(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Handwritten Digit Recognition

A Project Report of Capstone Project - 2

Submitted by

Kumar Abhishek

(1613101344 / 16SCSE101208)

in partial fulfilment for the award of the degree

of

Bachelor of Technology

IN

Computer Science and Engineering

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Under the supervision of

Mrs Sonia Kukreja

Assistant Professor

APRIL / MAY -2020

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report **"HANDWRITTEN DIGIT RECOGNITION"** is the bonafide work of **"KUMAR ABHISHEK(1613101344)"** who carried out the project work under my supervision.

SIGNATURE OF HEAD	**SIGNATURE OF SUPERVISOR**
Dr. MUNISH SHABARWAL,	**Mrs Sonia Kukreja,**
PhD (Management), PhD (CS)	**Assistant Professor**
Professor & Dean,	**School of Computer Science & Engineering**
School of Computer Science & Engineering	

ACKNOWLEDGEMENT:

This is an excellent opportunity to acknowledge and to thanks, all those persons without whose support and help this project would be impossible. We might prefer to add some heartfelt words for those who were a part of this project in numerous ways.

I would prefer to because of my project guide **Mrs Sonia Kukreja**, for her indefatigable guidance, valuable suggestion, moral support, constant encouragement, and contribution of your time for the successful completion of project work. I'm very grateful to her, for providing all the facilities needed during the project development. At the outset, I sincerely thank all faculty members of my institution **GALGOTIAS UNIVERSITY** for their extra effort to create our session online and inspire all ideas.

I would prefer to thank all those that helped me directly or indirectly. Last but not the least, I'd prefer to acknowledge the continuing support of my friends, whose patience and encouragement during these long days and nights are paramount in making this project a reality.

THANK YOU.

DECLARATION:

I hereby declare that this submission is my very own work which, to the simplest of my knowledge and belief, it contains no material previously published or written by another person nor material which to a considerable extent has been accepted for the award of the other degree or diploma of the university or other institute of upper learning, except where due acknowledgment has been made within the text.

I inform that every data used in this report if it's taken from any site is clearly referenced under the reference section.

SIGNATURE

Kumar Abhishek

16SCSE101208

Date: 18-may-2020

TABLE OF CONTENTS:

TITLE **PAGE NO.**

ABSTRACT

In recent times, with the increase of Artificial Neural Network (ANN), deep learning has brought a dramatic twist in the field of machine learning by making it more artificially intelligent. Deep learning is remarkably used in vast ranges of fields because of its diverse range of applications such as surveillance, health, medicine, sports, robotics, drones etc. The Handwritten Digit Recognition is one such ability of computers to recognize human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different flavours. The goal of this project is to provide the solution to this problem which uses the image of a digit and recognizes the digit present in the image by using the concept of Convolutional Neural Network. In this project, we train our model using Modified National Institute of Standards and Technology (MNIST) dataset. This dataset is trained using convolutional neural network algorithm with the help of Keras which is python library for extensive computation of neural nodes supported by Tensorflow framework at backend. After training, iterative testing with more accurate model is formed. With this formed model, we will be able to predict the handwritten digits in an image. Once model gets the desired testing results, a GUI is developed for users where they can make their inputs either by drawing the digits by themselves or inserting an image file which consist of handwritten digits to get the prediction and accuracy percentage from my model.

LISTS OF FIGURES:

FIGURE TITLE **FIGURE NO.**

1. INTRODUCTION

1.1 OVERALL DESCRIPTION

Full Text Available Handwritten digit recognition plays a significant role in many user authentication applications in the modern world. As the handwritten digits are not of the same size, thickness, style and orientation. Therefore these challenges are to be faced to resolve this problem in my project. The objective of this project is to build a Graphical User Interface (GUI) in which we can draw the digit and recognize it straight away. I will be using a special type of deep neural network that is Convolutional Neural Network which is applied in analysing visual imagery where large set of pixel data in images are converted to conserve useful data of images which can be fed as input layer data to Artificial Neural Network for training purpose.

After that system will use hidden layers of CNN to develop a model for handwritten digit recognition. I will apply a 7 layer LeNet-5 Convolution Neural Network algorithm on Modified National Institute of Standards and Technology (MNIST) dataset which includes handwritten digits total of 70,000 images. Keras, a Neural Network library written in python will be used. Stochastic gradient and backpropagation algorithm are used for training the network and the forward algorithm is used for testing. Once the model is ready, user can input their image which consist of digit on our GUI and they will get correct prediction of their input.

1.2 PURPOSE

This project aims to meet the following objectives:

i. To develop handwritten digit recognizing system that enables users to automate the process of digit recognition using this deep learning model.

ii. To test the accuracy of the model

iii. Efficient model which is less computation intensive

1.3 MOTIVATION AND SCOPE

The task of handwritten digit recognition using a classifier has great importance and use such as online handwriting recognition on computer tablets, recognize zip codes on mail for postal mail sorting, processing bank check amounts, numeric entries in forms filled up by hand and so on. There are different 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. Our goal is to implement a pattern classification method to recognize the handwritten digits provided by the user.

The general problem we predicted we would face in this digit classification problem was the similarity between the digits like 1 and 7, 5 and 6, 3 and 8, 8 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.

2. LITERATURE SURVEY

Some of the works in the field of handwritten digit recognition have been listed below:
Pal and Singh [1] utilized multilayer perceptron (MLP) for recognizing handwritten English characters and achieved accuracy up to 94% and improved computation time for training the dataset.

Dutt and Dutt [2] demonstrated multilayer CNN using Keras and Theano libraries which attained 98.7% recognition accuracy on MNIST dataset. Ghosh and Maghari [3] did comparative study on three neural network approaches demonstrating that DNN was the best algorithm with 98.08% accuracy. However, every neural network has some error rate due to similarity in digit shape (e.g. 3 and 8 and 6 and 9).

Hamid [4] have performed handwritten digit recognition over MNIST dataset using CNN, SVM (Support Vector Machines) and KNN (K-Nearest Neighbour) classifiers. In their work, KNN and SVM predicted the outcomes correctly on datasets but Multilayer perceptron fails to recognize the digit 9 due to non-convex function as it gets stuck in the local minima. It was concluded that the accuracy would improve by using CNN with Keras.

LeNet-5 Architecture we used in this project was taken from ref. [5] which is going to form core part of our model.

Dataset used in this project was taken from yen Lecun MNIST dataset ref. [6]

After deep analysis of the related literature, it comes to know that CNN is supposed to be the best classifier than support vector machine (SVM), K-Nearest Neighbour (KNN) and random forest classifier (RFC) for HDR. Therefore, in this project, the task of HDR is accomplished by using the CNN, incorporating a 5 layer sequential CNN framework, with rectified linear units (ReLU) activations that have never been reported. The goal is achieved by establishing a model that can recognize and determine the handwritten digits from its image with high accuracy and low computation time. We aim to complete this by using the concepts of convolutional neural network. The proposed CNN framework is well equipped with suitable parameters for high accuracy of MNIST digit classification. The time factor is also considered for training the system. Furthermore, high accuracy is counter verified by changing the amount of CNN layers. Employment of additional pooling layers removes discretionary details in images and implants other higher-level characteristics. The MNIST dataset was used to train the network in experiments. MNIST is a handwritten digit dataset, which consists of 60,000 training images and 10,000 images in the test set. The digits are centered in a fixed size (28X28) image. These algorithms are employed to determine the accuracy with which these digits are classified. CNN classification proposed for HDR seems to be superior to other approaches used for handwritten characters/pattern identification in terms of high accuracy and low computational time. It was noticed that 7 layer LeNet-5 CNN training and prediction speed was efficient and quite good. So, the goal to create a model which can recognize the digits by implementing CNN-based framework for HDR will produce higher accurate and precise results

3. PROBLEM STATEMENT

Following are the constraints faced when computers approach to recognize handwritten digits:

1. The Handwritten digits are not always of the same size, width, orientation and justified to margins as they differ from writing of person to person.

2. The similarity between digits such as 1 and 7, 5 and 6, 3 and 8, 2 and 7 etc. So, classifying between these numbers is also a major problem for computers.

3. The uniqueness and variety in the handwriting of different individuals also influence the formation and appearance of the digits.

4. EXISTING SYSTEM

Handwritten digit recognition finds its application in various fields such as post mail sorting system where scanned images of mail envelopes are made into queue and extract the section describing postcode to be delivered. With the help of digit recognizer, sorting of mails can be done based on these postcodes according to their region.

Another application that utilizes this technique is form processing, digts are extracted from certain columns of a form and users put certain filters to get the desired results they want.

But there is no interface for a user to get their images scanned and recognized which makes the task complicated to use for a normal user.

5. PROPOSED METHODOLOGY

Deep Learning has emerged as a central tool for self-perception problems like understanding images, voice from humans, robots exploring the world. The project aims to implement the concept of Convolutional Network which is one of the important architecture of deep learning. Understanding CNN and applying it to the handwritten recognition system, is the major target of the proposed system. This project is divided into 3 sections:

4.1 Image Feature Extraction

During our method, we use CNN LeNet-5 [5] to obtain more diverse features from each handwritten digit image. The LeNet architecture is considered as the first architecture for convolutional neural networks. We can easily see from the LeNet-5 in Fig. 1 that many feature maps are generated in each layer. So we can obtain more diverse features than using other common methods. The LeNet-5 is an excellent architecture for handwritten digit recognition. The LeNet-5 has two parts, one is feature extraction, whereas the other one is classification which is used to classify objects. Given an image of $32 \times 32 \times 1$, firstly, a convolution layer with six 5×5 filters with the stride of 1 is used and an output matrix of $28 \times 28 \times 6$ is generated. With the stride of 1 and no padding, the feature map is reduced from 32×32 to 28×28. Then average pooling with the filter width of 2 and the stride of 2 is taken and the dimension is reduced by the factor of 2 and ends up with $14 \times 14 \times 6$. Furthermore, another convolution layer with sixteen 5×5 filters is used leading to an output matrix of $10 \times 10 \times 16$. Then another pooling layer is involved and ends up with an output matrix of $5 \times 5 \times 16$. Therefore, we extract sixteen 5×5 feature maps from each image, and each feature map (5×5) is treated as a column vector (25×1). Overall, there are two convolution layers, two subsampling layers, and two fully connected layers in the LeNet-5.

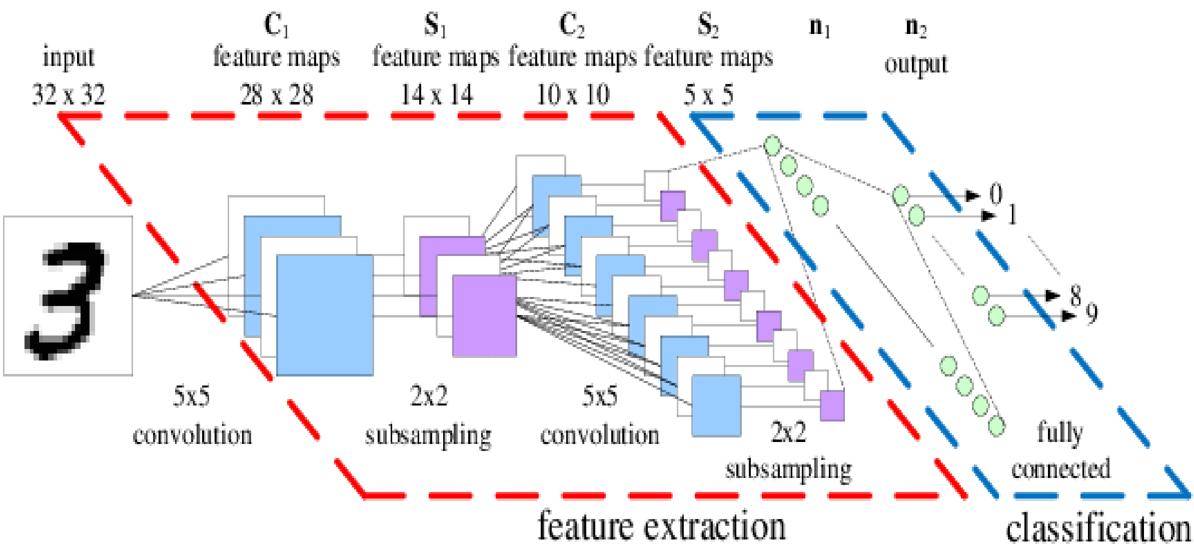


Figure 1: Architecture of LeNet-5 CNN Model

4.2 Image Classification

Once the feature extraction has been done, Pooled Feature Map is flattened to get fully connected layer. This fully connected layer has 120 feature maps each of size 1x1. Each of the 120 units is connected to all the 400 nodes (5x5x16) in the fully connected layer.

Sixth layer is a fully connected layer with 84 units in order to reduce number of trainable parameters from 48120 (with 120 units layer 5) to 10164.

Finally, there is a fully connected softmax ouput layer y^ with possible values corresponding to the digits from 0 to 9.

4.3 GUI development for digits prediction

After we get the desired testing input, an interface is developed for the purpose of enabling users with a choice to detect the digits depicted or written in images or drawer respectively. When users opens up the interface, they will be provided with an option to choose whether they wants to draw the digits all by themselves or insert image files from their local directory containing digits.

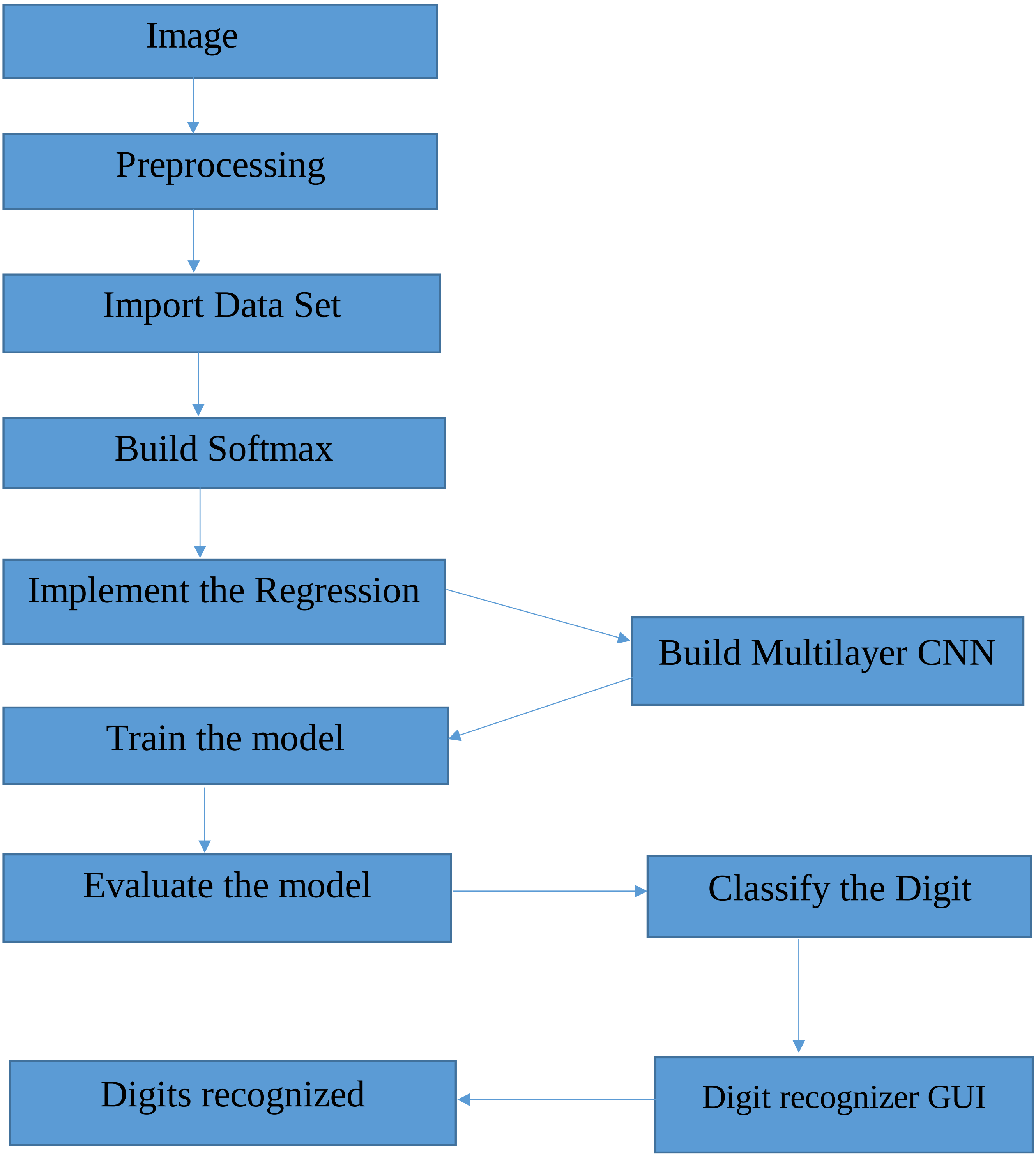
If users chooses first option, he will be guided to a drawer interface where he can draw digits by themselves and get their digits recognized along with their accuracy.

If users chooses second option, he will be asked to insert image file from their local directory and can get their digits written in image files predicted with optimum accuracy and along with percentage.

So by giving recognized digits as a results against inputs made by users to its users, this project fulfils all its objective

6. SYSTEM DESIGN

The following figure (Figure 1) provides a high level design of the system and the association between various modules used.



Data flow diagram

The DFD's have been show below:

DFD Level-0

The DFD Level-0 consists of two external entities, the UI and the Output, along with a process 1.0, representing the CNN for Digit Recognition as shown (figure 2). Output is obtained after processing.

Figure 2 DFD level-0

DFD Level-1

The DFD Level-1 (figure 3) consists of 2 external entities, the GUI and the Output, along with five process blocks and 2 data stores MNIST data and the Input image store, representing the internal workings of the CNN for Digit Recognition System. Process block 1.3 imports MNIST data from keras library. Process block 1.1 imports the image and process it and sends it to block 1.2 where regression model is built. It sends objects with probabilities to CNN where weights are updated and multiple layers are built. Block 1.5 trains and evaluates the model to generate output.

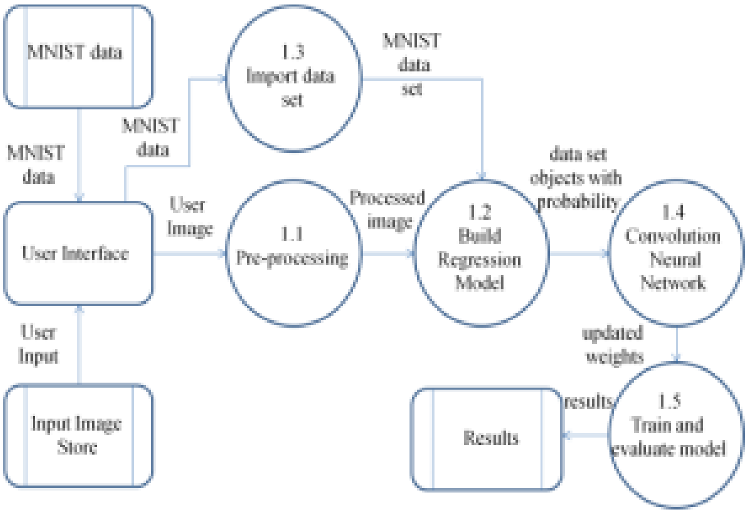


Figure 3 DFD level-1

DFD Level-2

The DFD Level-2 for import data(figure 4) consists of two external data and one entity UI along with three process blocks, representing the three functionalities of the CNN for Digit Recognition System. It imports data from MNIST data store and stores on the system.

Figure 4 DFD level-2

DFD Level-2(regression model)

The DFD Level-2 for regression model (figure 5) consist of a MNIST data store and 4 process block. Block 1.2.1 creates placeholders for inputs. Variables are created and initialized (block

1.2.2) weights are assigned to variables (block 1.2.3) and regression model is built (block 1.2.4).

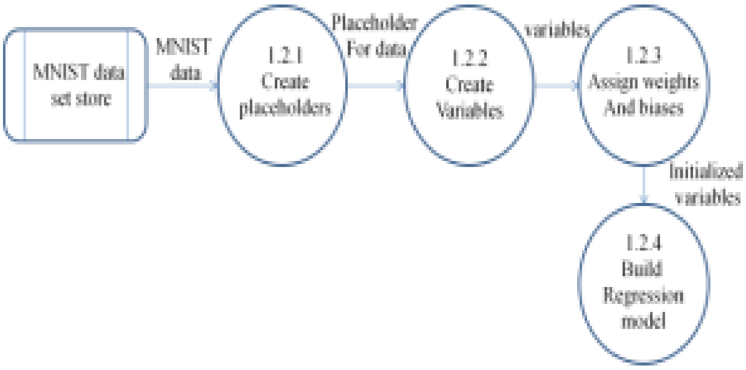


Figure 5 DFD level-2(regression model)

DFD Level-2(Build CNN)

The DFD Level-2 for build CNN (figure 6) consists of an external entity Regression Model along with 7 process blocks, weights are initialized (block 1.4.1) and passed to convolutional layer and pooling layers (block 1.4.2). Then layers of CNN are built (blocks 1.4.3, 1.4.4, 1.4.5) and dropout (block 1.4.6) and then final readout is obtained.

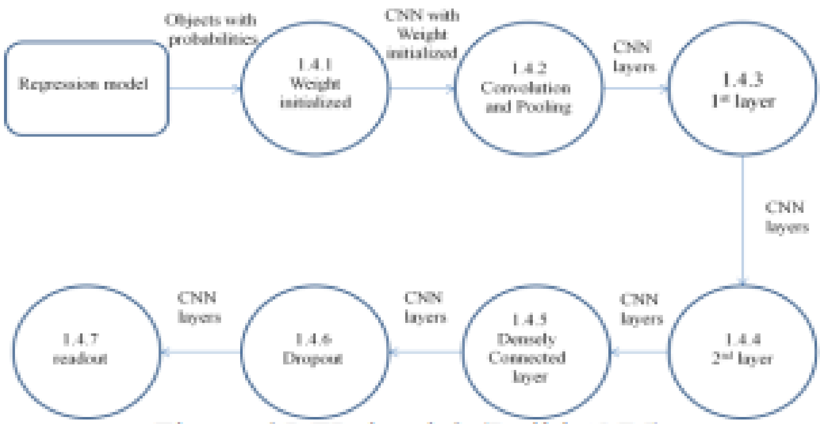


Figure 6 DFD level-2 (Build CNN)

DFD Level-2(Train and Evaluate model)

The DFD Level-2 for (Train and Evaluate Model) (figure 7) consists of an external entity CNN along with 4 process blocks, cross entropy (block 1.5.1) and optimization (block 1.5.2) is done on the results obtained from CNN block. Labels are predicted using aggregation function (block 1.5.3) and accuracy is determined (block 1.5.4) and results are stored in results data store.

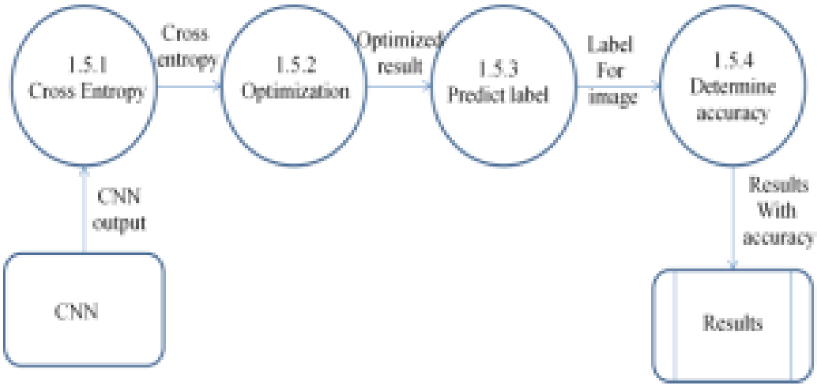


Figure 7 DFD level-2 (Train and Evaluate Model)

7. IMPLEMENTATION

Following steps were involved in implementing this whole project

6.1 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 starts working with it. The **mnist.load_data()** method returns us the training data, its labels and also the testing data and its labels.

```
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import AveragePooling2D
from keras.layers import Flatten
from keras.layers import Dense
from keras import utils as np_utils
import matplotlib.pyplot as plt
from keras.preprocessing import image
```

6.2 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 model. 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) which means we have 60000 images , each of dimension 28x28 and 1 color channel i.e. grayscale.

```
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)
y_train =np_utils.to_categorical(y_train, num_classes=10)
y_test = np_utils.to_categorical(y_test, num_classes=10)
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
```

6.3 Create the model

Now we will create our CNN model in this project. A CNN model is consists of convolutional layers 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. We used LeNet-5 algorithm for the purpose of creating our CNN model and it consists of alternative convolutional and pooling layers followed by 3 fully connected layers and give softmax regression layer as output to classify whole outputs into 10 category.

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 for fast optimization due to its fast learning rate.

```
model = Sequential()
model.add(Convolution2D(6,5,5,input_shape=(28,28,1),activation='relu'))
model.add(AveragePooling2D())
model.add(Convolution2D(16,5,5,activation='relu'))
model.add(AveragePooling2D())
model.add(Flatten())
model.add(Dense(units=120,activation='relu'))
model.add(Dense(units=84,activation='relu'))
model.add(Dense(units=10,activation='softmax'))

model.compile(optimizer='Adadelta',loss='categorical_crossentropy',metrics=['accuracy'])
```

6.4 Train the model

The **model.fit() function** of keras will start the training of the model**. It takes the training data, valiadation 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. This file is used to predict digits in our GUI.

```
hist = model.fit(x_train, y_train,batch_size=128,epochs=30,verbose=1,
                 validation_data=(x_test, y_test))
model.save('mnist.h5')
```

6.5 Evaluate the model

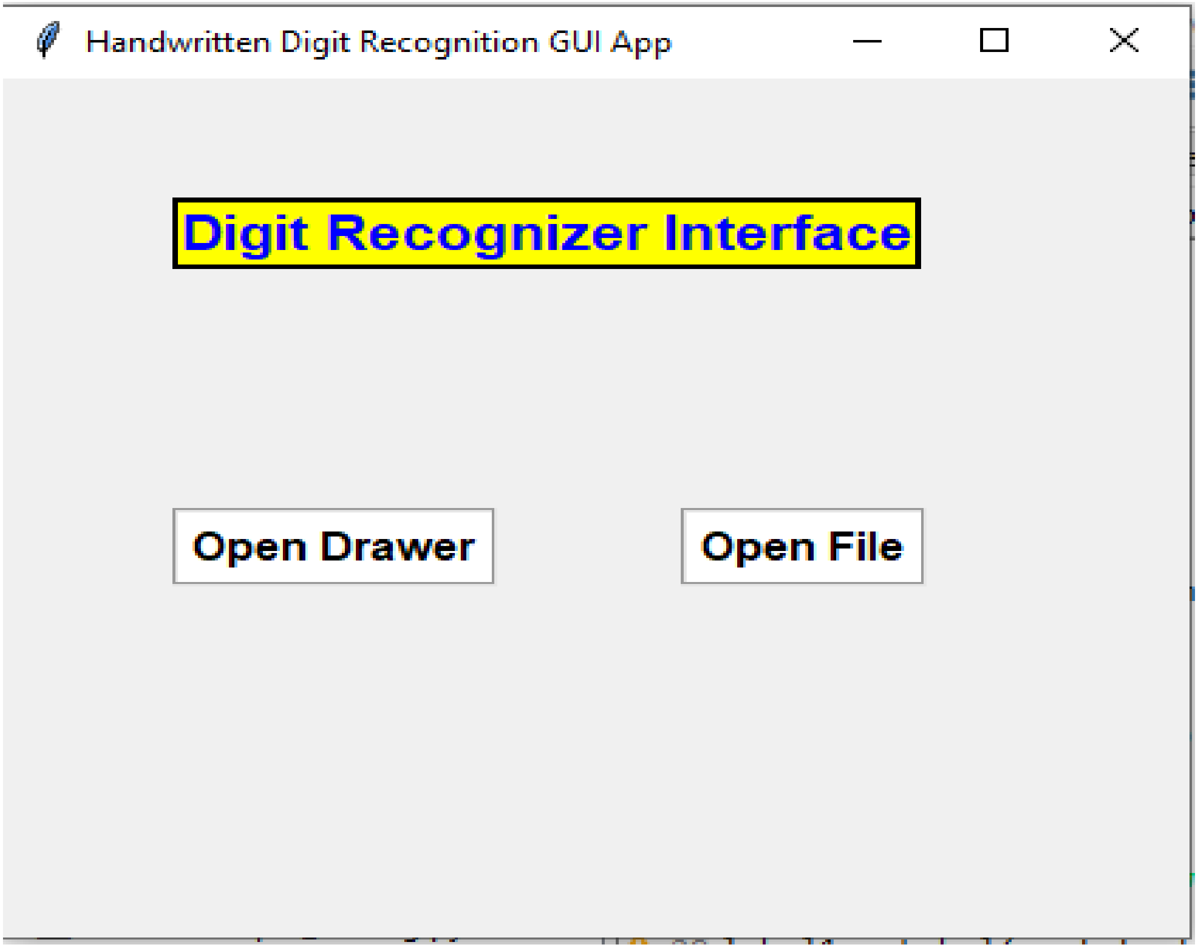
We have 10,000 images in our dataset which will be used to evaluate how good our model works. The testing data was not involved in the training of the data, therefore, it is new data for our model. The MNIST dataset is well balanced so we can get optimum accuracy.

```
print("TESTING THE MODEL:")
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

6.6 Create GUI to predict digits

An interface is created which enables users with option to give their inputs either by drawing digits on a canvas or uploading image containing digits to the interface.

In the drawer option, user has to draw digits on a canvas and with a button, they can get their drawn digits recognized whereas they can also upload their handwritten digits image to the system to get it recognized. We have used Tkinter library which comes in the python standard library.we have created a function **recognize_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 recognize_digit() function and display the results.



8.OUPUT AND RESULTS

When tested upon by users we got following results from out GUI app and based on these results, app is ready to be used.

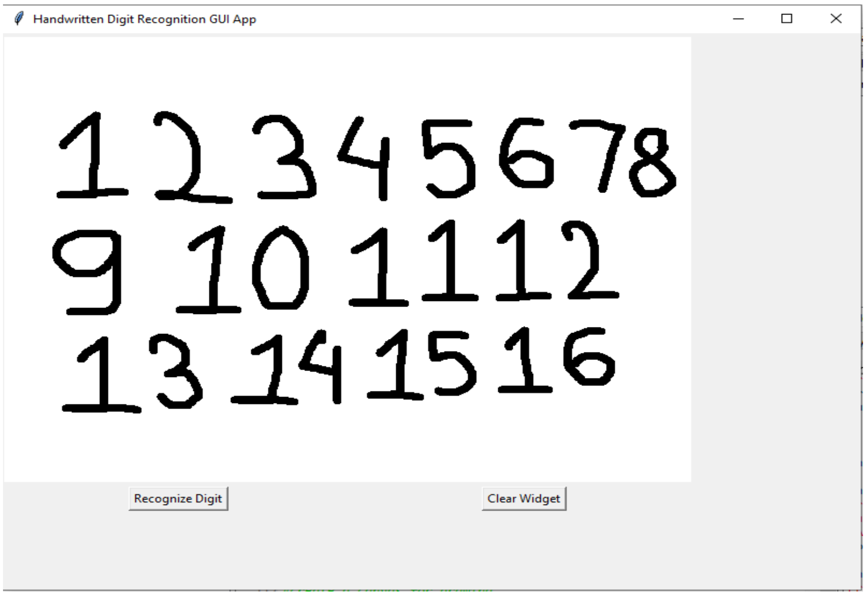


Figure 8(a): User Drawing on canvas

In figure above, user has drawn some digits for the purpose of getting them recognized.

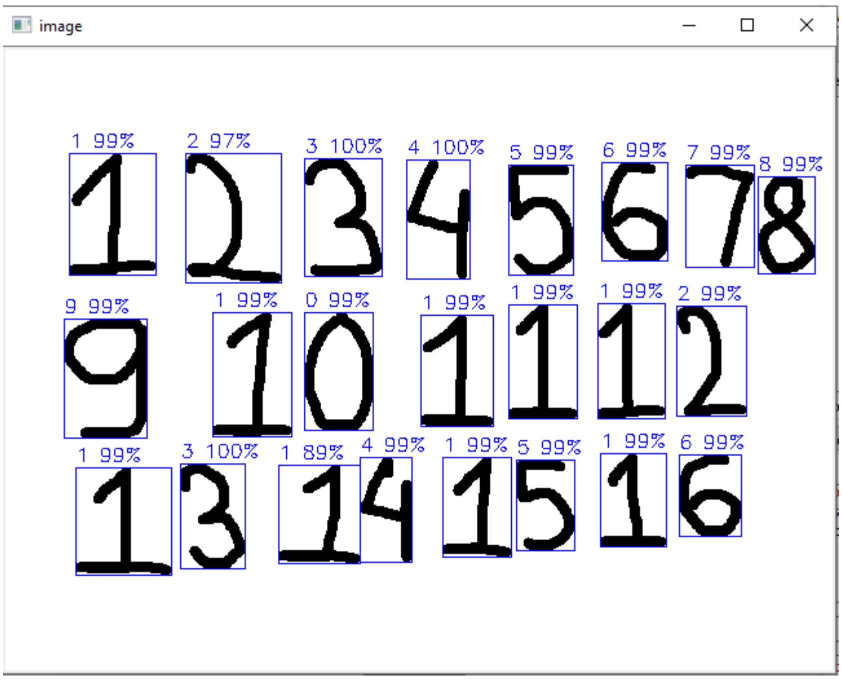


Figure 8(b): Recognition result for option 1

In the above figure, user gets their recognition result where the model has been predicting the numbers drawn along with their accuracy percentage.

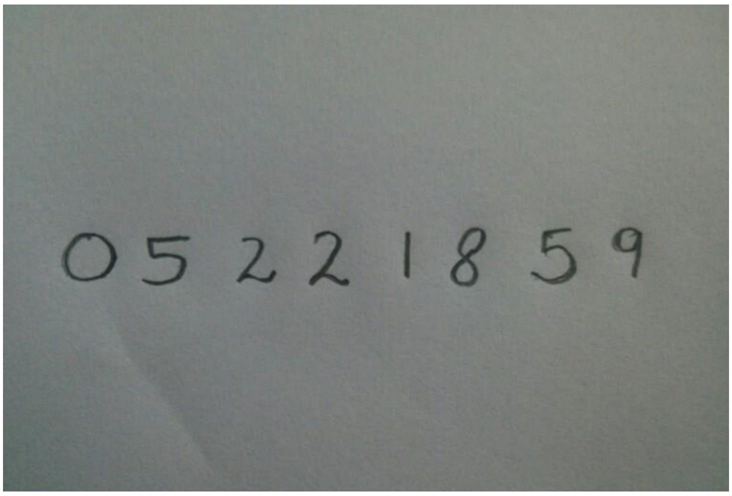


Figure 8(c): Inserted image from user

In figure 8(c), user chose option 2 and inserted an image which has handwritten digits to be recognized.

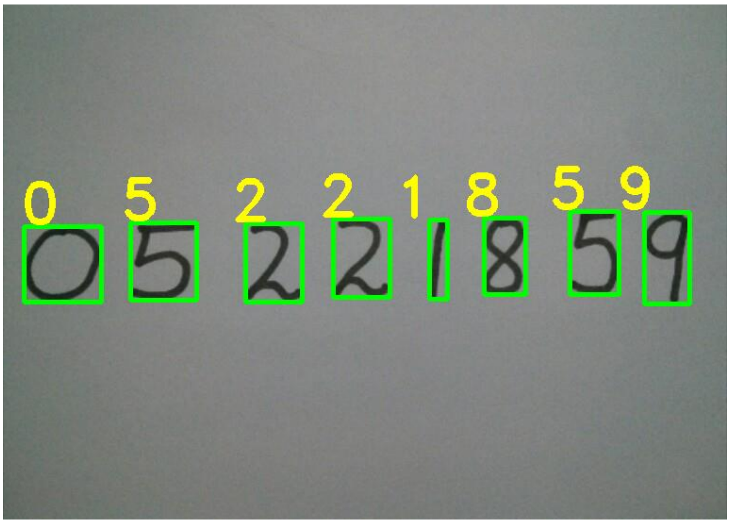
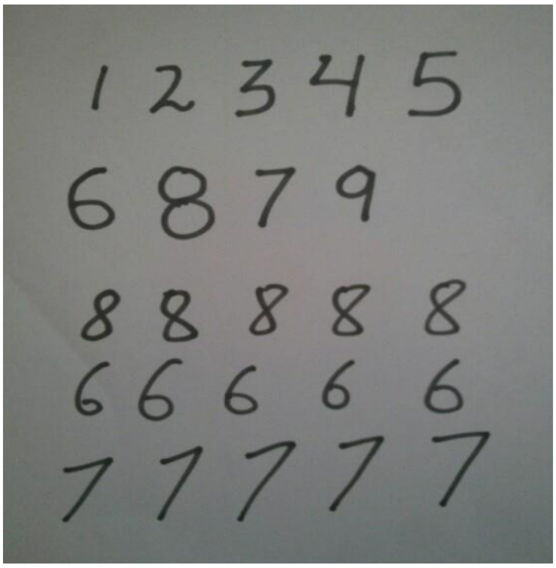
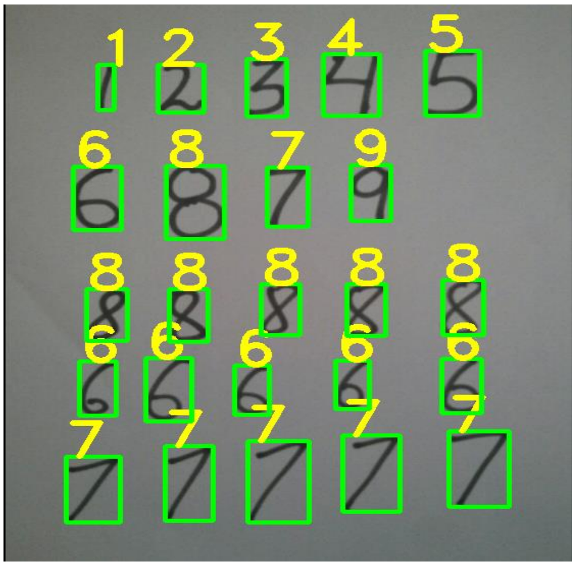


Figure 8(d): Recognized result for option 2

In above figure, our app has successfully recognized digits written in an image file and user gets the desired result.





Above two figures shows multi-digit detection using option two and with different writing styles of some digits.

9. CONCLUSION AND FUTURE SCOPE

The objectives with which this project was initiated such as to develop handwritten digit recognizing system that enables users to recognize their handwritten digits using this deep learning model, less compuatation intensive efficient model has been achieved. The model which I built got an average accuracy of 98.23%.

Also the underlying problems of not having the same size ,width, orientation, and margin always has been taken care of with the help of computer vision’s opencv library’s functionalities. The problem of difficulty in distinguishing the difference between digits such as 1 and 7, 5 and 6, 3 and 8 etc has been resolved to a great extent with the opencv’s edge detection and contour features. Also problems of dim lighting and blurry or unclear edges in images are corrected with the help of Gaussian blur technique. Now users can find their handwritten digits in one go without much complications.

This project is based on a deep neural network where users are going to get an interface for recognition of their digit images. On the top of this model, this project can be extended to append various functionalities which can be used to filter the desired results based on digits recognized by this model.

For instance, if any academic institute wants to disburse scholarship to their talented students who lacks money can use this model to process forms submitted by students and filters the needy students.

8. REFERENCES

[1] Pal A, Singh D (2010) Handwritten english character recognition using neural network. Int J Comput Sci Commum 1(2): 141-144

[2] Dutt A, Dutt A (2017) Handwritten digit recognition using deep learning. Int J AdvRes Comput Eng Technol 6(7):990–997

[3] Ghosh MMA, Maghari AY (2017) A comparative study on handwriting digit recognition using neural networks. In: International conference on promising electronic technologies, pp 77–81

[4] Hamid, N.B.A., Sjarif, N.N.B.A.: Handwritten recognition using SVM, KNN and neural network. www.arxiv.org/ftp/arxiv/papers/1702/1702.00723

[5] Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review Scientific Figure ResearchGate Available from:

https://www.researchgate.net/figure/Architecture-of-LeNet-5-LeCun-et-al-1998_fig2_317496930

[6] http://yann.lecun.com/exdb/mnist