The Project Review-1Report

On

HANDWRITTEN TEXT RECOGNITION USING MACHINE LEARNING

Submitted in partial fulfillment of the

requirement for the award of the degree

of

COMPUTER SCIENCE AND ENGINEERING

GALGOTIAS UNIVERSITY

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

Under The Supervision of
Mr. GAUTAM KUMAR
Assistant Professor

Submitted By

SAUMYA KUMARI – 19SCSE1010243 / 19021011432
PRERNA SONI – 19SCSE1010382 / 19021011563

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA
INDIA

FALL 2020-21

Table of Content

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the thesis/project/dissertation, entitled **"Handwritten Text Recognition using Machine Learning"** in partial fulfillment of the requirements for the award of the Project review submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of month, Year to Month and Year, under the supervision of Name... Designation, Department of Computer Science and Engineering/Computer Application and Information and Science, of School of Computing Science and Engineering , Galgotias University, Greater Noida.

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

Saumya Kumari - 19SCSE1010243-

Prerna Soni - 19SCSE1010382-

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Mr. Gautam Kumar

AP/SCSE

Galgotias University

ACKNOWLEDGMENT

I am sincerely thankful to Galgotias University, Greater Noida for providing me with the opportunity to write a research paper on the topic **"Handwritten Text Recognition using Machine Learning".**

We would like to thank our guide **Mr. Gautam Kumar** for guiding us in every stage of this project. Without his support, it would have been very difficult for me to prepare so meaningful and interesting project.

This paper has helped me a lot to learn about machine learning and deep learning and I hope it helps people to have a basic understanding of handwritten text recognition and its detection.

ABSTRACT

The aim of this project is to review existing methods for the handwritten character recognition problem using machine learning algorithms and implement one of them for a user-friendly Android application .Handwriting Detection is a technique or way of a Computer to receive and interpret intelligible handwritten input from source such as paper documents, touch screen, photo graphs etc. Handwritten Text recognition is one of area pattern recognition. The purpose of pattern recognition is to categorizing or classification data or object of one of the classes or categories. Handwriting recognition is defined as the task of transforming a language represented in its spatial form of graphical marks into its symbolic representation. The goal of handwriting is to identify input characters or image correctly then analyzed to many automated process systems. We have designed to detect the writings of different format. Our main objective is to compare the accuracy of the models stated above along with their execution time to get the best possible model for digit recognition.

LIST OF TABLES

LIST OF FIGURES

ACRONYMS

B.TECH	BACHELOR OF TECHNOLOGY
M.TECH	MASTER OF TECHNOLOGY
BCA	BACHELOR OF COMPUTER APPLICATION
MCA	MASTER OF COMPUTER APPLICATION
B.SC. (CS)	BACHLEOR OF SCIENCE
M.SC. (CS)	MASTER OF SCIENCE
SCSE	SCHOOL OF COMPUTER SCIENCE AND ENGIINEERING

1. INTRODUCTION

In spite of the bounty of specialized composing devices, many individuals actually decide to take their notes customarily with pen and paper. In any case, there are downsides to penmanship text. It's hard to store and access actual archives in an effective way, search through them productively, and share them with others In this manner, a great deal of significant information gets lost or doesn't get inspected in view of the way that reports won't ever get moved to computerized design. We have along these lines chosen to handle this issue in our task since we accept the fundamentally more prominent simplicity of the board of computerized text thought about to composing text will help individuals all the more successfully access, search, share and dissect their records, while as yet permitting them to utilize their favoured composing technique. The point of this venture is to additionally investigate the errand of ordering manually written text and to change over transcribed text into the advanced organization. Written by hand text is an exceptionally broad term, and we needed to limit the extent of the project by determining the importance of the manually written text for our motivations. In this undertaking, we assumed the test of grouping the picture of any manually written word, which may be as cursive or square composition.The purpose of our model is to further explore the task of classifying handwritten text and to change or convert handwritten text into the digital format. Handwritten text is a very general term, and we wanted to narrow down the scope of the project by specifying the meaning of handwritten text for our purposes. In this model, we took on the challenge of classifying the image of any handwritten word, which might be of the form of cursive or block writing. Our model can be combined with algorithms that segment the word images in a given line image, which can in turn be combined with algorithms that segment the line images in a given image of a whole handwritten page.

1.1 PROPOSED WORK

A. Handwritten text recognition

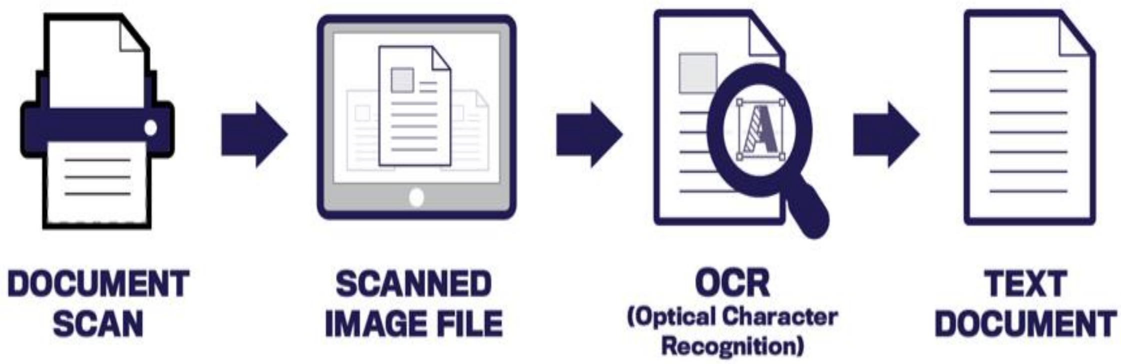
Handwritten Text Recognition (HTR) systems consist of handwritten text in the form of scanned images as shown in figure 1. we are going to build a Neural Network (NN) which is trained on word-images from the IAM dataset. because the input layer (and therefore also all the opposite layers) are often kept small for word-images, NN-training is possible on the CPU (of course, a GPU would be better). For the implementation of HTR, the minimum requirement is TF.

Fig. 1: Image of word taken from IAM Dataset

B. Character Recognition Algorithms

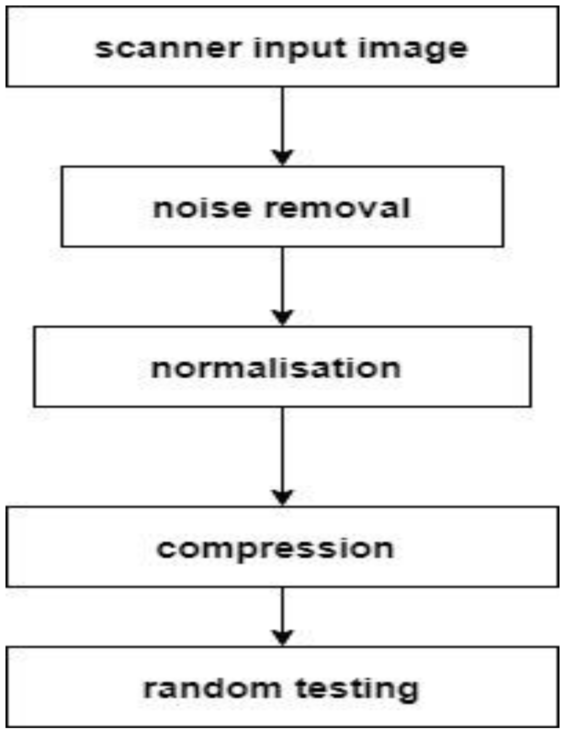
The algorithms used in character recognition can be divided into three categories: Image Pre-processing, Feature Extraction, and Classification.

They are normally used in sequence – image pre-processing helps makes feature extraction a smoother process, while feature extraction is necessary for correct classification.



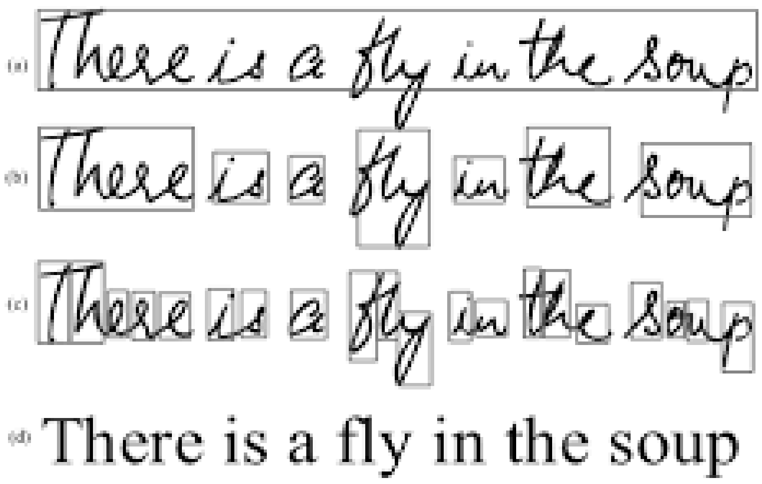
C. Image pre-processing

Image pre-processing is crucial in the recognition pipeline for correct character prediction. These methods typically include noise removal, image segmentation, cropping, scaling, and more. The recognition system first accepts a scanned image as an input. The images can be in JPG format.



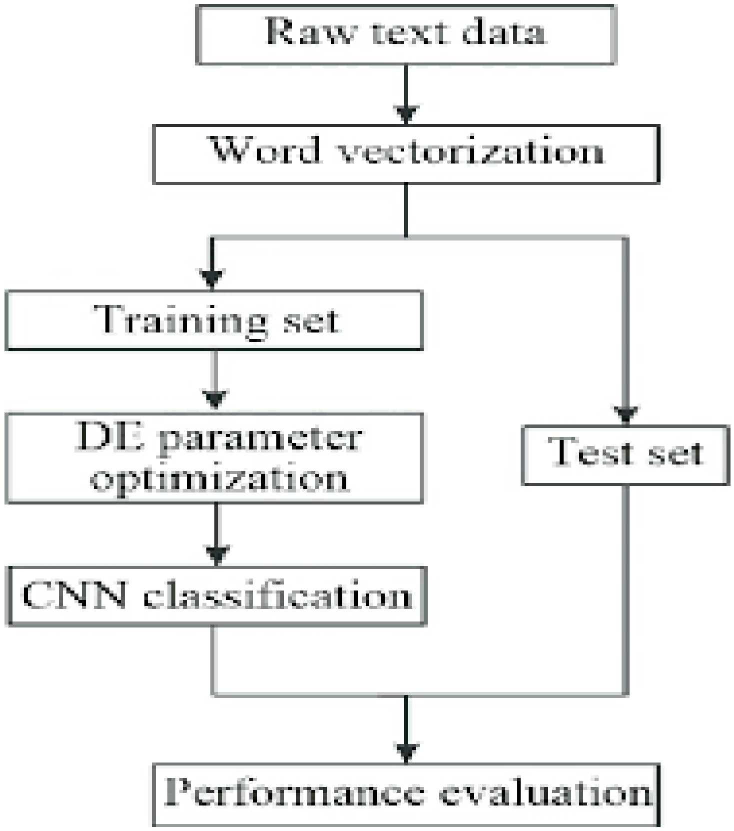
D. Segmentation

In the segmentation stage, a sequence of characters is segmented into a sub-image of an individual character. Each character is resized into 30×20 pixels.



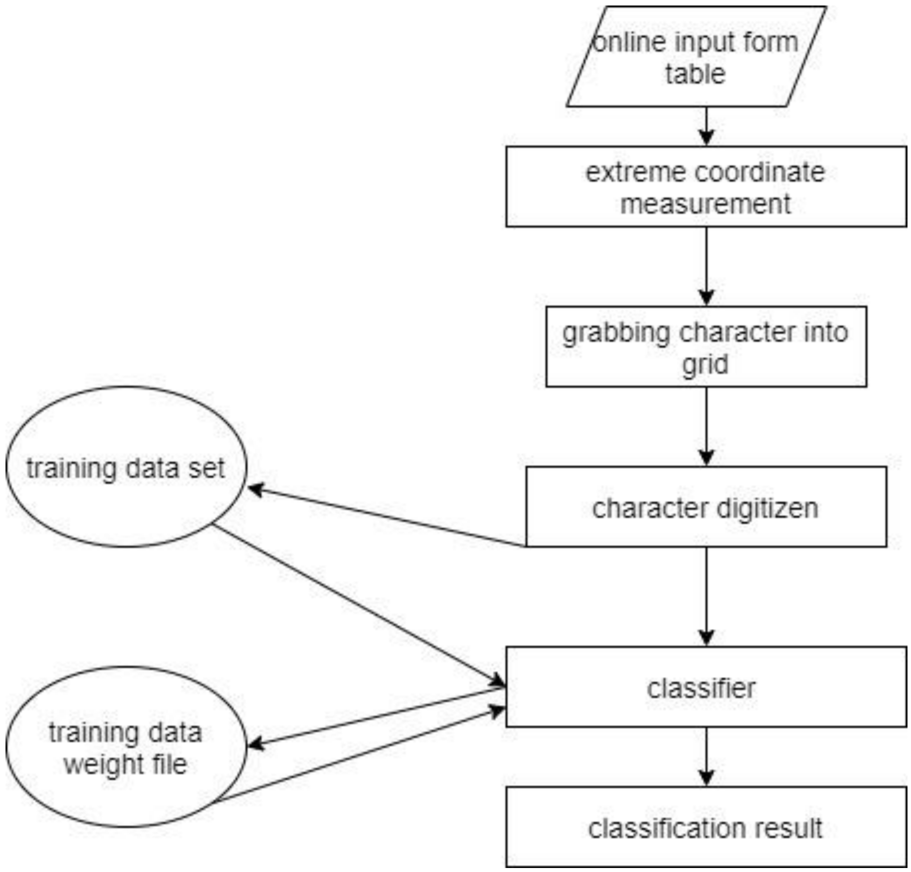
E. Classification and Recognition

This stage is the decision making stage of the recognition system. The classifier contains two hidden layers, using a log sigmoid activation function to train the algorithm.



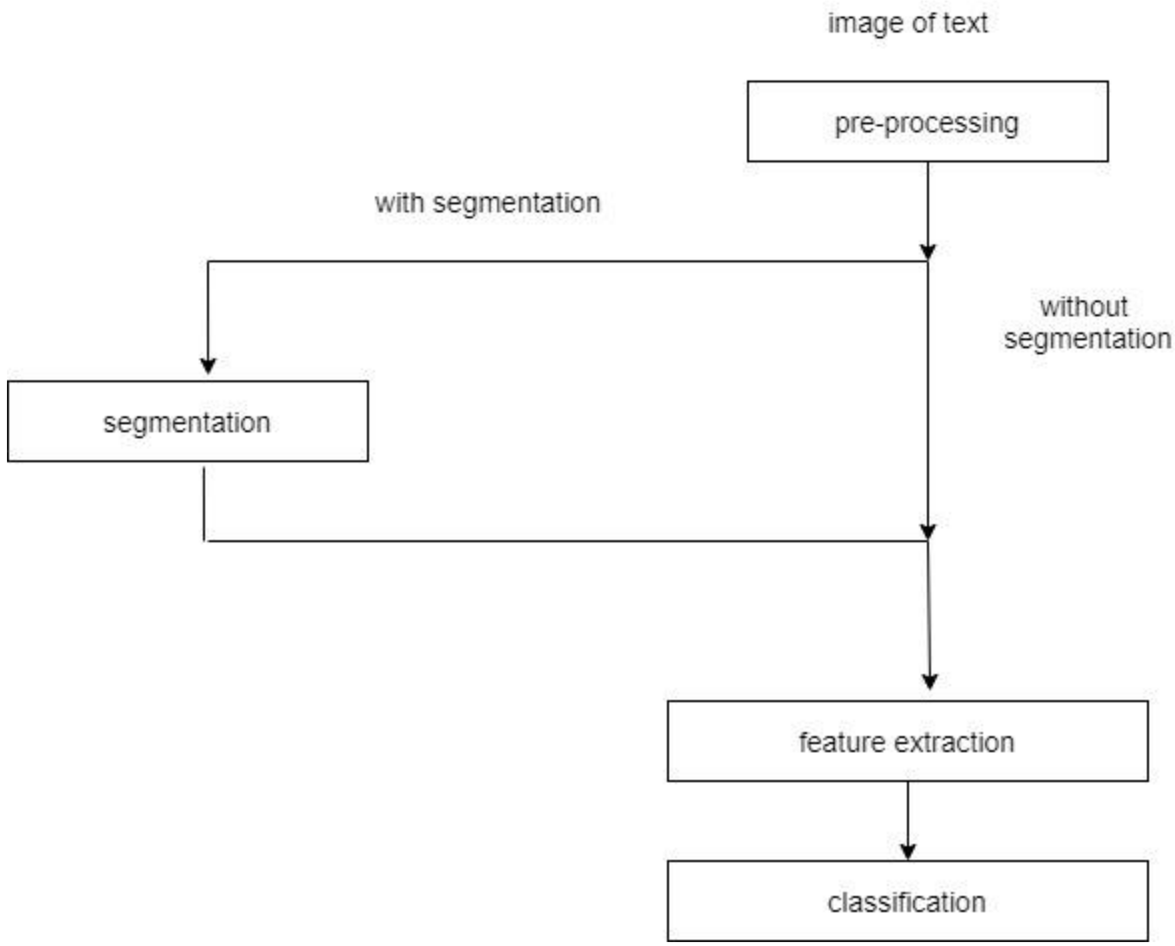
F. Feature Extraction

The features of input data are the measurable properties of observations, which is used to analyse or classify these instances of data. The task of feature extraction is to identify relevant features that discriminate the instances that are independent of each other.



G. Recognition

A method for continuous handwritten word recognition is derived when the word is segmented into triplets (containing 3 letters). Two subsequent triplets have 2 common letters. The biggest challenge for recognition systems is to perform operations on a continuous word. In this, each word is subdivided into triplets, each containing three letters. Figure 10a shows triplet “aba” and figure 10b shows triplet “ban”. Two neighbour triplets always contain two common letters which represent the overlapping between letters. This kind of overlapping results is a higher recognition rate.



1.2. Introduction to Machine Learning

Machine learning is a subfield of artificial intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people.

Although machine learning is a field within computer science, it differs from traditional computational approaches. In traditional computing, algorithms are sets of explicitly programmed instructions used by computers to calculate or problem solve.

Machine learning algorithms instead allow for computers to train on data inputs and use statistical analysis in order to output values that fall within a specific range. Because of this, machine learning facilitates computers in building models from sample data in order to automate decision-making processes based on data inputs.

Machine Learning Methods

Two of the most widely adopted machine learning methods are **supervised learning** which trains algorithms based on example input and output data that is labeled by humans, and **unsupervised learning** which provides the algorithm with no labeled data in order to allow it to find structure within its input data. Let's explore these methods in more detail.

Supervised Learning

In supervised learning, the computer is provided with example inputs that are labeled with their desired outputs. The purpose of this method is for the algorithm to be able to "learn" by comparing its actual output with the "taught" outputs to find errors, and modify the model accordingly. Supervised learning therefore uses patterns to predict label values on additional unlabeled data.

For example, with supervised learning, an algorithm may be fed data with images of sharks labeled as fish and images of oceans labeled as water. By being trained on this data, the supervised learning algorithm should be able to later identify unlabeled shark images as fish and unlabeled ocean images as water.

Unsupervised Learning

In unsupervised learning, data is unlabeled, so the learning algorithm is left to find commonalities among its input data. As unlabeled data are more abundant than labeled data, machine learning methods that facilitate unsupervised learning are particularly valuable.

The goal of unsupervised learning may be as straightforward as discovering hidden patterns within a dataset, but it may also have a goal of feature learning, which allows the computational machine to automatically discover the representations that are needed to classify raw data.

Unsupervised learning is commonly used for transactional data. You may have a large dataset of customers and their purchases, but as a human you will likely not be able to make sense of what similar attributes can be drawn from customer profiles and their types of purchases.

Without being told a “correct” answer, unsupervised learning methods can look at complex data that is more expansive and seemingly unrelated in order to organize it in potentially meaningful ways. Unsupervised learning is often used for anomaly detection including for fraudulent credit card purchases, and recommender systems that recommend what products to buy next. In unsupervised learning, untagged photos of dogs can be used as input data for the algorithm to find likenesses and classify dog photos together.

1.3. What is Deep Learning?

Machine learning is about computers being able to perform tasks without being explicitly programmed… but the computers still think and act like machines. Their ability to perform some complex tasks — gathering data from an image or video, for example — still falls far short of what humans are capable of.

Deep learning models introduce an extremely sophisticated approach to machine learning and are set to tackle these challenges because they've been specifically modeled after the human brain. Complex, multi-layered "deep neural networks" are built to allow data to be passed between nodes (like neurons) in highly connected ways. The result is a non-linear transformation of the data that is increasingly abstract.

While it takes tremendous volumes of data to 'feed and build' such a system, it can begin to generate immediate results, and there is relatively little need for human intervention once the programs are in place.

Deep Learning are categorised into two types:

Convolutional Neural Networks

Convolutional neural networks are specially built algorithms designed to work with images. The 'convolution' in the title is the process that applies a weightbased filter across every element of an image, helping the computer to understand and react to elements within the picture itself.

This can be helpful when you need to scan a high volume of images for a specific item or feature; for example, images of the ocean floor for signs of a shipwreck, or a photo of a crowd for a single person's face.

This science of computer image/video analysis and comprehension is called 'computer vision', and represents a high-growth area in the industry over the past 10 years.

Recurrent Neural Networks

Recurrent neural networks, meanwhile, introduce a key element into machine learning that is absent in simpler algorithms: memory. The computer is able to keep past data points and decisions 'in mind', and consider them when reviewing current data – introducing the power of context.

This has made recurrent neural networks a major focus for natural language processing work. Like with a human, the computer will do a better job understanding a section of text if it has access to the tone and content that came before it. Likewise, driving directions can be more accurate if the computer 'remembers' that everyone following a recommended route on a Saturday night takes twice as long to get where they are going.

1.4. Deep learning versus Machine learning.

- Deep learning is a type of machine learning, which is a subset of artificial intelligence.
- Machine learning is about computers being able to think and act with less human intervention; deep learning is about computers learning to think using structuredmodels on the human brain.
- Machine learning requires less computing power; deep learning typically needs less ongoing human intervention.
- Deep learning can analyze images, videos, and unstructured data in ways machine learning can’t easily do.
- Every industry will have career paths that involve machine and deep learning.

1.5. What is handwritten text recognition?

Handwriting recognition is the ability of a computer or device to take as input handwriting from sources such as printed physical documents, pictures and other devices, or to use handwriting as a direct input to a touchscreen and then interpret this as text.

The input is usually in the form of an image such as a picture of handwritten text that is fed to a pattern-recognition software, or as real-time recognition using a camera for optical scanning.

Optical character recognition (OCR) is the most mainstream technique used for handwriting recognition. This is done by scanning a handwritten document and then converting it into a basic text document. This also works by taking a picture of a handwritten text. OCR is basically a form of image recognition that is meant to recognize handwriting instead of faces or shapes such as landmarks.

1.6. How does CNN work?

A **Convolutional Neural Network (ConvNet/CNN)** is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

Convolutional neural networks are made up of several layers of artificial neural neutrons and arrival neutrons. A rough indication of this biological counterpart, a mathematical function calculates the weighted sum of the multiple inputs and generates a trigger value.

1.7. Tools and Technology Used :

HARDWARE REQUIREMENTS:

- System : Pentium or later processors (2.4 GHz).
- Hard Disk : 10 GB.
- Monitor
- Keyboard & Mouse
- Ram : 4 GB

SOFTWARE REQUIREMENTS:

- Operating system : Windows 7 or later versions
- Coding Language : Python
- IDE : Python IDE (Google Colaboratory)

2. Literature Survey

We have used the dataset of for this model ,there have been several accomplishments that has been achieved using this dataset. Even before using machine learning, Handwritten text recognition has been made possible and easy, however their accuracies were really low or they had a relatively small dataset as said by Line. In this paper, usage of OCR has been discussed such as in Speech Recognition, Radio Frequency, Vision systems, Magnetic Stripes, Bar Code and Optical Mark Reading. A popular machine learning task is classifying the MNIST dataset, which is dataset of numbers. Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis by Simard, Steinkraus and Platt is a valuable paper for understanding usage of convoluted neural networks (CNNs). For word recognition, a Paper by Pham et al., used a 2-layer CNN which fed into a bidirectional recurrent neural networks (RNN) with Long Short-Term Memory (LSTM) cells [3]. The best model implemented, according to us, is by Graves and Schmiduber with a multidimensional RNN structure. [4] Another paper on ’Handwritten Text Recognition’ by M.J. Castro-Bleda dealt with dataset with slanted words as well and corrected them during pre-processing. [5] Development of English Handwritten Recognition Using Deep Neural Network by Teddy Surya and Ahmad Fakhrur uses a Deep Neural Network model having two Encoding layer and one SoftMax layer on the EMNIST dataset.

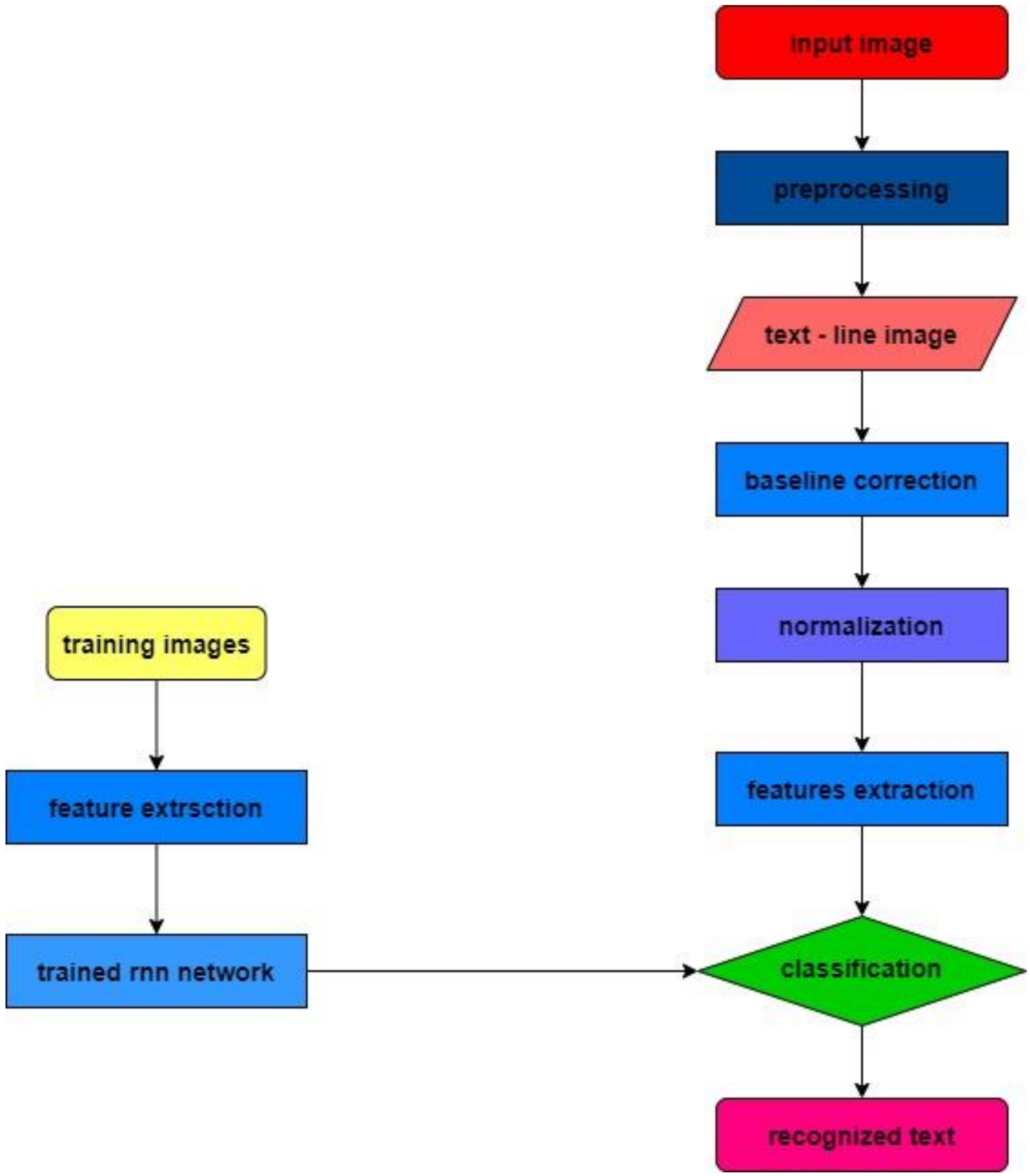


Figure shows the flow chart of the proposed project design

3. Functionality/Working of project

1. Mounting Google drive

```
[1] #Mount Google drive
    from google.colab import drive
    drive.mount('/content/drive')

    Mounted at /content/drive
```

```
[2] #Change the directory to the path where you have all your input files
    %cd /content/drive/My Drive/project work/Machine-Learning-Handwritten-Character-Recognition-main

    #Below command shows you all thee files present in your current directory
    #!ls

    /content/drive/My Drive/project work/Machine-Learning-Handwritten-Character-Recognition-main
```

2. Importing all the necessary libraries.

```
[7] import matplotlib.pyplot as plt
    import cv2
    import numpy as np
    from tensorflow.keras.optimizers import SGD, Adam
    from keras.callbacks import ReduceLROnPlateau, EarlyStopping
    from tensorflow.keras.utils import to_categorical
    import pandas as pd
    import numpy as np
    from sklearn.model_selection import train_test_split
    from sklearn.utils import shuffle
```

```
# After installing these packages you have to import the dataset you can get all thes files
# and data set in the GITHUB Account
# This data set is showing all the
data = pd.read_csv(r"A_Z Handwritten Data.csv").astype('float32')
print(data.head(20)) #We are getting first ten images data
#This data set contains all the Images in data form
```

```
      0  0.1  0.2  0.3  0.4  0.5  ...  0.643  0.644  0.645  0.646  0.647  0.648
0   0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
1   0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
2   0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
3   0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
4   0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
5   0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
6   0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
7   0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
8   0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
9   0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
10  0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
11  0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
12  0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
13  0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
14  0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
15  0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
16  0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
17  0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
18  0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0
19  0.0  0.0  0.0  0.0  0.0  0.0  ...    0.0    0.0    0.0    0.0    0.0    0.0

[20 rows x 785 columns]
```

3. Preparation for the model to get X and Y axis and also splitting the test and train dataset.

```
[8] #Preparation for the Machine Learning Model we have to get the X and Y axis.
    #In X axis we are droping the first column
    #The ‘0’ contains the labels, & so we drop the ‘0’ column from the data dataframe read
    X = data.drop('0',axis = 1)
    y = data['0'] #and put the 0 column in the label y
```

```
#IN this step Machine learning Model preparation for spliting test and Train data. We
#Test size would be your choise here I put it 0.2 % data
train_x, test_x, train_y, test_y = train_test_split(X, y, test_size = 0.2)
#The below step  is to reshape the Image and Label data according to our requirement
#shape size would be [28,28]

train_x = np.reshape(train_x.values, (train_x.shape[0], 28,28))
test_x = np.reshape(test_x.values, (test_x.shape[0], 28,28))
print("Train data shape: ", train_x.shape)
print("Test data shape: ", test_x.shape)
#After printing you will get the Train Data "Train data shape:  (297960, 28, 28)"
#After printing you will get the Test Data "Test data shape:  (74490, 28, 28)"

Train data shape:  (297960, 28, 28)
Test data shape:  (74490, 28, 28)
```

4. Introduced a self made dictionary.

```
[14] word_dict = {0:'A',1:'B',2:'C',3:'D',4:'E',5:'F',6:'G',7:'H',8:'I',9:'J',10:'K',11:'L',12:'M',13:'N',14:'O',15:'P',16:'Q',17:'R',18:'S',19:'T',20:'U',21:'V',22:'W',23:'X', 24:'Y',25:'Z'}
```

```
y_int = np.int0(y)
count = np.zeros(26, dtype='int')
print(count)
for i in y_int:
   count[i] +=1

[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
```

```
[12] alphabets = []
     for i in word_dict.values():
         alphabets.append(i)
```

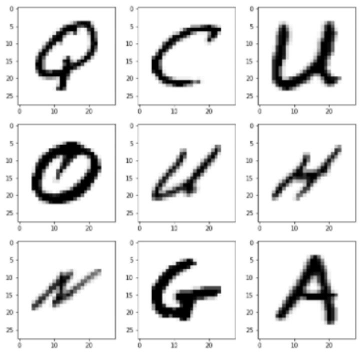
```
[13] alphabets[5]

'F'
```

5. Representing a bar graph for the number of occurrence of alphabets.

```
fig, ax = plt.subplots(1,1, figsize=(10,10))
ax.barh(alphabets, count)
plt.xlabel("Number of elements ")
plt.ylabel("Alphabets")
plt.grid()
plt.show()
```

6. Shuffle the threshold binary images.

```
shuff = shuffle(train_x[:10])
fig, ax = plt.subplots(3,3, figsize = (10,10))
axes = ax.flatten()
for i in range(9):
    shu = cv2.threshold(shuff[i], 30, 200, cv2.THRESH_BINARY)
    axes[i].imshow(np.reshape(shuff[i], (28,28)), cmap="Greys")
plt.show()
```



7. Reshaping of Data.

```
[18] #Data Reshaping
     #We changed the Images reshaping to "New shape of train data:  (297960, 28, 28, 1)"
     #For both Train and Test data
     train_X = train_x.reshape(train_x.shape[0],train_x.shape[1],train_x.shape[2],1)

     print("New shape of train data: ", train_X.shape)
     test_X = test_x.reshape(test_x.shape[0], test_x.shape[1], test_x.shape[2],1)
     print("New shape of train data: ", test_X.shape)

New shape of train data:  (297960, 28, 28, 1)
New shape of train data:  (74490, 28, 28, 1)
```

```
#Here we convert the single float values to categorical values.
#This is done as the CNN model takes input of labels & generates
#the output as a vector of probabilities.
train_yOHE = to_categorical(train_y, num_classes = 26, dtype='int')
print("New shape of train labels: ", train_yOHE.shape)
test_yOHE = to_categorical(test_y, num_classes = 26, dtype='int')
print("New shape of test labels: ", test_yOHE.shape)

New shape of train labels:  (297960, 26)
New shape of test labels:  (74490, 26)
```

8. Installation of new packages for Tenserflow and CNN.

```
#Here we have to install all the new packages required for Tensflow and CNN
from __future__ import print_function
import tensorflow as tf
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Flatten
from tensorflow.keras.layers import Conv2D
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras import backend as k
import tensorflow

model = tf.keras.Sequential()
model.add(Conv2D(filters=32, kernel_size=(3, 3), activation='relu', input_shape=(28,28,1)))
model.add(MaxPooling2D(pool_size=(2, 2), strides=2))
model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu', padding = 'same'))
model.add(MaxPooling2D(pool_size=(2, 2), strides=2))
model.add(Conv2D(filters=128, kernel_size=(3, 3), activation='relu', padding = 'valid'))
model.add(MaxPooling2D(pool_size=(2, 2), strides=2))
model.add(Flatten())
model.add(Dense(64,activation ="relu"))
model.add(Dense(128,activation ="relu"))
model.add(Dense(26,activation ="softmax"))
```

9. Checking accuracy and saving the model.

```
[23] model.compile(optimizer = tensorflow.keras.optimizers.Adam(learning_rate=0.001), loss='categorical_crossentropy', metrics=['accuracy'])
     history = model.fit(train_X, train_yOHE, epochs=1,  validation_data = (test_X,test_yOHE))

9312/9312 [==============================] - 408s 44ms/step - loss: 0.1647 - accuracy: 0.9545 - val_loss: 0.0852 - val_accuracy: 0.9772
```

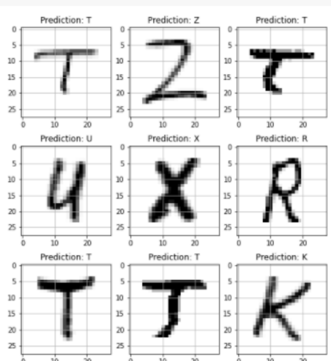
```
model.summary()
model.save(r'model_hand.h5')
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 32)        320

 max_pooling2d (MaxPooling2D  (None, 13, 13, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 13, 13, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 6, 6, 64)         0
 2D)

 conv2d_2 (Conv2D)           (None, 4, 4, 128)         73856

 max_pooling2d_2 (MaxPooling  (None, 2, 2, 128)        0
 2D)

 flatten (Flatten)           (None, 512)               0

 dense (Dense)               (None, 64)                32832

 dense_1 (Dense)             (None, 128)               8320

 dense_2 (Dense)             (None, 26)                3354

=================================================================
Total params: 137,178
Trainable params: 137,178
Non-trainable params: 0
```

```
print("The validation accuracy is :", history.history['val_accuracy'])
print("The training accuracy is :", history.history['accuracy'])
print("The validation loss is :", history.history['val_loss'])
print("The training loss is :", history.history['loss'])

The validation accuracy is : [0.9772318601608276]
The training accuracy is : [0.9545341730117798]
The validation loss is : [0.08515994995832443]
The training loss is : [0.16473297774791718]
```

10. Getting the resulting 9 images by prediction in the form of matrix.

```
fig, axes = plt.subplots(3,3, figsize=(8,9))
axes = axes.flatten()
for i,ax in enumerate(axes):
    img = np.reshape(test_X[i], (28,28))
    ax.imshow(img, cmap="Greys")

    pred = word_dict[np.argmax(test_yOHE[i])]
    ax.set_title("Prediction: "+pred)
    ax.grid()
```



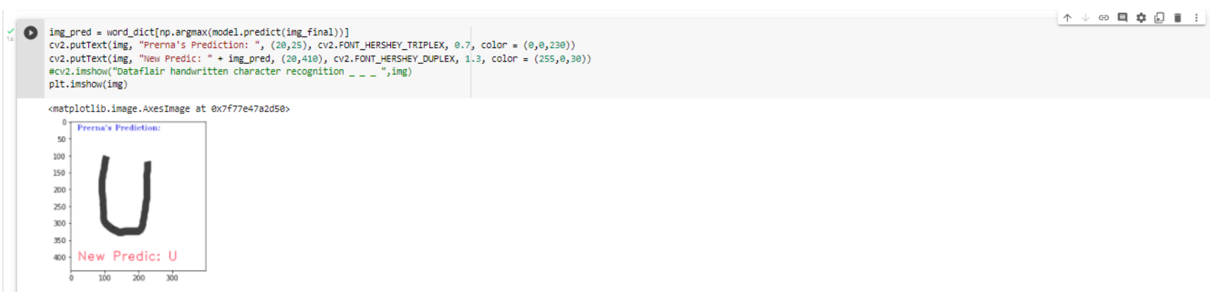
11. Uploading a jpeg file for checking the prediction of the model.

```
img = cv2.imread(r'prerna11.jpg')
img_copy = img.copy()
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
img = cv2.resize(img, (400,440))
plt.imshow(img)
```

```
<matplotlib.image.AxesImage at 0x7f77e3a07ad0>
```

```
#Gaussain Method used for Blur checking
img_copy = cv2.GaussianBlur(img_copy, (7,7), 0)
img_gray = cv2.cvtColor(img_copy, cv2.COLOR_BGR2GRAY)
_, img_thresh = cv2.threshold(img_gray, 100, 255, cv2.THRESH_BINARY_INV)
img_final = cv2.resize(img_thresh, (28,28))
img_final =np.reshape(img_final, (1,28,28,1))
```

12. Used Gaussian method for blur checking and presenting the final prediction.



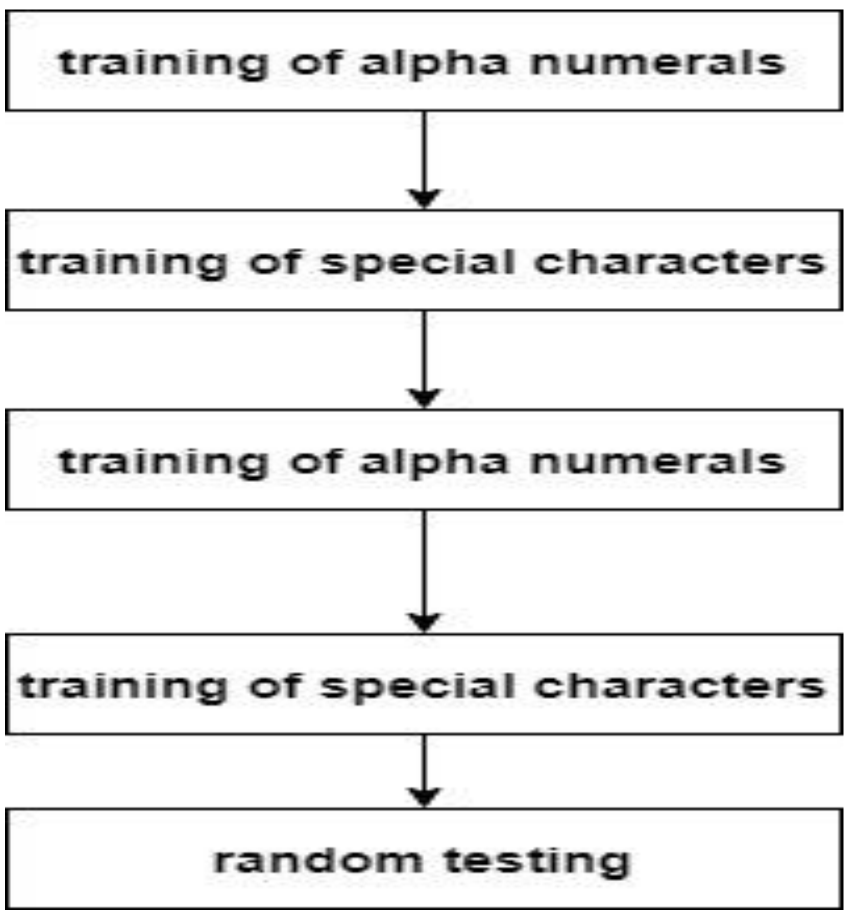
```
img_pred = word_dict[np.argmax(model.predict(img_final))]
cv2.putText(img, "Prerna's Prediction: ", (20,25), cv2.FONT_HERSHEY_TRIPLEX, 0.7, color = (0,0,230))
cv2.putText(img, "New Predic: " + img_pred, (20,410), cv2.FONT_HERSHEY_DUPLEX, 1.3, color = (255,0,30))
#cv2.imshow("Dataflair handwritten character recognition _ _ _ ",img)
plt.imshow(img)
```

```
<matplotlib.image.AxesImage at 0x7f77e47a2d50>
```


13. Used the below images for predictions and checking the model.

Training

In this part, we present the consequences of the organizations prepared on every one of the accessible information, in all dialects. We will allude to these organizations as Generic models, since they are not explicit to a language. We analyze the four organizations (Accurate, Fast, Fast Small and Faster Small) to the past form of A2iA Text Reader. AI system We prepared the organization to limit the Connectionist Temporal Classification [20] objective. We played out the advancement with stochastic angle plunge, utilizing the RMS Prop strategy with a base learning pace of 0.0004 and smaller than normal bunches of 8 models.



4. Review and Discussion

The CNN algorithm was implemented in MATLAB R2015a under Windows 7 operating system. The implemented program was run on intel core i7-2640 CPU with 4GB RAM. The test results are given in Table 1 below. The table provides three columns – in which the first column provides number of training images; the second column presents the number of testing sets and the last column lists the accuracy obtained by the CNN method for correctly classified images. It can be noticed from the experiment that the average accuracy increases with higher number of training images, since the higher number of images in the training produces more accurate information on the training parameters, which subsequently improves the accuracy in classification during testing phase. One checks that the accuracy obtained from 200 training images as 65.32% is improved gradually with increasing training images. The accuracy reaches to 92.91% with the 1000 training images. Thus, further increment of training images will continue to enhance the accuracy towards to certain limit – which cannot be exceeded due to numerical errors, and the constraints on the CNN capability of image differentiability for labels.

No. of Training Images	No. of Testing Images	Average Accuracy(%)
200	200	65.32%
300	200	74.43%
500	200	80.84%
600	200	85.21%
800	200	87.65%
1000	200	92.92%

5. CONCLUSION AND FUTURE SCOPE

CONCLUSION

In this project classification of characters takes place. The project is achieved through the conventional neural network. The accuracy we obtained in this is above 90.3%. This algorithm will provide both the efficiency and effective result for the recognition. The project gives best accuracy for the text which has less noise. The accuracy completely depending on the dataset if we increase the data, we can get more accuracy. If we try to avoid cursive writing then also it gives best results.

FUTURE SCOPE

In future we are planning to extend this study to a larger extent where different embedding models can be considered on large variety of the datasets. The future is completely based on technology no one will use the paper and pen for writing. In that scenario they used write on touch pads so the inbuilt software which can automatically detects text which they writing and convert into digital text so that the searching and understanding very much simplified.

6. REFERENCES

1. G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, “Emnist: an extension of mnist to handwritten letters,” arXiv preprint arXiv:1702.05373, 2017.

2. L. Eikvil, “Optical character recognition,” citeseer. ist. psu. edu/142042. html, 1993.

3. Graves and J. Schmidhuber, “Offline handwriting recognition with multidimensional recurrent neural networks,” in Advances in neural information processing systems, 2009, pp. 545–552.

4. Pham, T. Bluche, C. Kermorvant, and J. Louradour, “Dropout improves recurrent neural networks for handwriting recognition,” in Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on. IEEE, 2014, pp. 285–290.

5. S. Espana-Boquera, M. J. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martinez, “Improving offline handwritten text recognition with hybrid hmm/ann models,” IEEE transactions on pattern analysis and machine intelligence,

6. Fischer, A., Frinken, V., Bunke, H.: Hidden Markov models for off-line cursive handwriting recognition, in C.R. Rao (ed.): Handbook of Statistics 31, 421 – 442, Elsevier, 2013

7. Gu J, Wang Z, Kuen J, Ma L, Shahroudy A, Shuai B, Liu T, Wang X, Wang G, Cai J and Chen T 2018 Patt. Recognition 77 354