

PROFESSIONAL TRAINING REPORT

at

Sathyabama Institute of Science and Technology (DEEMED TO BE UNIVERSITY)

Submitted in partial fulfilment of the requirements for the award of Bachelor of
Technology Degree in

INFORMATION TECHNOLOGY

By

DIVYA S

(REG NO: 37120025)



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,

CHENNAI –600119, TAMILNADU.

DECEMBER 2020



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" Grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600 119.

Phone: 044 - 2450 3150 / 51 / 52 / 54 / 55 Fax: 044 - 2450 2344

www.sathyabama.ac.in



SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this Professional Training Report is the bonafide work of **Divya S(Reg.No.37120025)** who underwent the professional training in **"HANDWRITTEN DIGIT RECOGNITION USING CNN"** under our supervision from June 2020 to August 2020.

Internal Guide

Dr. Y.BEVISH JINILA. M.E., Ph.D.,

Head of the Department

Dr .R.SUBHASHINI. M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **DIVYA S(37120025)** hereby declare that the Project Report entitled **“HANDWRITTEN DIGIT RECOGNIZER USING CNN”** done by me under the guidance of **DR.Y.BEVISH JINILA M.E., Ph.D** is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology.

DATE:

PLACE: Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D, Dean**, School of Computing , **Dr.SUBHASHINI M.E., Ph.D.** , Head of the Department of Information and Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **DR.Y.BEVISH JINILA M.E., Ph.D** .,for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

ONLINE COURSE CERTIFICATION



ABSTRACT

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 a very similar to this. It uses the special type of neural network that is Convolutional Neural Network(CNN), And the Handwritten digit recognition is the ability of computers to recognize human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different flavors and the different users have their own handwriting styles where here the main challenge fails to let computer system understand these different handwriting styles, recognize them as standard writing. The handwritten digit recognition is the solution to this problem which uses the image of a digit and recognizes the digit present in the image. Using MNIST dataset we can able to implement a handwritten digit recognition app. And the main aim to build a GUI with Keras library and the Tkinter library. This Working Prototype System can detect handwritten digits from a scanned image of an input form by using Neural Network Technique and it is very fast and Effective as compared to old fashioned image pixel comparison methodology. It is already widely used in the automatic processing of bank cheques, Postal addresses, Signature in mobile phones etc.

TABLE OF CONTENTS

CHAPTER NO	TITLE NAME	PAGE NO
	ABSTRACT	6
	LIST OF FIGURES	9
	LIST OF ABBREVIATION	10
1	INTRODUCTION	
	1.1 GENERAL	11
	1.2 OUTLINE OF THE PROJECT	13
2	AIM AND SCOPE OF THE PRESENT INVESTIGATION	
	2.1 AIM	14
	2.2 SCOPE	14
3	EXPERIMENTAL OR MATERIALS AND METHODS AND ALGORITHMS USED	
	3.2 ALGORITHM	15
	3.3 SUMMARY	16
4	RESULT AND DISCUSSION, PERFORMANCE ANALYSIS	
	4.1 REQUIREMENTS	20
	4.3 RESULT AND SCREENSHOTS	21

5	SUMMARY AND CONCLUSION	
	5.1 CONCLUSION	24
	5.2 FUTURE WORK	24
	APPENDIX	
	A.SOURCE CODE	26
	REFERENCES	25

LIST OF FIGURES

FIGURE NO	DIAGRAM NAME	PAGE NO
1.1.1	DIGIT PREDICTION	12
1.1.2	IMAGES OF DIGITS	12
3.9.1	GENERAL ARCHITECTURE	17
3.9.2	ERROR RATE AND ACCURACY	18
4.2.1	LIST OF PACKAGES	20
4.2.2	GUI SCREEN	21
4.3.3	DIGIT ZERO	22
4.2.4	DIGIT TWO	22
4.2.5	DIGIT THREE	23
4.2.6	DIGIT FIVE	23
4.2.7	DIGIT SEVEN	23

LIST OF ABBREVIATIONS

CNN	- Convolutional Neural Network
MNIST	- Modified National Institute and Technology database
GUI	- Graphical User Interface

CHAPTER 1

INTRODUCTION

1.1 GENERAL

The MNIST problem is a dataset developed by Yann LeCun, Corinna Cortes and Christopher Burges for evaluating machine learning models on the handwritten digit classification problem. The dataset was constructed from a number of scanned document dataset available from the National Institute of Standards and Technology (NIST). This is where the name for the dataset comes from, as the Modified NIST or MNIST dataset. Images of digits were taken from a variety of scanned documents, normalized in size and centered. This makes it an excellent dataset for evaluating models, allowing the developer to focus on the machine learning with very little data cleaning or preparation required. Each image is a 28 by 28 pixel square(784 pixels total). A standard split of the dataset is used to evaluate and compare models, where 60,000 images are used to train a model and a separate set of 10,000 images are used to test it. It is a digit recognition task. As such there are 10 digits (0 to 9) or 10 classes to predict. Results are reported using prediction error, which is nothing more than the inverted classification accuracy. Excellent results achieve a prediction error of less than 1%. State-of-the-art prediction error of approximately 0.2% can be achieved with large Convolutional Neural Networks. And It is a hard task for the machine because handwritten digits are not perfect and can be made with many different flavors. The handwritten digit recognition is the solution to this problem which uses the image of a digit and recognizes the digit present in the image.

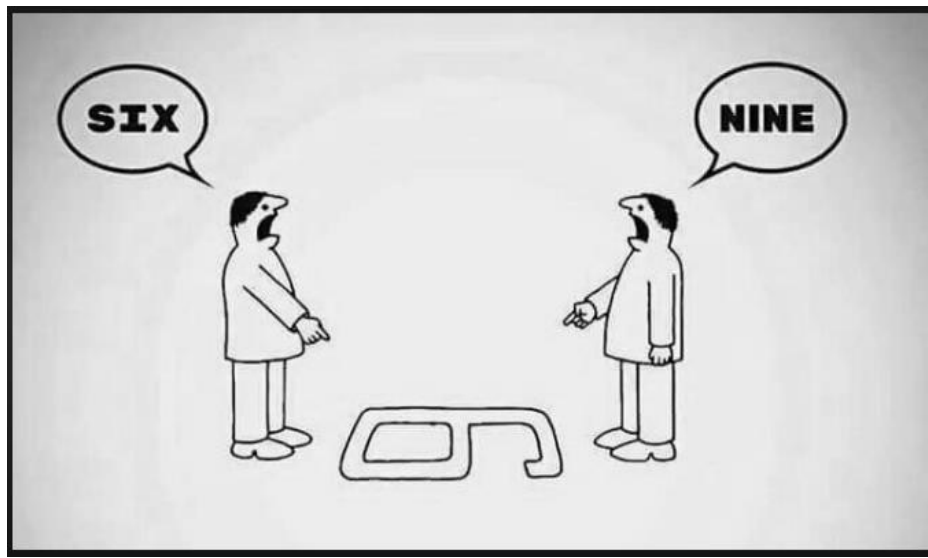


Fig 1.1.1 DIGIT PREDICTION

(This picture shows that there's always a difference when predicting the digit)

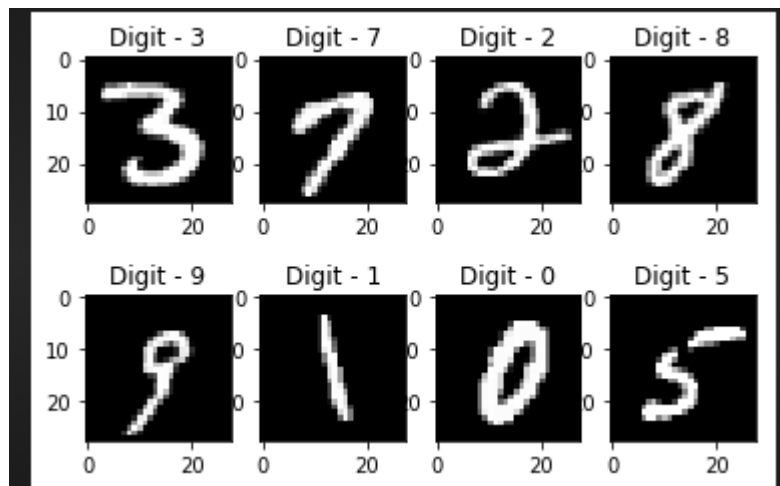


Fig 1.1.2 IMAGES OF DIGITS

1.2 OUTLINE OF 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. Machine Learning provides various methods through which human efforts can be reduced in recognizing the manually written digits. Deep Learning is a machine learning method that trains computers to do what easily falls into place for people: learning through examples. With the utilization of deep learning methods, human attempts can be diminished in perceiving, learning, recognizing and in a lot more regions. Using deep learning, the computer learns to carry out classification works from pictures or contents from any document. Deep Learning models can accomplish state-of-art accuracy, beyond the human level performance. The digit recognition model uses large datasets in order to recognize digits from distinctive sources. 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).

CHAPTER 2

AIM AND SCOPE

2.1 AIM

To implement an handwritten digit recognition app using the MNIST dataset. And using a special type of deep neural network that is Convolutional neural network. Then building a GUI to recognize the digit and display the output through it. convolutional neural networks (CNNs) are very effective in perceiving the structure of handwritten characters/words in ways that help in automatic extraction of distinct features and make CNN the most suitable approach for solving handwriting recognition problems. Our aim in the proposed work is to explore the various design options like number of layers, stride size, receptive field, kernel size, padding and dilution for CNN-based handwritten digit recognition

2.2 SCOPE

Handwritten digit recognition becomes vital scope and it is appealing many researchers because of it's using in variety of machine learning and computer vision applications. Because Handwritten digits are not perfect and can be made with many different flavors. The work can be extended to increase the results by using or adding some more relevant features. A lot of efforts have been made to get higher accuracy and there is tremendous scope of improving recognition accuracy by developing new feature extraction techniques or modifying the existing feature extraction techniques.

CHAPTER 3

ALGORITHMS AND METHODS

3.1 ALGORITHM

- The CNN algorithm is used for creating an handwritten digit recognizer. The name “convolutional neural network” indicates that the network employs a mathematical operation called Convolution.
- Convolution is a specialized kind of linear operation. Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers.

3.2 PACKAGE INSTALLATION

Installing the necessary libraries for this project using this command:

- Pip install numpy, tensorflow, keras, pillow

3.3 THE MNIST DATASET

- 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.

3.4 IMPORT THE LIBRARIES AND DATASET

- The first step is 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, The **mnist.load_data()** method returns us the training data, its labels and also the testing data and its labels.

3.5 PREPROCESS THE DATA

- The image data cannot be fed directly into the model so we need to perform some operations and process the data to make it ready for our neural network.
- The dimension of the training data is (60000,28,28). The CNN model will require one more dimension so we reshape the matrix to shape (60000,28,28,1).

3.6 CREAT THE MODEL

- Now we will create our **CNN** model in Python data science project.
- A CNN model generally consists of convolutional and pooling layers.
- It works better for data that are represented as grid structures, this is the reason why CNN works well for image classification problems.
- The dropout layer is used to deactivate some of the neurons and while training, it reduces over fitting of the model.
- We will then compile the model with the Adadelta optimizer.

3.7 TRAIN THE MODEL

- The **model.fit()** function of Keras will start the training of the model.
- It takes the training data, validation data, epochs, and batch size.

- It takes some time to train the model. After training, we save the weights and model definition in the 'mnist.h5' file.

3.8 EVALUATE THE MODEL

- We have 10,000 images in our dataset which will be used to evaluate how good our model works.
- The MNIST dataset is well balanced so we can get around 99% accuracy.

3.9 CREATE THE GUI TO PREDICT DIGITS

- Now for the GUI, we have created a new file in which we build an interactive window to draw digits on canvas and with a button, we can recognize the digit.
- The Tkinter library comes in the Python standard library, We have created a function **predict_digit()** that takes the image as input and then uses the trained model to predict the digit.
- Then we create the App class which is responsible for building the GUI for our app. We create a canvas where we can draw by capturing the mouse event and with a button, we trigger the predict_digit() function and display the results.

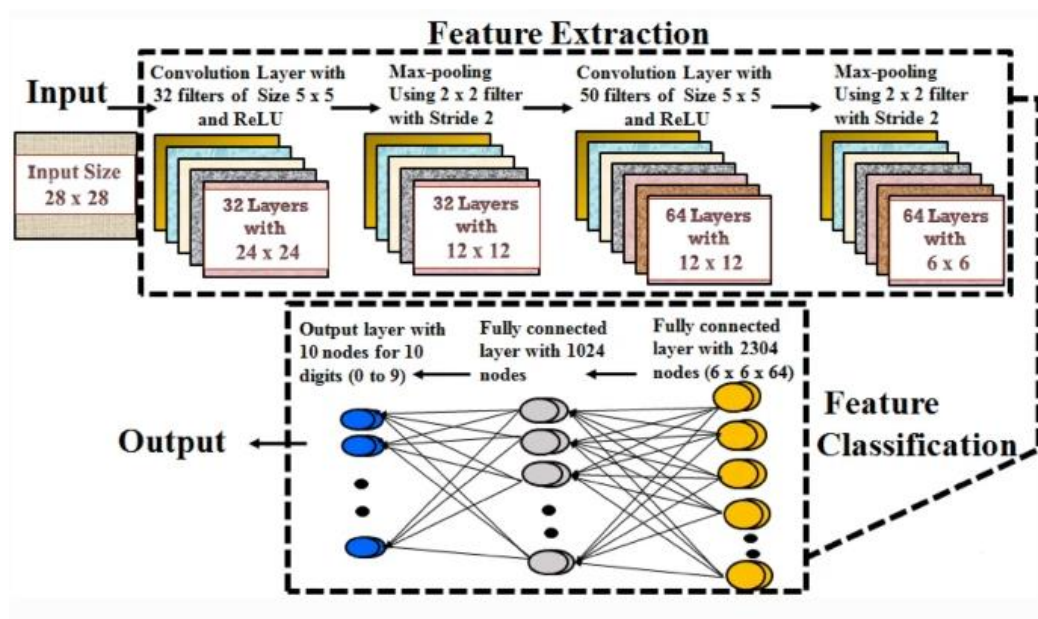


Fig 3.9.1 GENERAL ARCHITECTURE

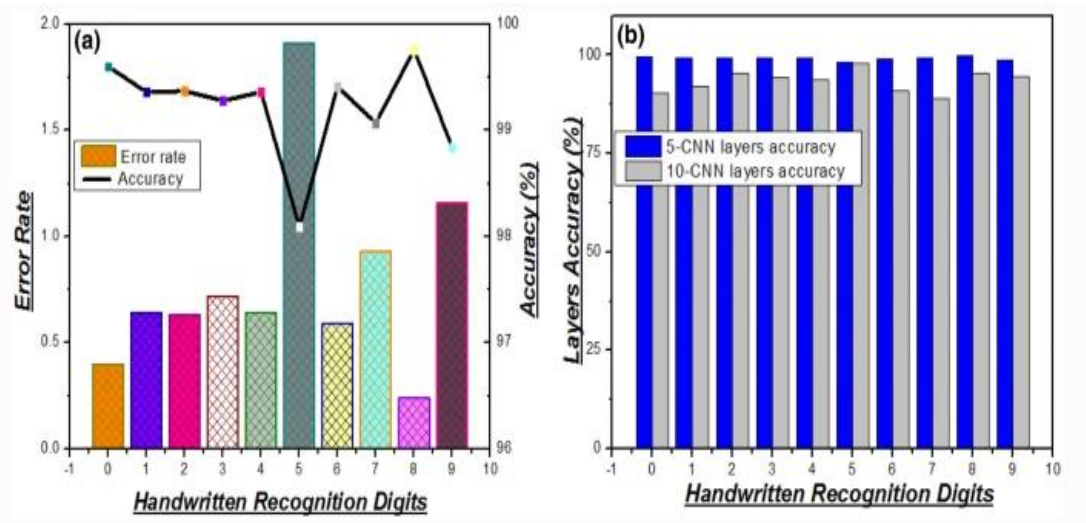


Fig 3.9.2 ERROR RATE AND ACCURACY

The above figure shows error rates and accuracy among different written digits. The minimum error rate was found with digit 8 while maximum with digit 5. However, the result of accuracy is in reciprocal to error rate. The highest accuracy (99.76%) was found with digit 8 and minimum with digit 5 (98.09%). The reason behind this variability is that we employed different writing styles and digit sizes while taking testing dataset. Consequently, proposed schemes attained the average accuracy of 99.21%. To increase the accuracy rate and decrease the error, we recommend a larger size dataset.

3.4 SUMMARY

we have successfully built a Python deep learning project on handwritten digit recognition app. We have built and trained the Convolutional neural network which is very effective for image classification purposes. then, we have build the GUI where we draw a digit on the canvas then we've classified the digit and shown the results.

CHAPTER 4

SYSTEM IMPLEMENTATION

4.1 REQUIREMENTS

I.HARDWARE REQUIREMENTS:

- Processor - Intel®Core™i5-4310M CPU @2.70 GHz
- RAM - 4GB
- Hard Disk - 400 GB

II. SOFTWARE REQUIRMENTS

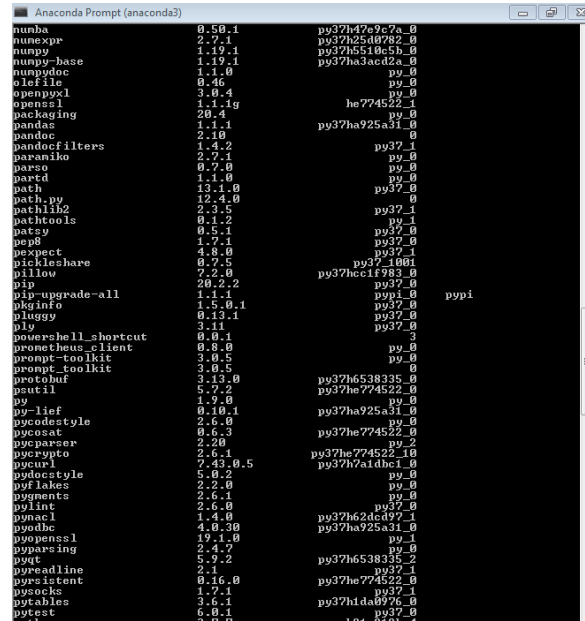
Software Requirements deal with defining software resource requirements and pre-requisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or pre-requisites are generally not included in the software installation package and need to be installed separately before the software is installed.

OPERATING SYSTEM: - Windows 7

ENVIRONMENT: - Anaconda Command Prompt

LANGUAGE: - Python

4.2 RESULTS AND SCREENSHOTS



```

Anaconda Prompt (anaconda3)
numba 0.50.1 py37h47e9c7a_0
numexpr 2.7.1 py37h25d8782_0
numpy 1.19.1 py37h5516c5b_0
numpy-base 1.19.1 py37ha3acd2a_0
numpydoc 1.1.0 py_0
olefile 0.46 py_0
openpyxl 3.0.4 py_0
openssl 1.1.1g he774522_1
packaging 20.4 py_0
pandas 1.1.1 py37ha925a31_0
pandoc 2.10 py_0
pandocfilters 1.4.2 py37_1
paramiko 2.7.1 py_0
parso 0.7.0 py_0
partd 1.1.0 py_0
path 13.1.0 py37_0
path.py 2.3.5 py37_1
pathlib2 0.1.2 py_1
pathtools 0.5.1 py37_0
patsy 1.2.1 py37_0
pep8 4.8.0 py37_1
pickleshare 0.2.5 py37_1001
pillow 7.2.0 py37hccf983_0
pip 20.2.2 py37_0
pip-upgrade-all 1.1.1 py37_0
pkginfo 1.5.0.1 py37_0
pluggy 0.13.1 py37_0
ply 3.11 py37_0
powershell_shortcut 0.0.1 3
prometheus_client 0.8.0 py_0
prompt_toolkit 3.0.5 py_0
prompt_toolkit 3.0.5 py_0
protobuf 3.13.0 py37h6538335_0
psutil 5.2.2 py37he774522_0
py 1.9.0 py_0
py-lief 0.10.1 py37ha925a31_0
pycodestyle 2.6.0 py_0
pycosat 0.6.3 py37he774522_0
pyparser 2.20 py_2
pycvpyto 2.6.1 py37he774522_10
pycurl 7.43.0.5 py37h7a1dbc1_0
pydocstyle 5.0.2 py_0
pyflakes 2.2.0 py_0
pygments 2.6.1 py_0
pylint 2.6.0 py37_0
pynacl 1.4.0 py37h624c4d7_1
pyodbc 4.0.30 py37ha925a31_0
pyopenssl 19.1.0 py_1
pyparso 2.4.2 py_0
pyqt 5.7.2 py37h6538335_2
pyreadline 2.1 py37_1
pyrsistent 0.16.0 py37he774522_0
pysocks 1.7.1 py37_1
pytables 3.6.1 py37h1da0976_0
pytest 5.0.5 py37_0

```

Fig 4.2.1 LIST OF PACKAGES



Fig 4.2.2 GUI SCREEN

Displaying handwritten digits and it's accuracy on the GUI

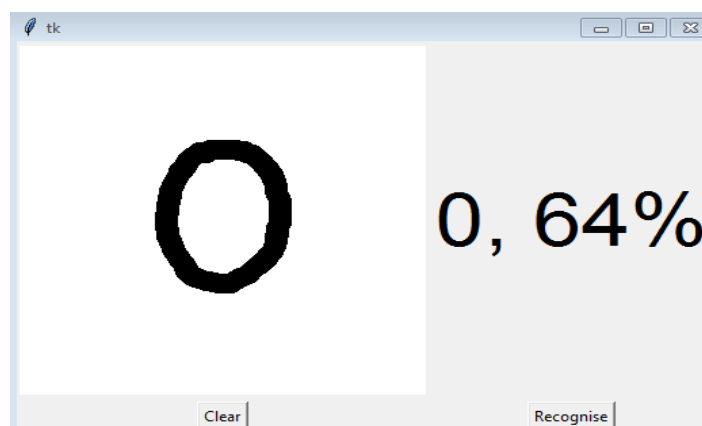


FIG 4.2.3 Digit Zero

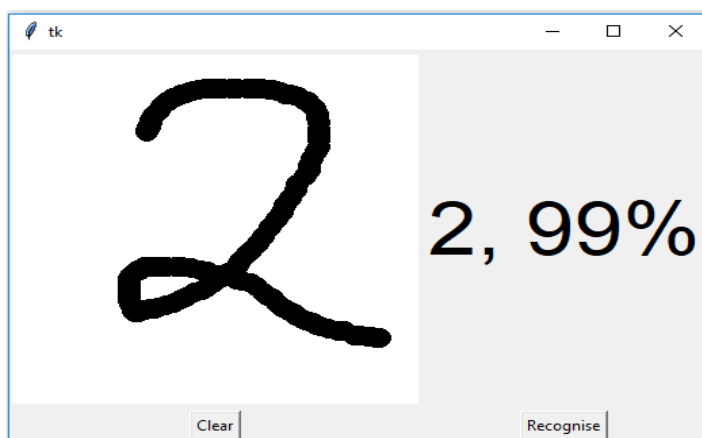


FIG 4.2.4 Digit Two

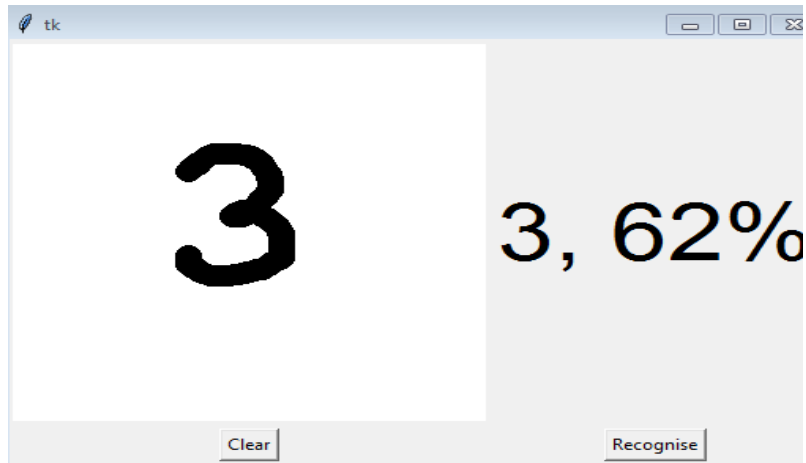


FIG 4.2.5 Digit Three

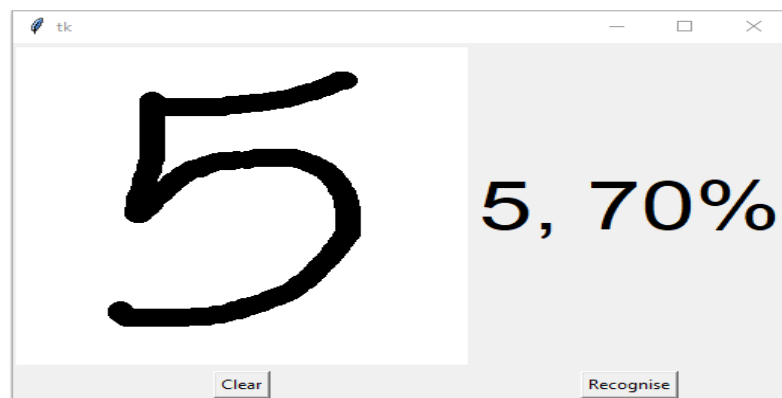


FIG 4.2.6 Digit Five



FIG 4.2.7 Digit Seven

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

As using machine learning algorithms are used like convolutional Neural networks along with different parameters and feature scaling vectors, we also saw the different comparison among the classifiers in terms of the most important feature of accuracy and timing. Accuracy can alter as it depends on the splitting of training and testing data, and this can further be improved if the number of training and testing data is provided. There is always a chance to improve accuracy if the size of data increases. Every classifier has its own accuracy and time consumption. We can conclude that the classifier can perform with better accuracy and less time and better results can be observed.

5.2 FUTURE WORK

Future studies might consider using the architecture of the convolution network which gave the best result on the MNIST database and the proposed recognition system is implemented on handwritten digits. Such more system can be designed for handwritten characters recognition, object recognition, image segmentation, handwriting recognition, text language recognition, and future studies also might consider on hardware implementation on online digit recognition system with more performance and efficiency with live results from live testing case scenarios.

REFERENCES

1. <https://data-flair.training/blogs/python-deep-learning-project-handwritten-digit-recognition/>
2. <https://nextjournal.com/gkoehler/digit-recognition-with-keras>
3. <https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-from-scratch-for-mnist-handwritten-digit-classification/>
4. <https://becominghuman.ai/handwritten-digit-prediction-using-cnn-part-1-142dbb756110>

APPENDIX

A. SOURCE CODE

```
//Importing libraries
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)

//Preprocessing the data
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, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)
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')

//Creating the model
batch_size = 128
num_classes = 10
epochs = 10
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=input_shape))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
```

```

model.add(Dense(256, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
model.compile(loss=keras.losses.categorical_crossentropy,optimizer=keras.optimizers
.Adadelta(),metrics=['accuracy'])

```

//Train the model

```

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")
model.save('mnist.h5')
print("Saving the model as mnist.h5")
//Evaluating the model
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])

```

//Creating GUI to predict digits

```

from keras.models import load_model
from tkinter import *
import tkinter as tk
import win32gui
from PIL import ImageGrab, Image
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="Thinking..", 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
        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()

```