### HANDWRITTEN DIGIT RECOGNITION USING MACHINE LEARNING



Mini Project submitted in partial fulfillment of the requirement for the award of the

degree of

**BACHELOR OF TECHNOLOGY IN**

**Information Technology** Under the esteemed guidance of **Ms. GNANA MAYURI**

**Assistant Professor**

By

**T. HARINI - 19R11A1252**

**M. SAI CHARAN - 19R11A1228**

**K. GOVTHAM -19R11A1221**



**Department of Information Technology**

**Accredited by NBA**

**Geethanjali College of Engineering and Technology**

**(UGC Autonomous)**

(Affiliated to J.N.T.U.H, Approved by AICTE, New Delhi) Cheeryal (V), Keesara (M), Medchal.Dist.-501 301.

**DEC-2022**

Geethanjali College of Engineering & Technology

**(UGC Autonomous)**

(Affiliated to JNTUH, Approved by AICTE, New Delhi) Cheeryal (V), Keesara(M), Medchal Dist.-501 301.

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**Accredited by NBA**



# CERTIFICATE

This is to certify that the B.Tech Mini Project report entitled “**HANDWRITTEN DIGIT RECOGNITION USING MACHINE LEARNING”** is a bonafide work done by by **T.HARINI (19R11A1252), M.SAI CHARAN (19R11A1252), K.GOVTHAM**

**(19R11A1221)** in partial fulfillment of the requirement of the award for the degree of Bachelor of Technology in **“INFORMATION TECHNOLOGY”** from Jawaharlal Nehru Technological University, Hyderabad during the year 2022-2023.

**Internal Guide HOD - IT**

**Ms. K. Gnana Mayuri Dr.K.Srinivas**

**Assistant professor Professor**

**External Examiner**

Geethanjali College of Engineering & Technology

**(UGC Autonomous)**

**(Affiliated to JNTUH Approved by AICTE, New Delhi) Cheeryal (V), Keesara(M), Medchal Dist.-501301**

**DEPARTMENT OF INFORMATION TECHNOLoGY**

**Accredited by NBA**

**DECLARATION BY THE CANDIDATE**

We, **T. Harini, M. Sai Charan, K. Govtham** bearing Roll Nos. **19R11A1252, 19R11A1228, 19R11A1221** hereby declare that the project report entitled **“HANDWRITTEN DIGIT RECOGNITION USING MACHINE LEARNING”** is done

under the guidance of **Ms. K.Gnana Mayuri**, **Assistant professor**, Department of Information Technology, Geethanjali College of Engineering and Technology, is submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Information Technology**.

This is a record of bonafide work carried out by us in **Geethanjali College of Engineering and Technology** and the results embodied in this project have not been reproduced or copied from any source. The results embodied in this project report have not been submitted to any other University or Institute for the award of any other degree or diploma.

**T. HARINI - 19R11A1252,**

**M. SAI CHARAN - 19R11A1228,**

**K. GOVTHAM -19R11A1221,**

**Department of IT, Geethanjali College of Engineering and Technology,**

**Cheeryal.**

# ACKNOWLEDGEMENT

We are greatly indebted to the authorities of Geethanjali College of Engineering and Technology, Cheeryal, Medchal Dist., for proving us the necessary facilities to successfully carry out this mini project work titled **“HAND WRITTEN DIGIT RECOGNITION USING MACHINE LEARNING”.**

Firstly, we thank and express our solicit gratitude to **Professor, Dr.K.Srinivas, HOD, IT department**, Geethanjali College of Engineering and Technology, for his invaluable help and support which helped us a lot in successfully completing our mini project.

Secondly, we express our gratitude to **Assistant Prof. Ms. Gnana Mayuri**, internal guide, Geethanjali College of Engineering and Technology, for his suggestion and encouragement which helped us in the successful completion of our mini project.

We would like to express our sincere gratitude to our **Principal Dr. S. UDAY KUMAR**

for providing the necessary infrastructure to complete our project.

We convey our gratitude to our **Chairman, Mr. G. RAVINDER REDDY,** for his invaluable support and encouragement for propelling our innovation forward.

Finally, we would like to express our heartfelt thanks to our parents who were very supportive both financially and mentally and for their encouragement to achieve our set goals.

**T. HARINI - 19R11A1252**

**M. SAI CHARAN - 19R11A1228**

**K. GOVTHAM -19R11A1221**

i

**ABSTRACT**

We introduce the task of colorization from natural language, a previously unexplored source of colour specifications. We develop a fully automatic image colorization system. Our approach leverages recent advances in deep networks, exploiting both low-level and semantic representations. As many scene elements naturally appear according to multimodal colour distributions, we train our model to predict per-pixel colour histograms.

This paper aims to transform a grayscale image with 8-bit pixel representation to a pseudo color image with 24-bit pixel representation. The system proposed is robust and flexible with image resolution and reference color image used .The algorithm is unsupervised. Firstly, the input grey image is mapped with the reference gray image formed by the reference color image and then applying k-mean clustering similar gray levels are clustered to 256 clusters. This method is used to enhance, segment and analyze the image in better way, even though they are of gray in nature. After some many experiments there is still gap between the output pseudo color image and actual color image. Thus, the work may include more realistic color image transformation and robust output.

ii

## LIST OF IMAGES

|  |  |  |
| --- | --- | --- |
| **S.No** | **Name** | **Page no.** |
| 1. | Output 1 | 21 |
| 2. | Output 2 | 22 |
|  |  |  |
| 3. | Plagiarism Report | 30 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure no.** | **Figure Description** | **Page no.** |
| 4.1 | System Architecture | 9 |
| 4.2 | Use case Diagram | 10 |
| 4.3 | Sequence Diagram | 13 |
| 4.4 | Data Flow Diagram | 12 |

## LIST OF ABBREVIATIONS

|  |  |  |
| --- | --- | --- |
| **S. No** | **Abbreviation** | **Full form** |
| 1 | **CNN** | Convolutional Neural Network |
| 2 | **MNIST** | Modified National Institute of Standards and Technology |

**TABLE OF CONTENTS**

S. No Contents Page no

**Abstract**

**List of Figures List of Tables**

**List of Screen shots**

**List of Symbols & Abbreviations**

1. [Introduction 1](#_TOC_250002)
   1. [About the project](#_TOC_250001)
   2. Objective
2. [System Analysis 3](#_TOC_250000)
   1. Existing System
   2. Proposed System
      1. Details
      2. Impact on Environment
      3. Safety
      4. Ethics
      5. Cost
      6. Type
      7. Standards
   3. Scope of the Project
   4. Modules Description
   5. System Configuration
3. **Literature Overview 7**
4. **System Design 9**
   1. System Architecture
   2. UML Diagrams
   3. System Design
5. **Sample Code 14**
   1. Coding
6. **Output Screens 25**
7. **Conclusion 28**
   1. Conclusion
   2. Further Enhancements
8. **Bibliography 29**
   1. Books References
   2. Websites References

**9. Appendices**

**10. Plagiarism Report**

## INTRODUCTION

### ABOUT THE PROJECT

Recognition is identifying or distinguishing a thing or an individual from the past experiences or learning. Similarly, Digit Recognition is nothing but recognizing or identifying the digits in any document. Digit recognition framework is simply the working of a machine to prepare itself or interpret the digits. Handwritten Digit Recognition is the capacity of a computer to interpret the manually written digits from various sources like messages, bank cheques, papers, pictures, and so forth and in various situations for web based handwriting recognition on PC tablets, identifying number plates of vehicles, handling bank cheques, digits entered in any forms etc.

Handwriting recognition of characters has been around since the 1980s. The task of handwritten digit recognition, using a classifier, has extraordinary significance and use such as – online digit recognition on PC tablets, recognize zip codes on mail, processing bank check amounts, numeric sections in structures filled up by hand (for example ‐ tax forms) and so on. There are diverse challenges faced while attempting to solve this problem. The handwritten digits are not always of the same size, thickness, or orientation and position relative to the margins. The main objective was to actualize a pattern characterization method to perceive the handwritten digits provided in the MINIST data set of images of handwritten digits (0‐9).

.

### PROBLEM IDENTIFICATION AND OBJECTIVE

The goal of this project is to create a model that will be able to recognize and determine the handwritten digits from its image by using the concepts of Convolution Neural Network. Though the goal is to create a model which can recognize the digits, it can be extended to letters and an individual’s handwriting. The major goal of the proposed system is Understanding Convolutional Neural Network, and applying it to the handwritten recognition system.

* The handwritten digit recognition is the solution to this problem which uses the image of digit and recognizes the digit present in the image.
* To implement a handwritten digit recognition app using the MNIST dataset and by using a special type of deep neural network that is Convolutional Neural Networks.

## SYSTEM ANALYSIS

### EXISTING SYSTEM

These days, an ever-increasing number of individuals use pictures to transmit data. It is additionally main stream to separate critical data from pictures. Image Recognition is an imperative research area for its generally used applications. In general, the field of pattern recognition, one of the difficult undertakings is the precise computerized recognition of human handwriting. Without a doubt, this is a very difficult issue because there is an extensive diversity in handwriting from an individual to another individual. In spite of the fact that, this difference does not make any issues to people, yet, anyway it is increasingly hard to instruct computers to interpret general handwriting. For the image recognition issue, for example, handwritten classification, it is essential to make out how information is depicted onto images.**.**

### 2.1.1 DISADVANTAGES

✈Time consuming

✈ Requires lot of manual effort

### PROPOSED SYSTEM

* + - The proposed handwritten digit recognition system follows the standard model of feature based classification systems consisting of the digit image database, an essential feature extraction sub-block and a main classification sub-block. the mnist benchmark database of handwritten digits has been considered in this work.
    - We are going to implement a handwritten digit recognition app using the MNIST dataset.
    - We will create our CNN model in Python data science project. A CNN model generally consists of convolutional and pooling layers.
    1. **ADVANTAGES**

✈Reduces manual work.

✈ Speeds up the process

✈ Increases accuracy

### IMPACT ON ENVIRONMENT

No impact on the environment.

Doesn’t cause any damage to the environment.

### COST

Eliminate the need of expensive image transferring equipments for astronomical images.

### ETHICS

This project follows the general Machine Learning ethics. This system does not harm an individual in any way.

### SAFETY

Handwritten Digit Recognition is a secured application where the data provided by the users is kept safe with us. We constantly monitor and work towards closing any threats that might put it at any risk. Our application will not display any disturbing advertisements or pop ups.

### TYPE

The goal of this project is to create a model that will be able to recognize and determine the handwritten digits from its image by using the concepts of Convolution Neural Network. Though the goal is to create a model which can recognize the digits, it can be extended to letters and an individual’s handwriting. The major goal of the proposed system is understanding Convolutional Neural Network, and applying it to the handwritten recognition system.

### MODULES DESCRIPTION

### There are three modules in our project. They are listed below with detailed descriptions.

1. Input Module
2. Processing Module
3. Output Module

Input Module: Takes audio file from the dataset.

Processing Module: Converts speech to text and then applies classifiers. Output module: Displays the recognized emotion.

### SYSTEM CONFIGURATION

* + 1. **HARDWARE REQUIREMENTS**

✈Processor : Any updated Processor

✈ Hard Disk : 1 TB

✈ Ram : 4 GB or above

### SOFTWARE REQUIREMENTS

✈Operating System : Windows 10 (any windows operating system)

✈Coding Language : Python

✈Front-End : Python

✈ IDE : Anaconda IDE - Jupyter Notebook

## 3. LITERATURE OVERVIEW

An early notable attempt in the area of character recognition research was by Grimsdale in 1959. The origin of a great deal of research work in the early sixties was based on an approach known as analysis by-synthesis method suggested by Eden in 1968. The great importance of Eden's work was that he formally proved that all handwritten characters are formed by a finite number of schematic features, a point that was implicitly included in previous works. This notion was later used in all methods in syntactic (structural) approaches of character recognition.

1. K. Gaurav, Bhatia P. K. , his paper deals with the various pre-processing techniques involved in the character recognition with different kind of images ranges from a simple handwritten form based documents and documents containing colored and complex background and varied intensities.In this, different preprocessing techniques like skew detection and correction, image enhancement techniques of contrast stretching, binarization, noise removal techniques, normalization and segmentation, morphological processing techniques are discussed.
2. Sandhya Arora , used four feature extraction techniques namely, intersection, shadow feature, chain code histogram and straight line fitting features. Shadow features are computed globally for character image while intersection features, chain code histogram features and line fitting features are computed by dividing the character image into different segments.
3. Brakensiek, J. Rottland, A. Kosmala, J. Rigoll, in their paper a system for off-line cursive handwriting recognition is described which is based on Hidden Markov Models (HMM) using discrete and hybrid modeling techniques. Handwriting recognition experiments using a discrete and two different hybrid approaches, which consist of a discrete and semi-continuous structures, are compared. It is found that the recognition rate performance can be improved of a hybrid modeling technique for HMMs, which depends on a neural vector quantizer (hybrid MMI), compared to discrete and hybrid HMMs, based on tired mixture structure (hybrid - TP), which may be caused by a relative small data set.

## CNN ARCHITECTURE

CNNs are a class of Deep Neural Networks that can recognize and classify particular features from images and are widely used for analyzing visual images. Their applications range from image and video recognition, image classification, medical image analysis, computer vision and natural language processing. The term ‘Convolution'' in CNN denotes the mathematical function of convolution which is a special kind of linear operation wherein two functions are multiplied to produce a third function which expresses how the shape of one function is modified by the other. In simple terms, two images which can be represented as matrices are multiplied to give an output that is used to extract features from the image. Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernels), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1. The below figure is a complete flow of CNN to process an input image and classifies the objects based on values.

**6.1 Basic Architecture**

There are two main parts to a CNN architecture

* A convolution tool that separates and identifies the various features of the image for analysis in a process called as Feature Extraction
* A fully connected layer that utilizes the output from the convolution process and predicts the class of the image based on the features extracted in previous stages.

**6.1 CNN Layers:** The multiple occurring of these layers shows how deep our network is, and this formation is known as the deep neural network.

* Input: raw pixel values are provided as input.
* Convolutional layer: Input layers translate the results of the neuron layer. There is a need to specify the filter to be used. Each filter can only be a 5\*5 window that slides over input data and gets pixels with maximum intensities.
* Rectified linear unit [ReLU] layer: provided activation function on the data taken as an image. In the case of back propagation, ReLU function is used which prevents the values

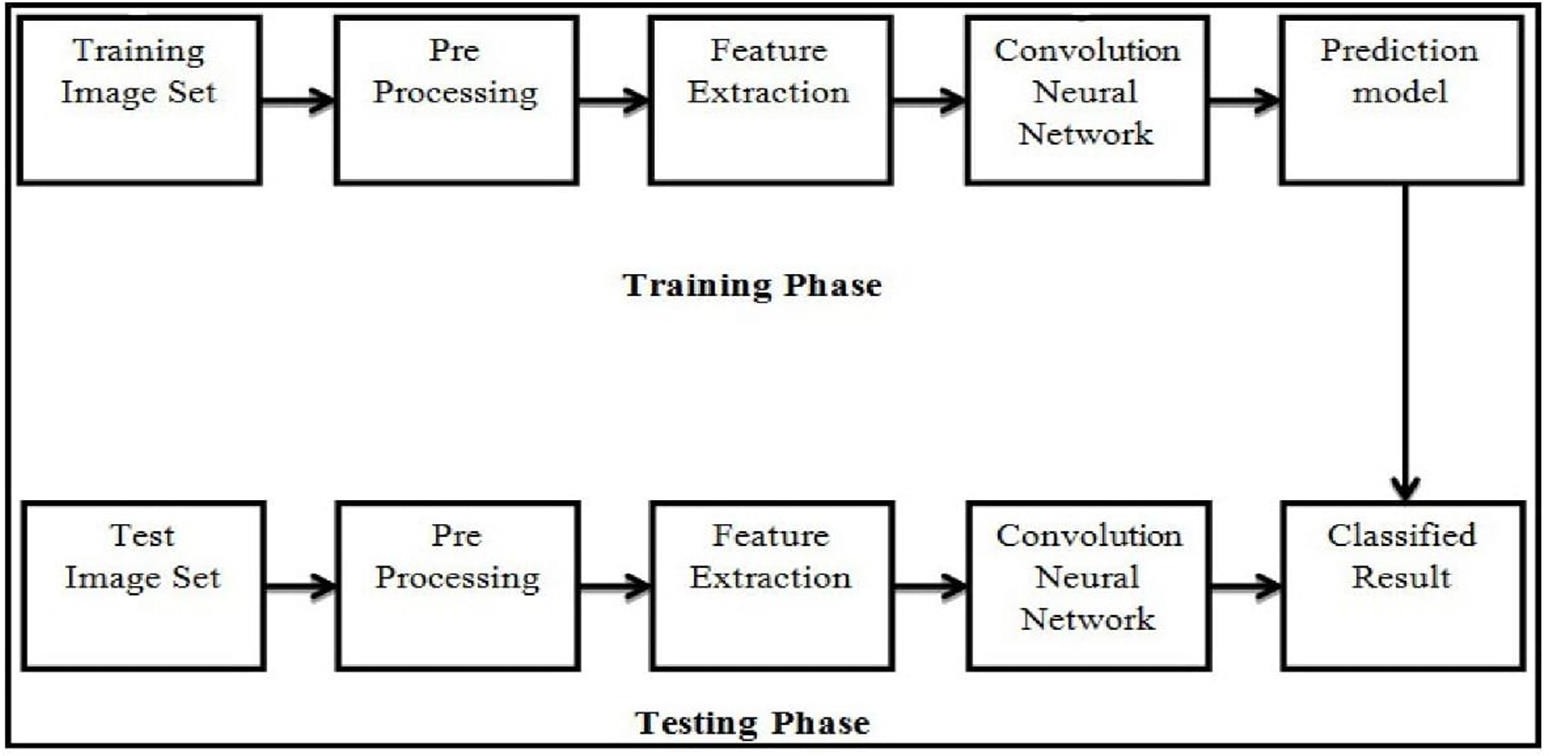
of pixels from changing.

* Pooling layer: Performs a down-sampling operation in volume along the dimensions (width, height).

## SYSTEM DESIGN

## SYSTEM ARCHITECTURE

The Architecture of the proposed system is the design diagram which depicts the scope of the project with the whole system design. In the architecture diagram, it highlights the modules with its various functions as a process. It aims to convey the internal design of the proposed system as follows.



The reason behind this document is to look into the design possibilities of the proposed system, such as architecture design, block diagram, sequence diagram, data flow diagram and user interface design of the system in order to define the steps such as pre-processing, feature extraction,segmentation, classification and recognition of digits.

Architecture of the Proposed System

The above Figure 1 illustrates the architecture diagram of the proposed system. The proposed model contains the four stages in order to classify and detect the digits:

1. **Pre-processing**
2. **Segmentation**
3. **Feature Extraction**
4. **Classification and Recognition**

### INPUT UNIT

In the input unit, the Facial images for Face Recognition and Video frames for person

detection are captured from camera input devices respectively.

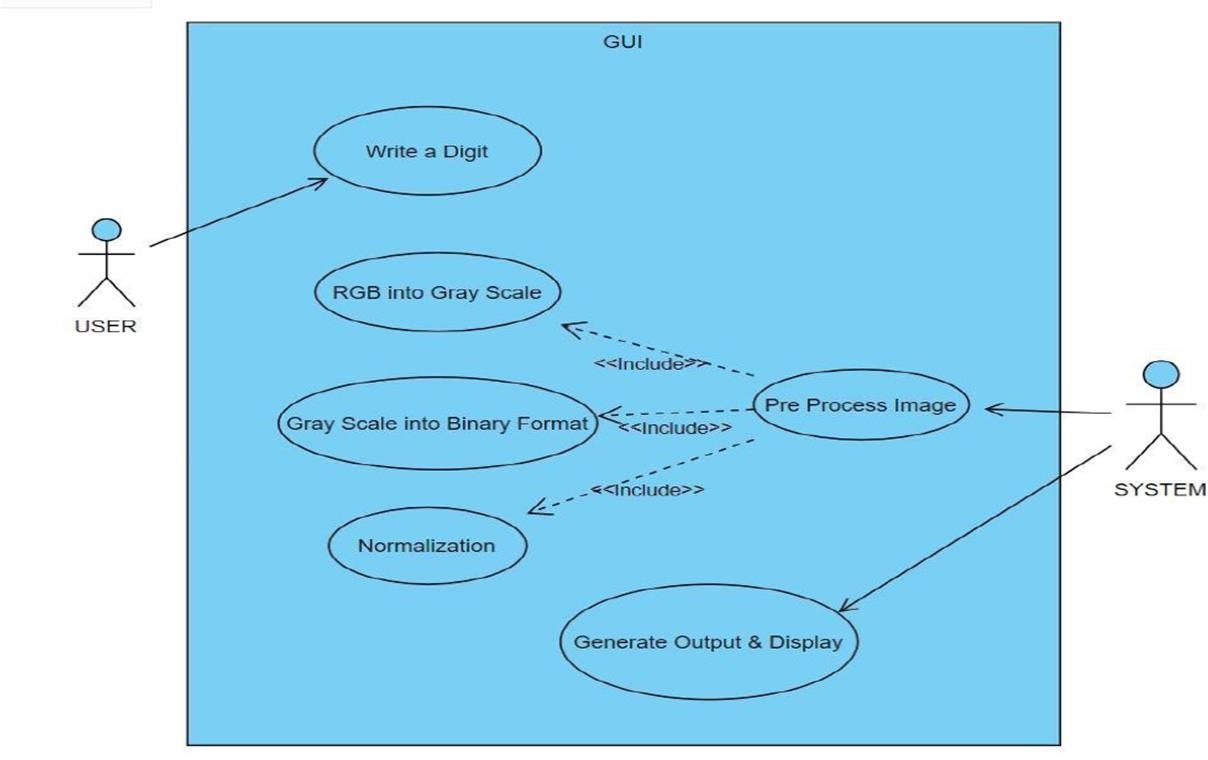
### PROCESSING UNIT

The data which is collected from Input unit that is captured Image and Video frames input is fed into the processing unit in which the processing or calculations are performed on the proposed person detection and door lock system module, Here the processing unit is nothing but a Raspberry Pi board, along with code scripts of the implemented modules.

### UML DIAGRAMS

* + 1. **USE-CASE DIAGRAM**

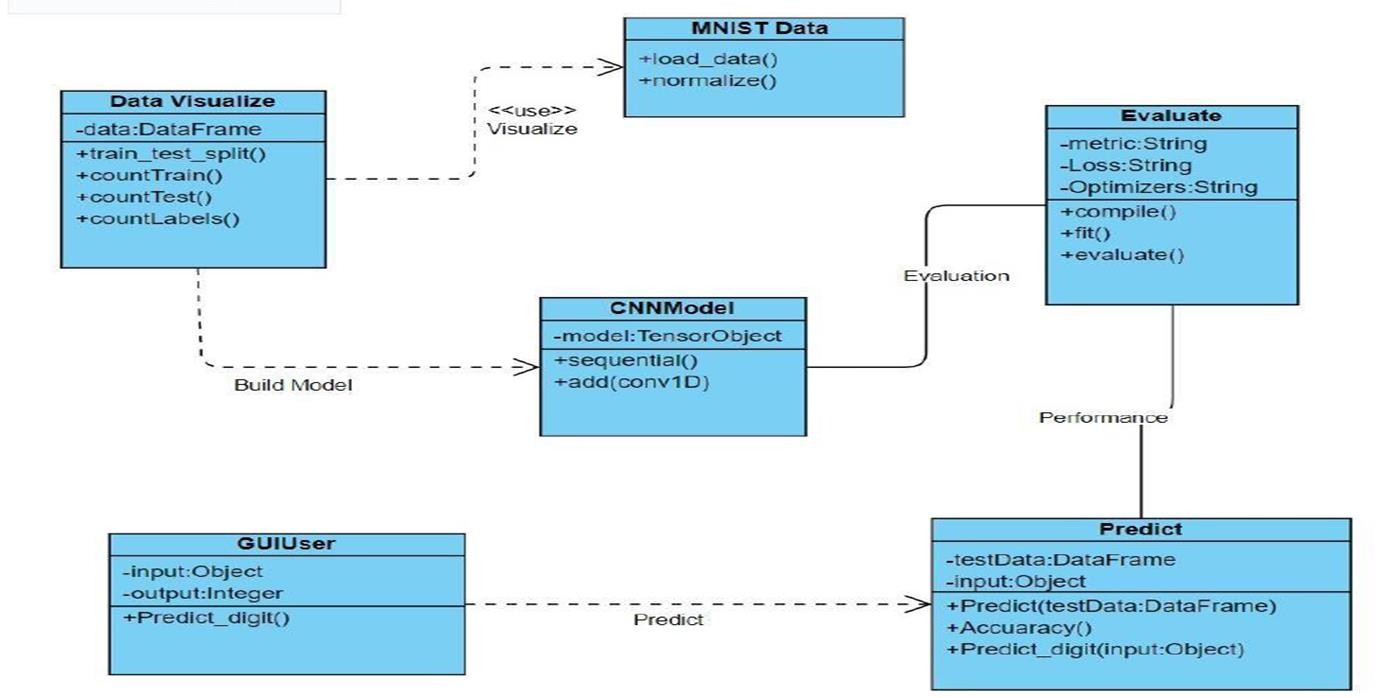
A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use case in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well**.**



### CLASS DIAGRAM

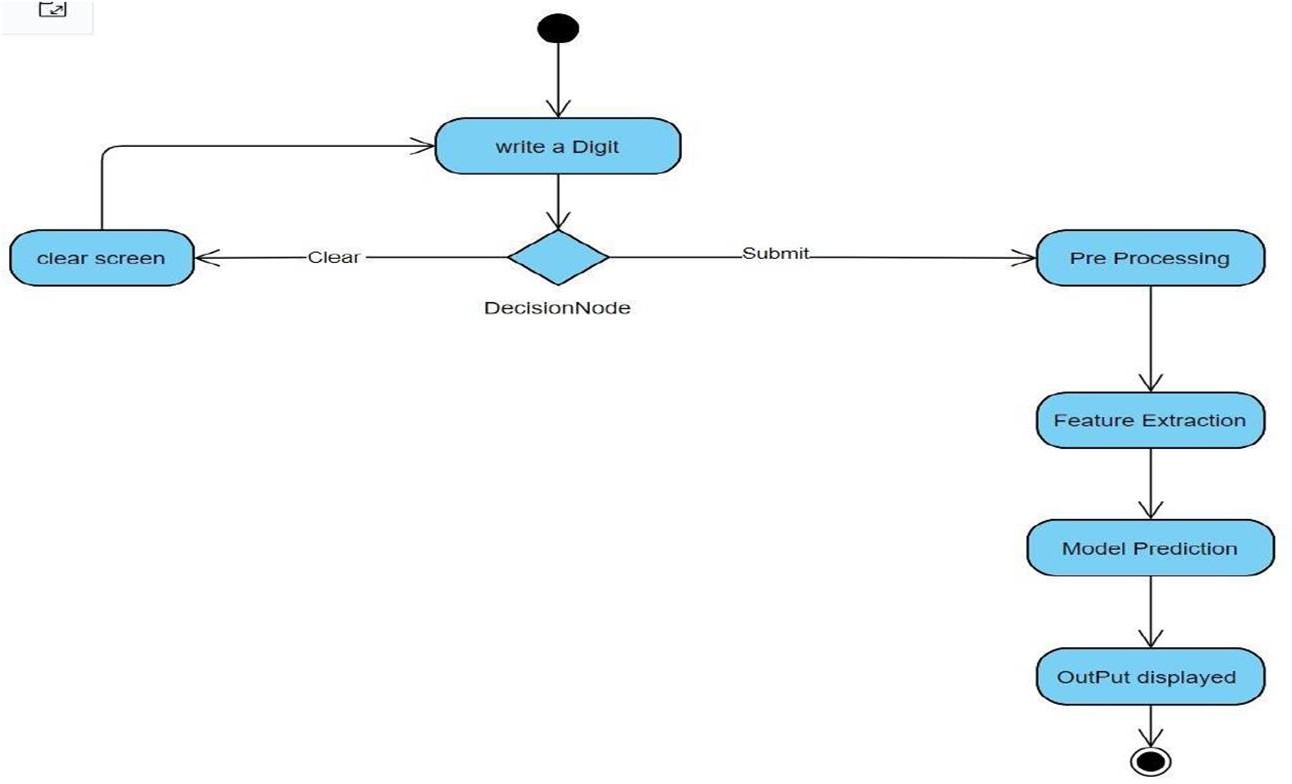
Class diagram in the Unified Modeling Language(UML) is a type of static structure diagram that describes the structure of a system by showing the system’s classes, their attributes,

operations (or methods), and the relationships among objects.



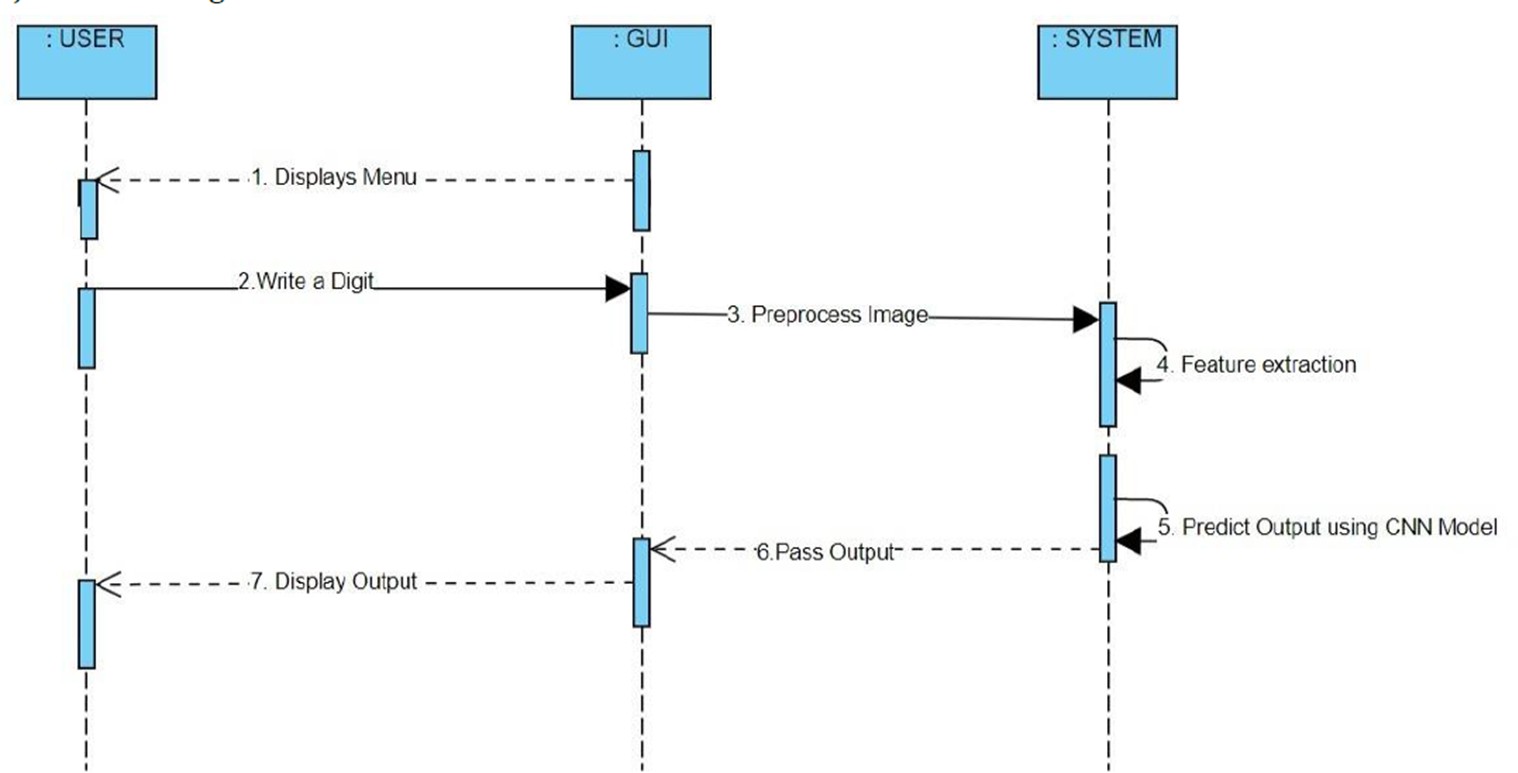
**4.2.4 ACTIVITY DIAGRAM**

An activity diagram is essentially a flowchart that depicts the movement of information from one action to the next .



### 4.2.3 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.



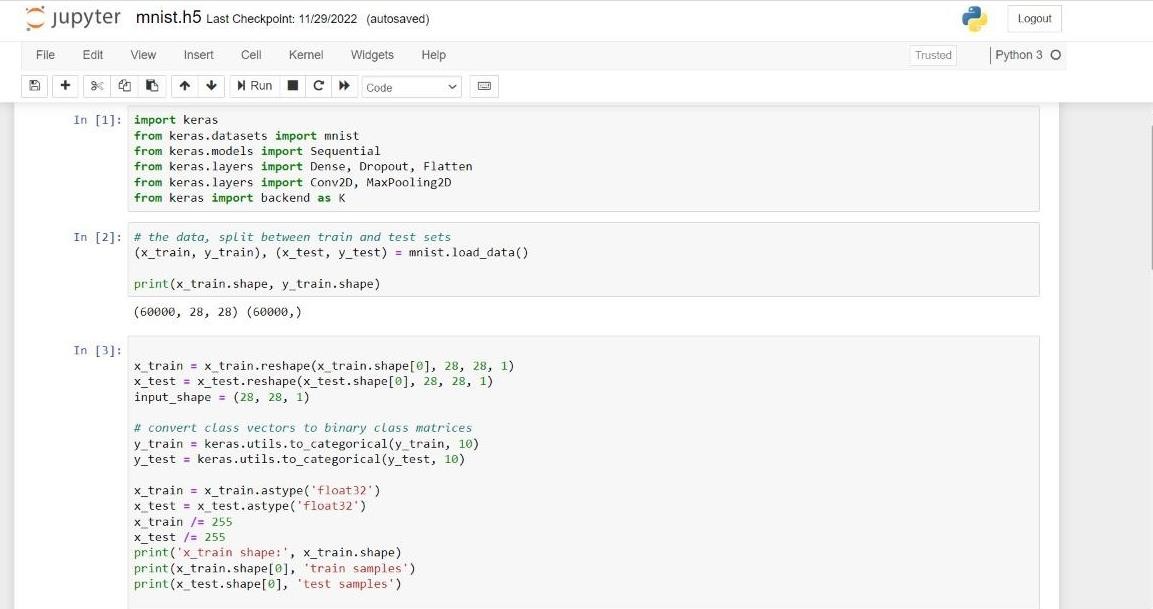
## IMPLEMENTATION

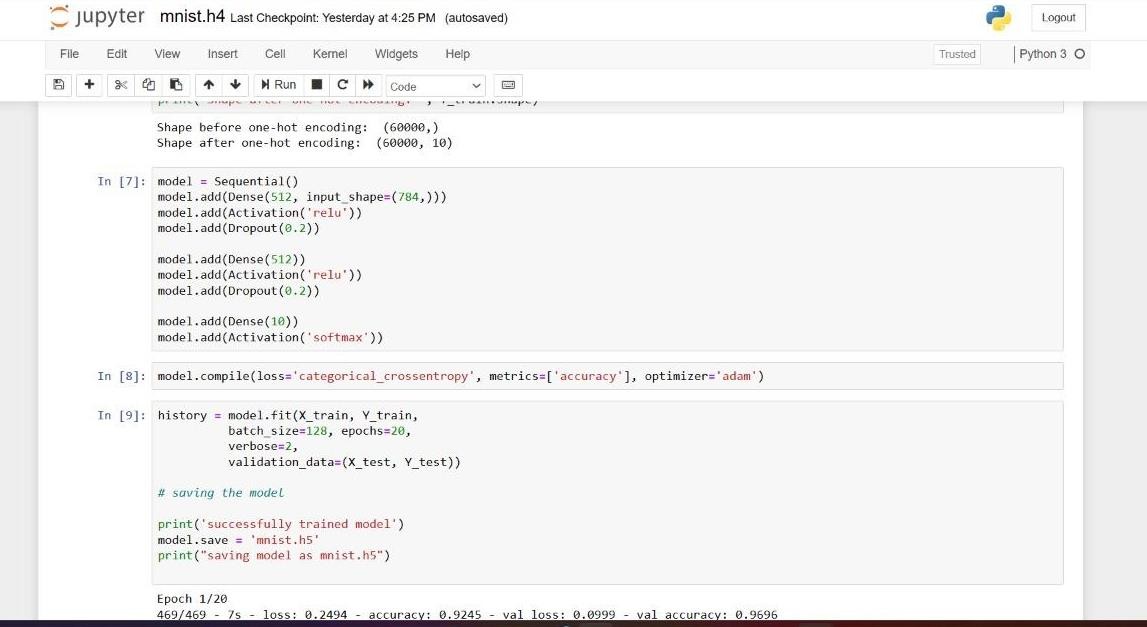
### CODING

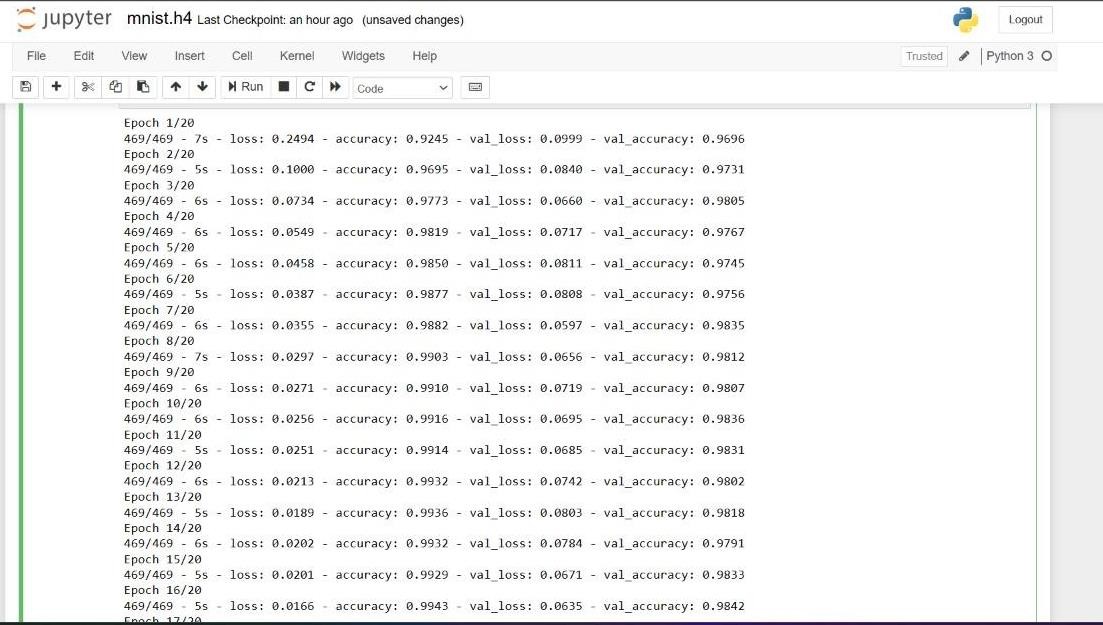
Import the libraries and load the dataset:

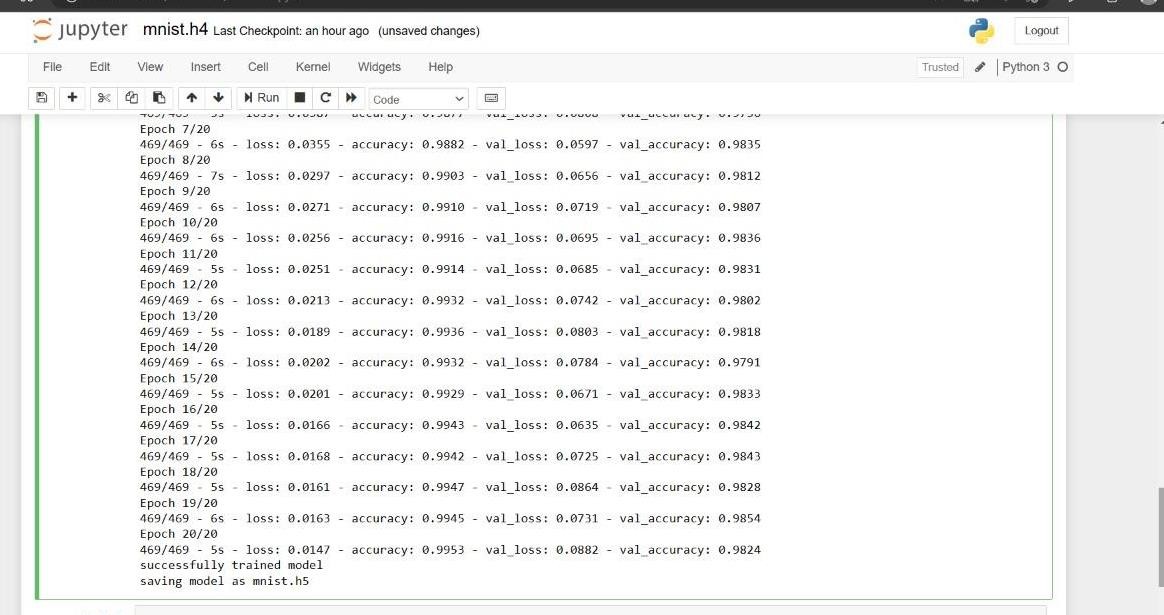
First, we are going to import all the modules that we are going to need for training our model. The Keras, library already contains some datasets and MNIST is one of them. So we can easily import the dataset and start working with it.

The mnist.load\_data() method returns us the training data, its labels and also the testing data and its labels.









### CODE:GRAPHICAL USER INTERFACE

import keras

from keras.datasets import mnist from keras.models import Sequential

from keras.layers import Dense, Dropout, Flatten from keras.layers import Conv2D, MaxPooling2D from keras import backend as K

# the data, split between train and test sets

(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data() print(x\_train.shape, y\_train.shape)

x\_train = x\_train.reshape(x\_train.shape[0], 28, 28, 1)

x\_test = x\_test.reshape(x\_test.shape[0], 28, 28, 1)

input\_shape = (28, 28, 1)

# convert class vectors to binary class matrices y\_train = keras.utils.to\_categorical(y\_train, 10) y\_test = keras.utils.to\_categorical(y\_test, 10) x\_train = x\_train.astype('float32')

x\_test = x\_test.astype('float32') x\_train /= 255

x\_test /= 255

print('x\_train shape:', x\_train.shape) print(x\_train.shape[0], 'train samples')

print(x\_test.shape[0], 'test samples') batch\_size = 128

num\_classes = 10

epochs = 10

model = Sequential()

model.add(Conv2D(32, kernel\_size=(5, 5),activation='relu',input\_shape=input\_shape)) model.add(MaxPooling2D(pool\_size=(2, 2)))

model.add(Conv2D(64, (3, 3), activation='relu'))

model.add(MaxPooling2D(pool\_size=(2, 2))) model.add(Flatten())

model.add(Dense(128, activation='relu')) model.add(Dropout(0.3)) model.add(Dense(64, activation='relu')) model.add(Dropout(0.5))

model.add(Dense(num\_classes, activation='softmax')) model.compile(loss=keras.losses.categorical\_crossentropy,optimizer=keras.optimizers.Adadelt a(),metrics=['accuracy'])

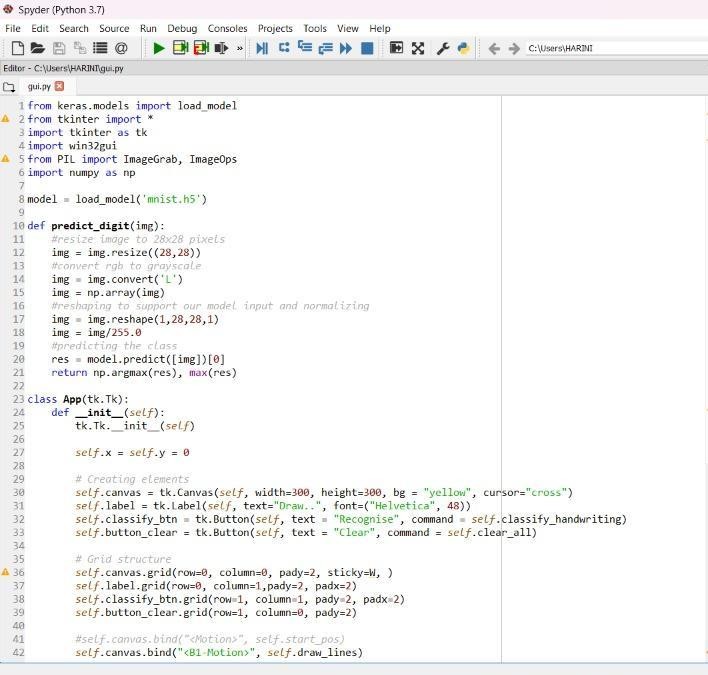
hist = model.fit(x\_train, y\_train,batch\_size=batch\_size,epochs=epochs,verbose=1,validation\_data=(x\_test, y\_test)) print("The model has successfully trained")

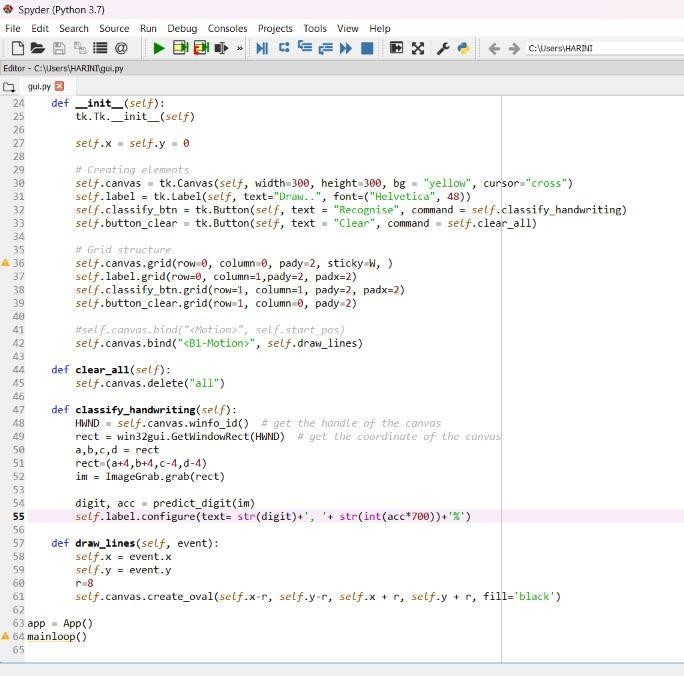
score = model.evaluate(x\_test, y\_test, verbose=0) print('Test loss:', score[0])

print('Test accuracy:', score[1]) model.save('mnist.h5')

print("Saving the model as mnist.h5")

## GRAPHICAL USER INTERFACE CODE





### CODE :

from keras.models import load\_model from tkinter import \*

import tkinter as tk import win32gui

from PIL import ImageGrab, ImageOps import numpy as np

model = load\_model('mnist.h5') def predict\_digit(img):

#resize image to 28x28 pixels img = img.resize((28,28)) #convert rgb to grayscale

img = img.convert('L') img = np.array(img)

#reshaping to support our model input and normalizing img = img.reshape(1,28,28,1)

img = img/255.0 #predicting the class

res = model.predict([img])[0] return np.argmax(res), max(res)

class App(tk.Tk): def init (self):

tk.Tk. init (self) self.x = self.y = 0

# Creating elements

self.canvas = tk.Canvas(self, width=300, height=300, bg = "white", cursor="cross") self.label = tk.Label(self, text="Draw..", font=("Helvetica", 48))

self.classify\_btn = tk.Button(self, text = "Recognise", command = self.classify\_handwriting)

self.button\_clear = tk.Button(self, text = "Clear", command = self.clear\_all)

# Grid structure

self.canvas.grid(row=0, column=0, pady=2, sticky=W, )

self.label.grid(row=0, column=1,pady=2, padx=2) self.classify\_btn.grid(row=1, column=1, pady=2, padx=2) self.button\_clear.grid(row=1, column=0, pady=2) #self.canvas.bind("<Motion>", self.start\_pos) self.canvas.bind("<B1-Motion>", self.draw\_lines)

def clear\_all(self): self.canvas.delete("all")

def classify\_handwriting(self):

HWND = self.canvas.winfo\_id() # get the handle of the canvas

rect = win32gui.GetWindowRect(HWND) # get the coordinate of the canvas a,b,c,d = rect

rect=(a+4,b+4,c-4,d-4)

im = ImageGrab.grab(rect)

digit, acc = predict\_digit(im)

self.label.configure(text= str(digit)+', '+ str(int(acc\*100))+'%')

def draw\_lines(self, event): self.x = event.x

self.y = event.y r=8

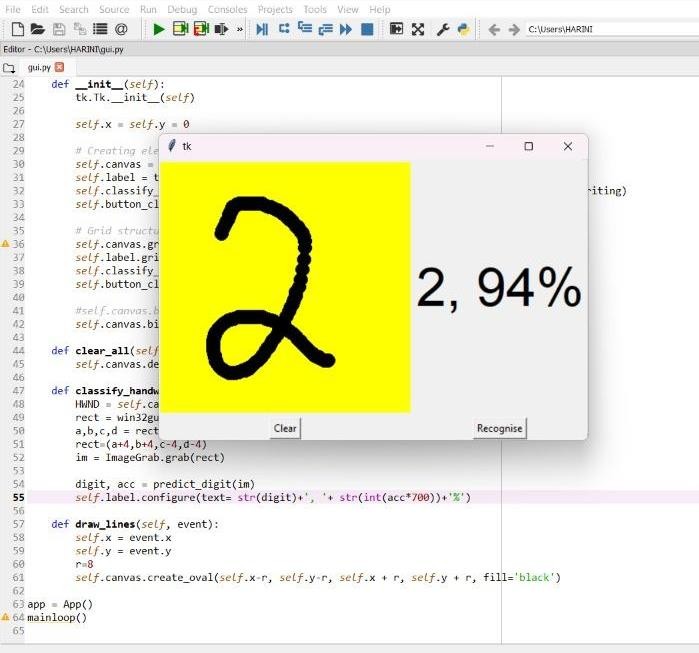
self.canvas.create\_oval(self.x-r, self.y-r, self.x + r, self.y + r, fill='black')

app = App() mainloop()

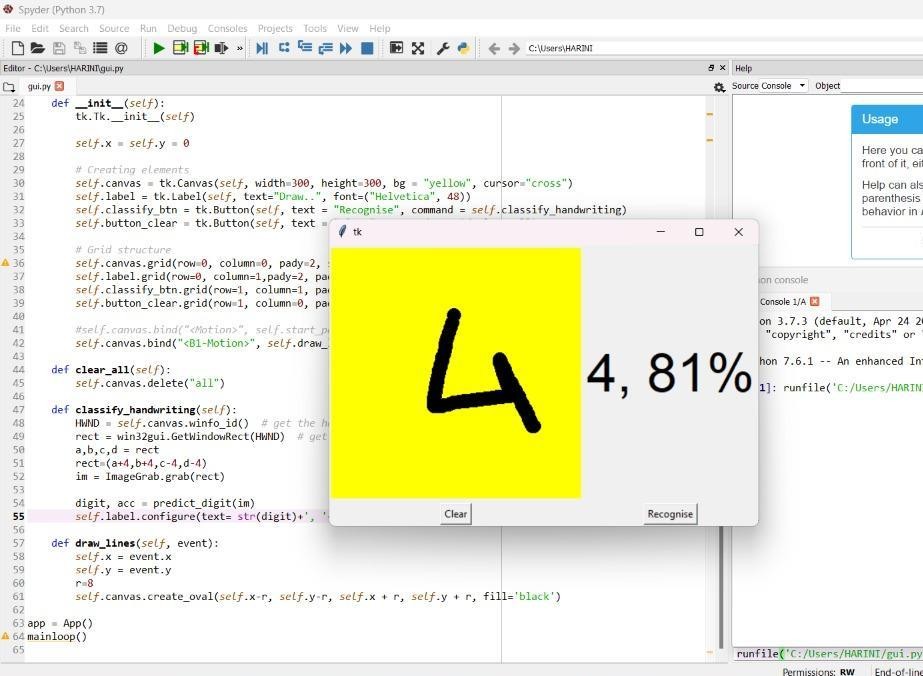
## OUTPUT SCREENS

### IMAGES OF RESULTS

**OUTPUT SCREEN 1:**



**OUTPUT SCREEN 2**



* 1. **CONCLUSION**

## CONCLUSION

In this project, the Handwritten Digit Recognition using Deep learning methods has been implemented. The most widely used Machine learning algorithm CNN has been trained and tested on the same data in order to acquire the comparison between the classifiers. Utilizing these deep learning techniques, a high amount of accuracy can be obtained. Compared to other research methods, this method focuses on which classifier works better by improving the accuracy of classification models by more than 99%. Using Keras as backend and TensorFlow as the software, a CNN model is able to give accuracy of about 98.72%. In this initial experiment, CNN gives an accuracy of 98.72%.

### FURTHER ENHANCEMENTS

The proposed system takes 28x28 pixel sized images as input. The same system with further modifications and improvements in the dataset and the model can be used to build Handwritten Character Recognition System which recognizes human handwritten characters and predicts the output.

* 1. **BOOK REFERENCE**

## BIBLIOGRAPHY

1. R. Alhajj and A. Elnagar, "Multiagents to separating handwritten connected digits," in IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol. 35, no. 5, pp. 593-602, Sept. 2005. https://doi.org/10.1109/TSMCA.2005.843389 https:/[/www.tensorflow.org/l](http://www.tensorflow.org/learn)e[arn](http://www.tensorflow.org/learn)
2. H. F. Schantz, optical character recognition: a history. Recognition technologies users association,manchestercenter,vt, 982.https://[www.geeksforgeeks.org/python-tkinter-](http://www.geeksforgeeks.org/python-tkinter-) tutorial/
3. Computer and computing history, internet, dreamers goldberg, emanuel. [online]. Goldberg's can be found at https://history-computer.com/internet/dreamers/goldberg.html.

### WEBSITE REFERENCES

1. https://[www.tensorflow.org/learn](http://www.tensorflow.org/learn)
2. https://[www.geeksforgeeks.org/python-tkinter-tutorial/](http://www.geeksforgeeks.org/python-tkinter-tutorial/)
3. ttps://medium.com/the-andela-way/applying-machine-learning-to-recognize handwrittencharacters-babcd4b8d705
4. https://nanonets.com/blog/handwritten-character-recognition/
5. https://[www.geeksforgeeks.org/python-tkinter-tutorial/](http://www.geeksforgeeks.org/python-tkinter-tutorial/)
6. ttps://medium.com/the-andela-way/applying-machine-learning-to-recognize handwrittencharacters-babcd4b8d705
7. https://nanonets.com/blog/handwritten-character-recognition/
8. <https://www.tensorflow.org/learn>
9. ttps://[www.kdnuggets.com/2018/11/top-python-deep-learning-libraries.html](http://www.kdnuggets.com/2018/11/top-python-deep-learning-libraries.html)
10. https://analyticsindiamag.com/7-types-classification-algorithms/
    1. **SOFTWARE USED**

## 9. APPENDICES

### SOFTWARE REQUIREMENTS

These are the software configurations used:

**Operating system:** windows 10.

**IDE:** Jupyter Notebook.

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

**Jupyter Notebook:** Jupyter is a free, open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text and multimedia resources in a single document. Computational notebooks have been around for decades, but Jupyter in particular has exploded in popularity over the past couple of years. This rapid uptake has been aided by an enthusiastic community of user–developers and a redesigned architecture that allows the notebook.

### METHODOLOGIES USED

* + 1. **Basic steps in constructing a Machine Learning model**

**2.2.1.1 - Data Collection**

* The quantity & quality of your data dictate how accurate our model is
* The outcome of this step is generally a representation of data (Guo simplifies to specifying a table) which we will use for training
* Using pre-collected data
  + - 1. **- Data Preparation**
* Wrangle data and prepare it for training
* Clean that which may require it (remove duplicates, correct errors, deal with missing values, normalization, data type conversions, etc.)
* Randomize data, which erases the effects of the particular order in which we collected and/or otherwise prepared our data
* Visualize data to help detect relevant relationships between variables or class imbalances (bias alert!), or perform other exploratory analysis
* Split into training and evaluation sets

### - Choose a Model

* Different algorithms are for different tasks; choose the right one

### 9.3.1.4- Train the Model

* The goal of training is to answer a question or make a prediction correctly as often as possible
* Linear regression example: algorithm would need to learn values for m (or W) and b (x is input, y is output)
* Each iteration of process is a training step

### 9.4.1.5 - Evaluate the Model

* Uses some metric or combination of metrics to "measure" objective performance of model
* Test the model against previously unseen data 17
* This unseen data is meant to be somewhat representative of model performance in the real world, but still helps tune the model (as opposed to test data, which does not)
* Good train/eval split? 80/20, 70/30, or similar, depending on domain, data availability, dataset particulars, etc.

### - Parameter Tuning

* This step refers to hyperparameter tuning, which is an "artform" as opposed to a science
* Tune model parameters for improved performance
* Simple model hyperparameters may include: number of training steps, learning rate, initialization values and distribution, etc.

### - Make Predictions

* Using further (test set) data which have, until this point, been withheld from the model (and for which class labels are known), are used to test the model; a better approximation of how the model will perform in the real world.

### Methodologies for Handwritten Digit Recognition System

We used MNIST as a primary dataset to train the model, and it consists of 70,000 handwritten raster images from 250 different sources out of which 60,000 are used for training, and the rest are used for training validation. Our proposed method mainly separated into stages, preprocessing, Model Construction, Training & Validation, Model Evaluation & Prediction. Since the loading dataset is necessary for any process, all the steps come after it.

### Import the libraries:

Libraries required are Keras, Tensor flow, Numpy, Pillow, Tkinkter.

1. **Keras:** Keras is a powerful and easy-to-use free open source Python library for developing and evaluating deep learning models. It wraps the efficient numerical computation libraries Theano and TensorFlow and allows you to define and train neural network models in just a few lines of code. It uses libraries such as Python, C#, C++ or standalone machine learning toolkits. Theano and TensorFlow are very powerful libraries but difficult to understand for creating neural networks. Keras is based on minimal structure that provides a clean and easy way to create deep learning models based on TensorFlow or Theano. Keras is designed to quickly define deep learning models. Well, Keras is an optimal choice for deep learning applications.
2. **TensorFlow:** TensorFlow is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning

models directly or by using wrapper libraries that simplify the process built on top of TensorFlow. TensorFlow tutorial is designed for both beginners and professionals. Our tutorial provides all the basic and advanced concept of machine learning and deep learning concept such as deep neural network, image processing and sentiment analysis.

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

1. **Numpy:** NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, Fourier transform, and matrices. Numpy which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. This tutorial explains the basics of NumPy such as its architecture and environment. It also discusses the various array functions, types of indexing, etc. It is an open source project and you can use it freely. NumPy stands for Numerical Python. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important.
2. **Pillow:** Pillow is a free and open source library for the Python programming language that allows you to easily create &s manipulate digital images. Pillow is built on top of PIL (Python Image Library). PIL is one of the important modules for image processing in Python. However, the PIL module is not supported since 2011 and doesn’t support python

3. Pillow module gives more functionalities, runs on all major operating system and support for python. It supports wide variety of images such as “jpeg”, “png”, “bmp”, “gif”, “ppm”, “tiff”. You can do almost anything on digital images using pillow module. Apart from basic image processing functionality, including point operations, filtering images

using built-in convolution kernels, and color space conversions.

**5. Tkinkter:** Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. We need to import all the modules that we are going to need for training our model. The Keras library already contains some datasets and MNIST is one of them. So we can easily import the dataset through Keras. The mnist.load\_data() method returns the training data, its labels along with the testing data and its labels.

### TESTING METHODS USED

This is probably one of the most popular datasets among machine learning and deep learning enthusiasts. The MNIST dataset contains 60,000 training images of handwritten digits from zero to nine and 10,000 images for testing. So, the MNIST dataset has 10 different classes.

The handwritten digits images are represented as a 28×28 matrix where each cell contains grayscale pixel value.

## 10. PLAGIARISM REPORT

