difficult task. In convolutional neural networks, only few set of parameters are needed for training of the system. So, convolutional neural networks is the key solution capable to map correctly datasets for both input and output by varying the trainable parameters and number of hidden layers with high accuracy . For the experiments and verification of system's performance, the normalized standard MNIST dataset is utilized.

The new system proposed for Optical Character Recognition (OCR) is a big step forward. Instead of using lots of different algorithms to figure out what characters are in an image, Convolutional Neural Networks (CNN) are being used. These networks are like super-smart filters that can learn to recognize patterns in images all on their own. It's kind of like how our brains see things; there's no need for someone to explain every detail. Convolutional Neural Networks can extract local features and learn complex representations from input images. Despite their complexity, convolutional neural networks require fewer trainable parameters compared to traditional methods, making them more efficient and scalable.

6.2 Comparison of Existing and Proposed System

Existing system:(Multilayer Perceptron Algorithm)

A neural network based classifier, called Multilayer perception (MLP), is used to classify the handwritten digits. Multilayer perceptron consists of three different layers, input layer, hidden layer and output layer. Each of the layers can have certain number of nodes also called neurons and each node in a layer is connected to all other nodes to the next layer.For this reason it is also known as feed forward network. The number of nodes in the input layer depends upon the number of attributes present in the dataset. The number of nodes in the output layer relies on the number of apparent classes exist in the dataset. The convenient number of hidden layers or the convenient number of nodes in a hidden layer for a specific problem is hard to determine. But in general, these numbers are selected experimentally. In multilayer perceptron, the connection between two nodes consists of a weight. During training process, it basically learns the accurate weight adjustment which is corresponds to each connection.For the learning purpose, it uses a supervised learning technique named as Back propagation algorithm.

Existing system:(Support Vector Machine)

SVM or Support Vector Machine is a specific type of supervised machine learning method that intents to classify the data points by maximizing the margin among classes in a high-dimensional space . SVM is a representation of examples as points in space, mapped due to the examples of the separate classes are divided by a fair gap that is as extensive as possible. After that new examples are mapped into that same space and anticipated to reside to a category based on which side of the gap they fall on . The optimum algorithm is developed through a “training” phase in which training data are adopted to develop an algorithm capable to discriminate between groups earlier defined by the operator (e.g. patients vs. controls), and the “testing” phase in which the algorithm is adopted to blind-predict the group to which a new perception belongs. It also provides a very accurate classification performance over the training records and produces enough search space for the accurate classification of future data parameters. Hence it always ensures a series of parameter combinations no less than on a sensible subset of the data. In SVM it’s better to scale the data always; because it will extremely improve the results. Therefore be cautious with big dataset, as it may leads to the increase in

the training time.

Proposed system:(CNN Algorithm)

To reduce error and obtain more efficiency overall, CNN can be used to implement digit recognition systems. so, the proposed system uses CNN with multiple pooling and convolutional layers alongside a kernel of 3x3 size. The model uses 60,000 28*28 grayscale images during the training process model is trained through a standard 5 epochs to achieve accuracy of the order of 99.16% which is much higher as compared to the traditional algorithms such as SVM, Multilayer Perceptron, Bayes Net, Random Forest, etc. Used to implement digit recognition systems. Aim of the proposed work is to recognize user defined digits and identify the complete number(in decimal number system by default) that is input by the user which is then converted to binary/octal/hexadecimal number system according to the user's choice. A Graphical User Interface (GUI) is created for the same in which the user is displayed with a canvas widget that anybody can use to draw digit strings for recognition and conversion accordingly. The canvas can then be cleared for further continuation.

Model	**Accuracy**	**Limitations**
Multilayer Perceptron	90%	Prone to overfitting
Support Vector Machine	78%	Sensitivity to hyperparameters
Neural Network	89%	Limited generalization
Decision Trees	81.5%	Prone to overfitting
Random Forests	85%	Slow training time
K-Nearest Neighbors	70%	Slow prediction time
Logistic Regression	76%	Limited to linear relations
Naive Bayes	80%	Assumes independence of features
Gradient Boosting	85%	Prone to overfitting
Ensemble Learning	80%	Complex implementation

Table 6.1: Comparison of Existing and Proposed System

6.3 Sample Code

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

# Load the MNIST dataset
mnist = keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Normalize the pixel values to the range [0, 1]
x_train = x_train.astype("float32") / 255
x_test = x_test.astype("float32") / 255

# Add a channel dimension to the images
x_train = np.expand_dims(x_train, -1)
x_test = np.expand_dims(x_test, -1)

# Convert the labels to one-hot encoding
y_train = keras.utils.to_categorical(y_train, 10)
y_test = keras.utils.to_categorical(y_test, 10)

# Define the CNN model architecture
model = keras.Sequential(
    [
        layers.Conv2D(32, (3, 3), activation="relu", input_shape=(28, 28, 1)),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, (3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Flatten(),
        layers.Dense(128, activation="relu"),
        layers.Dense(10, activation="softmax"),
    ]
)

# Compile the model
model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])

# Train the model
model.fit(x_train, y_train, batch_size=128, epochs=10, validation_split=0.1)

# Evaluate the model on the test set
test_loss, test_acc = model.evaluate(x_test, y_test)
print("Test accuracy:", test_acc)

model.save("my_model.h5")
```

Output

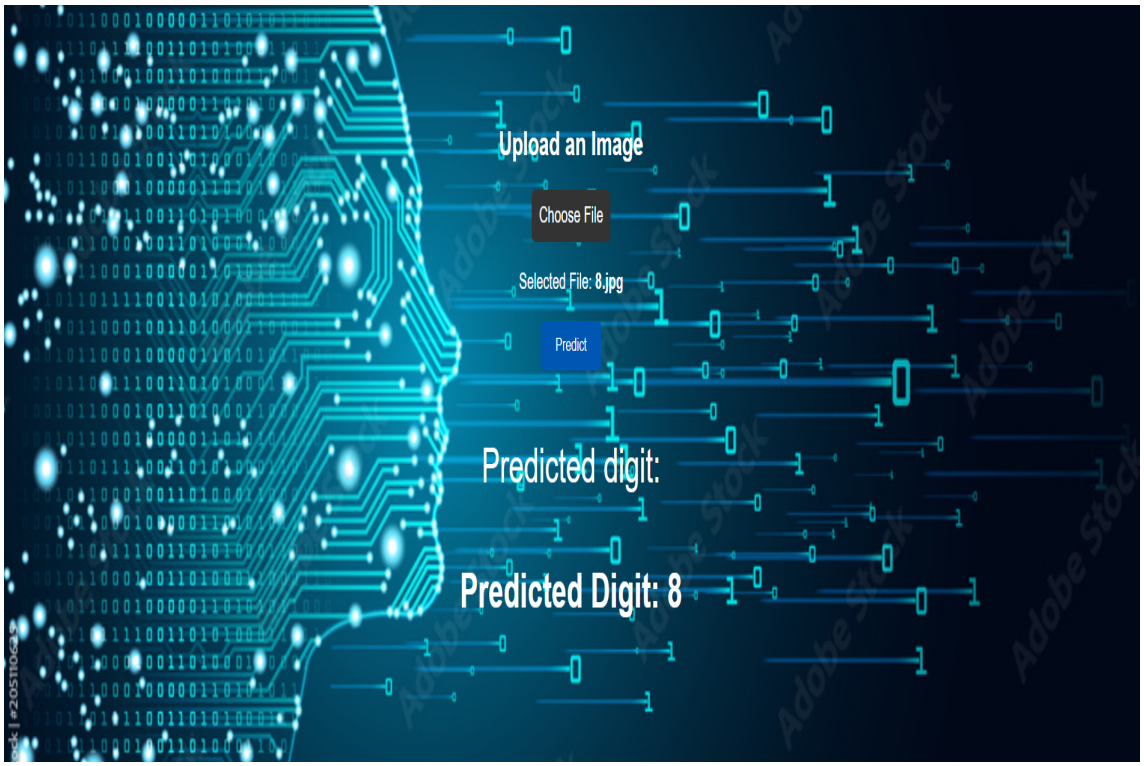

Figure 6.1: **Prediction Output using Convolutional Neural Networks**

The given input images are pre-processed. After preprocessing the image is sent to CNN filter where it converts the image into one dimensional matrix after in pooloing layer using max pooling the size of the data is reduced nearly to half of its orginal size. The convoluted image is compared with the already present test images and based on the accuracy the image is recognised.

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

The performance of Convolutional Neural Networks for digit recognition performed significantly. The proposed method obtained 92% accuracy and is able to identify real world images as well. The loss percentage in both training and evaluation is less than 0.1, which is negligible. The only challenging part is the noise present in the real world image, which needs to look after. The learning rate of the model is much dependent on the number of dense neurons and the cross-validation measure. Recognition of digit using Convolutional Neural Networks with Rectified Linear Units activation is implemented. The proposed Convolutional Neural Networks framework is well equipped with suitable parameters for high accuracy of MNIST digit classification. Time factor is also considered for training the system. Afterward, for further verification of accuracy, the system is also checked by changing the number of Convolutional Neural Networks layers. It is worth mentioning here that Convolutional Neural Networks architecture design consists of two convolutional layers. The experimented results demonstrate that the proposed Convolutional Neural Networks framework for MNIST dataset exhibits high performance in terms of time and accuracy as compared to previously proposed systems. Consequently, digits are recognized with high accuracy (92.21%).

Hand Written Digit classification with the good accuracy and prediction is much required for this real world. Hence, it is essential that the Convolutional Neural Networks model should identify any kind of hand written digit in minimal time. Convolutional Neural Networks model is the prominent tool used for many Machines Learning models, because it is multilayer algorithm. Convolutional Neural Networks

algorithm automatically extracts the important feature of the input image without the human supervision. Prediction model is also built to check whether the model is predicting the proper output. Convolutional Neural Networks Algorithm yet again proved to be very efficient to any classification done to the dataset of images when compare to other Algorithms.

7.2 Future Enhancements

This project will be enhanced with a great field of machine learning and artificial intelligence. The world can think of a software which can recognise the text from a picture and can show it to the others, for example a the shop name detector, or this project can be extended to a greater concept of all the character sets in the world. This project has not gone for the total english alphabet because there will be more and many more training sets and testing values that the neural network model will not be enough to detect. Think of a artificial intelligence modeled car sensor going with a direction modeling in the roadside, user shall give only the destination. All of these enhancement is an application of the texture analysis where advanced image processing, Neural network model for training and advanced artificial intelligence concepts will come. These applications can be modeled further. As this project is fully done by free and available resources and packages this can be also a limitation of the project. The fund is very important because all machine learning libraries and advanced packages are not available for free. Unless of those the most of the visualizing platforms like on which developers are doing some works like Watson Studio or Amazon Web Services. These all are mainly paid platforms where a lot of Machine learning projects are going on.

Chapter 8

PLAGIARISM REPORT

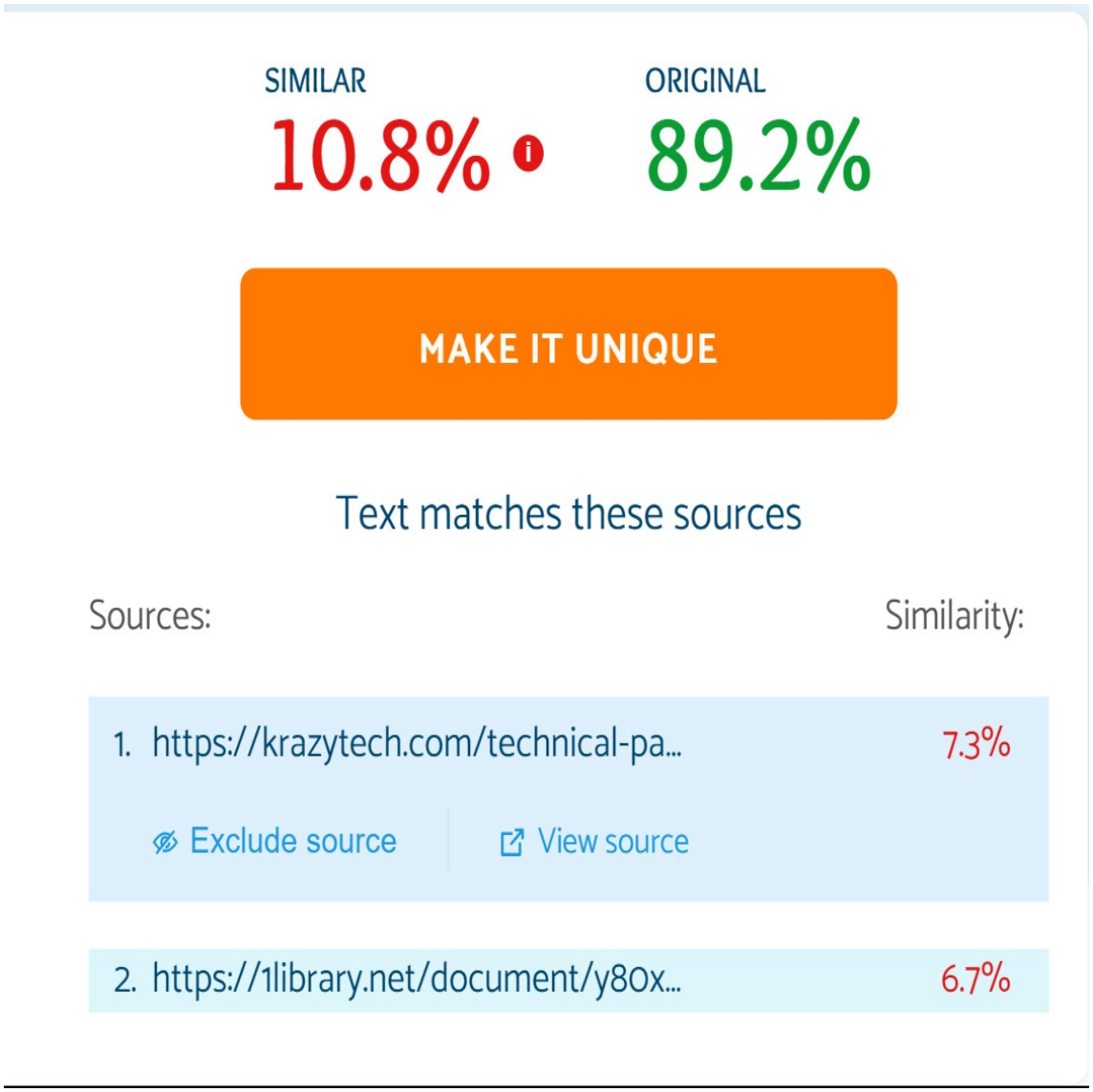

Figure 8.1: **Plagiarism Report**

Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

HTML CODE

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Handwritten Digit Recognition using Deep Learning</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            box-sizing: border-box;
            background-color: #f2f2f2;
        }
        header {
            background-color: #333;
            color: #fff;
            padding: 20px 0;
            text-align: center;
        }
        nav ul {
            list-style-type: none;
            margin: 0;
            padding: 0;
        }
        nav ul li {
            display: inline;
            margin: 0 10px;
        }
        nav ul li a {
            color: #fff;
            text-decoration: none;
        }
```

```
        .hero {
            background-image: url('https://as1.ftcdn.net/v2/jpg/02/22/28/06/1000_F_222280691_8wcIFV4NMxB6UxMbw381bh3XD1nYu8As.jpg'); /* Absolute path to your background image */
            background-size: cover;
            background-position: center;
            color: #fff;
            text-align: center;
            padding: 100px 0;
        }
        .hero h1 {
            font-size: 3em;
            margin-bottom: 20px;
        }
        .about {
            background-image: url('https://as1.ftcdn.net/v2/jpg/04/49/74/10/1000_F_449741015_7wFz1d5e1voLD3arDjsyST3TtQkZAhg6.jpg'); /* Absolute path to your background image */
            background-size: cover;
            background-position: center;
            padding: 50px 20px;
            text-align: center;
        }
        .file-upload {
            background-image: url('https://as1.ftcdn.net/v2/jpg/02/05/11/06/1000_F_205110625_baTKpiTjtpRkL1UBdPaWN1nspkn7DPPf.jpg'); /* Absolute path to your background image */
            background-size: cover;
            background-position: center;
            background-color: #444444;
            color: #ffffff;
            padding: 70px 20px;
            text-align: center;
        }
        .file-upload input[type="file"] {
            display: none;
        }
        .file-upload label {
            background-color: #333333;
            color: #ffffff;
            padding: 10px;
            border-radius: 5px;
            cursor: pointer;
            margin-bottom: 20px;
            display: inline-block;
        }
        .file-upload label:hover {
            background-color: #555555;
        }
        .selected-file {
```

```
            display: none;
            margin-bottom: 20px; /* Increase the gap between selected file display and buttons */
        }
        .selected-file-name {
            color: #ffffff;
            font-weight: bold;
        }
        .predict-button {
            background-color: #007bff;
            color: white;
            padding: 10px 20px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
        }
        .predict-button:hover {
            background-color: #0056b3;
        }
        #prediction {
            font-size: 36px;
            margin-top: 50px;
        }
        .team {
            background-color: #fff;
            padding: 50px 20px;
            text-align: center;
        }
        .team-member {
            display: inline-block;
            margin: 20px;
            vertical-align: top;
        }
        .supervisor {
            background-color: #333;
            color: #fff;
            padding: 50px 0;
            font-family: Arial, sans-serif;
            margin:0 auto;
            text-align: center;
        }
        .contact {
            background-color: #fff;
            padding: 50px 20px;
            text-align: center;
        }
        .contact form {
            display: inline-block;
            text-align: left;
        }
        .contact input[type="text"],
```

```
        .contact input[type="email"],
        .contact textarea {
            width: 100%;
            padding: 10px;
            margin-bottom: 10px;
            border-radius: 5px;
            border: 1px solid #fff;
        }
        .contact input[type="submit"] {
            background-color: #4CAF50;
            color: white;
            padding: 10px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
        }
        .contact input[type="submit"]:hover {
            background-color: #3e8e41;
        }
        #prediction {
            font-size: 36px;
            margin-top: 50px;
        }
    </style>
</head>
<body>
    <header>
        <nav>
            <ul>
                <li><a href="#">Home</a></li>
                <li><a href="#about">About</a></li>
                <li><a href="#team">Team</a></li>
                <li><a href="#contact">Contact</a></li>
            </ul>
        </nav>
    </header>

    <div class="hero">
        <h1>Handwritten Digit Recognition using Deep Learning</h1>
    </div>

    <div id="about" class="about">
        <h2>About the Project</h2>
        <p>This project involves computers to recognize characters in pictures using a technique
            called Optical Character Recognition (OCR). It's like showing a computer an image of a
            character and helping it understand what that character is, similar to how our brains
            recognize different digits. This technology combines Image Processing, which helps
            computers understand pictures, with machine learning and Neural Networks. The aim is to
            make computers smarter at identifying characters from images, which has various
            applications in technology and artificial intelligence.
```

```
172         </p>
173         <p>Through this project, we aim to demonstrate the potential of recognizing the digits from
                different sources like emails, bank cheque, postal codes, etc which can reduce human
                errors. while some believe Image processing in general can cause the opposition of some
                people from different societies, which can have various practical implications.</p>
174     </div>
175
176     <div class="file-upload">
177         <h2>Upload an Image</h2>
178         <input type="file" id="fileInput" accept="image/*" onchange="displayFileName(this)">
179         <label for="fileInput">Choose File</label>
180         <div class="selected-file" id="selectedFile">
181             Selected File: <span class="selected-file-name"></span>
182         </div>
183         <button class="predict-button" onclick="predictDigit()">Predict</button>
184     </div>
185
186     <div id="team" class="team">
187         <h2>Meet the Team</h2>
188         <div class="team-member">
189             <h3>Harsha Vardhan Reddy</h3>
190             <p>Student </p>
191         </div>
192         <div class="team-member">
193             <h3>Narayana Swamy Naidu</h3>
194             <p>Student </p>
195         </div>
196         <div class="team-member">
197             <h3>Pavan Kumar Pelluru</h3>
198             <p>Student </p>
199         </div>
200     </div>
201
202     <div class="supervisor">
203         <h3>SUPERVISOR</h3>
204         <p>M. Shankar</p>
205     </div>
206
207     <div id="contact" class="contact">
208         <h2>Contact Us</h2>
209         <form method="POST" action="submit_form.php">
210             <input type="text" name="name" placeholder="Your Name">
211             <br>
212             <input type="email" name="email" placeholder="Your Email">
213             <br>
214             <textarea name="message" placeholder="Your Message"></textarea>
215             <br>
216             <input type="submit" value="Send Message">
217         </form>
218     </div>
```

```
    <script>
        function displayFileName(input) {
            const selectedFile = document.getElementById('selectedFile');
            const fileName = document.querySelector('.selected-file-name');
            if (input.files && input.files[0]) {
                fileName.textContent = input.files[0].name;
                selectedFile.style.display = 'block';
            } else {
                fileName.textContent = '';
                selectedFile.style.display = 'none';
            }
        }

        function uploadImage() {
            const selectedFile = document.getElementById('fileInput').files[0];
            if (selectedFile) {
                // Add your upload image logic here
                alert('Image uploaded successfully!');
            } else {
                alert('Please select a file before submitting.');
            }
        }

        function predictDigit() {
            const selectedFile = document.getElementById('fileInput').files[0];
            if (selectedFile) {
                // Add your predict digit logic here
                alert('Digit prediction initiated!');
            } else {
                alert('Please select a file before predicting.');
            }
        }
    </script>
</body>
</html>
```

PYTHON CODE

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

# Load the MNIST dataset
mnist = keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Normalize the pixel values to the range [0, 1]
```

```
x_train = x_train.astype("float32") / 255
x_test = x_test.astype("float32") / 255

# Add a channel dimension to the images
x_train = np.expand_dims(x_train, -1)
x_test = np.expand_dims(x_test, -1)

# Convert the labels to one-hot encoding
y_train = keras.utils.to_categorical(y_train, 10)
y_test = keras.utils.to_categorical(y_test, 10)

# Define the CNN model architecture
model = keras.Sequential(
    [
        layers.Conv2D(32, (3, 3), activation="relu", input_shape=(28, 28, 1)),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, (3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Flatten(),
        layers.Dense(128, activation="relu"),
        layers.Dense(10, activation="softmax"),
    ]
)

# Compile the model
model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])

# Train the model
model.fit(x_train, y_train, batch_size=128, epochs=5, validation_split=0.1)

# Evaluate the model on the test set
test_loss, test_acc = model.evaluate(x_test, y_test)
print("Test accuracy:", test_acc)

model.save("my_model.h5")
from flask import Flask, render_template, request
from tensorflow import keras
import numpy as np
from PIL import Image

app = Flask(__name__)

# Load the trained model
model = keras.models.load_model('my_model.h5')




@app.route('/', methods=['GET', 'POST'])
def index():
```

```
    if request.method == 'POST':
        # Get the image file from the POST request
        image_file = request.files['image']

        # Open the image file and convert it to grayscale
        image = Image.open(image_file).convert('L')

        # Resize the image to 28x28 and convert it to a NumPy array
        image = image.resize((28, 28))
        image = np.array(image)

        # Preprocess the image
        image = image.reshape(1, 28, 28, 1) / 255.0

        # Use the trained model to make a prediction
        prediction = np.argmax(model.predict(image))

        # Render the result template with the predicted class label
        return render_template('result.html', prediction=prediction)

    return render_template('index.html')

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

9.2 Poster Presentation

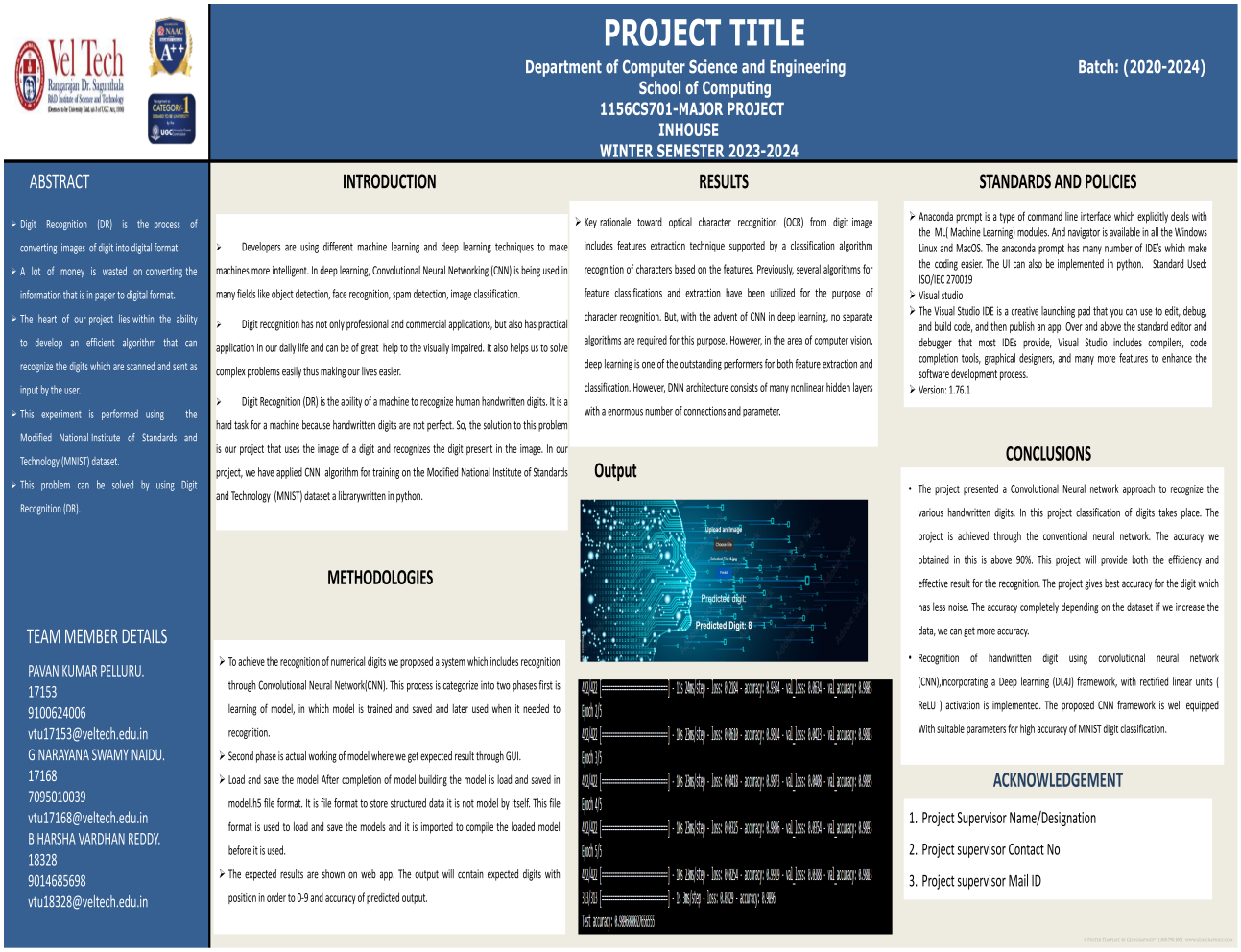

Figure 9.1: **Poster Presentation**

References

[1] A. B. Siddique, M. M. R. Khan, R. B. Arif, and Z. Ashrafi, "Study and Observation of the Variations of Accuracies for Digits Recognition with Various Hidden Layers and Epochs using Neural Network Algorithm," in 2019 4th International Conference on Electrical Engineering and Information Communication Technology (iCEEiCT), Jan 2019, pp. 118- 123: IEEE.

[2] C. C. Park, Y. Kim, and G. Kim, "Retrieval of sentence sequences for an image stream via coherence recurrent convolutional networks," IEEE transactions on pattern analysis and machine intelligence, vol. 40, no. 4, pp. 945-957, June 2019

[3] Dala, N.; Triggs, B. "Histograms of oriented gradients for human detection", In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05), San Diego, CA, USA, 20–26 June 2020.

[4] J. Pradeep, E. Srinivasan and S. Himavathi, "Diagonal Based Feature Extraction For Alphabets Recognition System Using Neural Network", Inter- national Journal of Computer Science Information Technology (IJCSIT), Vol 3, No 1, Sept 2020.

[5] Mayank Jain, Harshith Guptha, "Digits Recognition using CNN," in 2021 4th International Conference on Electrical Engineering and Information Communication Technology (iCEEiCT), 2021, pp. 118- 123: IEEE.

[6] M. M. A. Ghosh and A. Y. Maghari, "A Comparative Study on Digit Recognition Using Neural Networks," 2020 International Conference on Promising Electronic Technologies (ICPET), Deir El-Balah, 2021, pp. 77-81.

[7] R. B. Arif, M. A. B. Siddique, M. M. R. Khan, and M. R. Oishe, "Study and Observation of the Variations of Accuracies for Digits Recognition with Various Hidden Layers and Epochs using Convolutional Neural Network," in 43

2021 4th International Conference on Electrical Engineering and Information Communication Technology (iCEEiCT), 2021, pp. 112-117: IEEE

[8] R. Alhajj and A. Elnagar, “Multiagents to separating connected digits,” in IEEE. Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans, vol. 35, no. 5, pp. 593-602, Sept. 2021.

[9] Renata F. P. Neves, Alberto N. G. Lopes Filho, Carlos A.B.Mello, CleberZanchettin, “A SVM Based Off-Line ”Digit Recognizer”, International conference on Systems, Man and Cybernetics, IEEE Xplore, pp. 510-515, 9-12 Oct, 2021.

[10] T. Anita Pal Dayashankar Singh, “Character Recognition Using Neural,” Network International Journal of Computer Science Communication. Vol. 1, No. 2, July- December 2010, pp. 141-144,Dec 2021.

[11] T.Som, Sumit Saha, “Character Recognition Using Fuzzy Membership Function”, International Journal of Emerging Technologies in Sciences and Engineering, Vol.5, No.2, pp. 11-15, 2022.

[12] U. Pal, T. Wakabayashi and F. Kimura, “Numeral recognition of six popular scripts,” Ninth International conference on Document Analysis and Recognition ICDAR 07, Vol.2, pp.749- 753, 2022.