# Kathmandu University

## Department of Computer Science and Engineering

## Dhulikhel, Kavre

Mini-Project Report on

**“Optical Character Recognition”**

(For the partial fulfillment of Machine Learning)

**Submitted By:**

Amir Manandhar (21)

Saurav Maskey (22)

Subash Hada (56)

**Submitted To:**

Dr. Bal Krishna Bal

Department of Computer Science & Engineering

Submission Date: 11th March 2020

# Introduction

Optical Character Recognition (OCR) is the ability of a computer to receive and interpret intelligible handwritten input from sources such as paper documents, photographs, touchscreens and other devices. OCR is a widely used common method for digitalizing printed texts so that they can be electronically edited, searched, and stores more compactly. It is used in machine processes such as machine translation, text-to-speech and text mining.

## Problem Definition

The major problem with OCR (handwritten recognition) is that the handwritten digits are not always of the same (in size, width, orientation and justified to margins) as they differ from writing of a person to another. So the general problem would be while classifying the digits and characters due to the similarity between digits such as 1 and 7, 3 and 8, between letters such as a and o, i and l etc. Lastly, the uniqueness and variety in the handwriting of different individuals also influence the appearance of the digits.

This is why we introduce the concepts and algorithms of deep learning and machine learning to develop handwritten recognition (also called OCR) systems.

## Motivations for Doing the Project

We have several motivations to carry out this project.

OCR systems have not changed much in the past decade. The existing OCR systems don't do a particularly good job in recognizing the characters from handwritten input sources. A major motivation is to at least build a working model for OCR. A functional OCR would make it much easier in machine translation and processing of written inputs such as paper documents, images of such documents, which ultimately helps in easier editing, searching, and storing of the data.

## Objectives of the Project work

The major objectives of our Project work are:

- Build a working model
- To predict the words from the input images
- To learn about different neural networks
- To learn about training and testing of data

## Related works

Since a very long time, human used to write their thoughts in the form of letter, transcripts etc. in order to convey them to others. But since the development of computer technology the format of handwritten text changed rapidly to computer generated digital text and so people feel a need of such method that can transform the handwritten text to digital text as it makes the processing of such data fast and easy.

Rajib proposed a handwritten English character recognition system based on the Hidden Markov Model (HMM). This method made use of two different feature extractions (global and local feature extraction). Global feature include gradient features, projection features and curvature features in the numbers of four, six and four respectively. Whereas local features are calculated by dividing the sample image into nine equal blocks. This resulted into fifty features (local + global) for each sample image. Data post processing is also utilized by this method in order to decrease the cross classification of different classes.

Anshul Mehta proposed their work based on the heuristic segmentation algorithm. Their system performs identification of valid segmentation points between handwritten letters quite well. This method tried to provide classification of total 52 characters (26 upper case English letters and 26 lower case English letters).

T. Som used fuzzy membership function to improve the accuracy of hand written text recognition system. In this method, text images are normalized to $20 \times 10$ pixels and then fuzzy approach is used to each class. Bonding box is created around the character in order to determine the vertical and horizontal projection of the text. The images are cropped to a bounding box, and it is re-scaled to the size of $10\times10$ pixels. The method was fast but it provides a low accuracy.

## Datasets

In this project, we train and test a set of classifiers on the EMNIST database for pattern analysis in solving the handwritten character recognition problem.

EMNIST Dataset is an extension to the original MNIST dataset to also include letters. The EMNIST dataset is a set of handwritten character digits derived from the NIST Special Database 19 and converted to a 28x28 pixel image format and dataset structure that directly matches the MNIST dataset.

There are six different splits provided in this dataset. We have used EMNIST Balanced dataset. It consist of the following sets:

- Train: 112,800
- Test: 18,800
- Total: 131,600 (including Classes: 47 (balanced))

## Methods and Algorithms Used

Keras library is used in this project. The methods used are:

1. Loading the EMNIST Dataset: The first and most important step in any machine learning task is to prepare the data. EMNIST is already a python package which provides functionality to automatically download and cache the dataset, and to load it as Numpy arrays.
2. Preprocessing the input data/clean data:
   - 2.1. Flattening the images: The training dataset is structured as a 3-dimensional array of instance, image width and image height. For a multi-layer perceptron model we must reduce the images down into a vector of pixels. We need to reshape each image to the format of (M x N x 1). We'll use the reshape () method to perform this action. Finally, we normalize the image data by dividing each pixel value by 255 (since RGB value can range from 0 to 255).
   - 2.2. Normalize inputs from 0 to 1: We normalize the image data by dividing each pixel value by 255 (since RGB value can range from 0 to 255). We then create the model and train it using the testing data.
3. Define model architecture:

   The model is a simple neural network with one hidden layer with the same number of neurons as there are inputs (784). A rectifier activation function is used for the neurons in the hidden layer. A softmax activation function is used to turn the outputs into probability-like values and allow one class of the 10 to be selected as the model's output prediction.

4. Compiling the model:

   Logarithmic loss is used as the loss function (called categorical crossentropy in Keras) and the efficient ADAM gradient descent algorithm is used to learn the weights.

5. Fitting the model on training data:

   To fit the model, the batch size and number of epoch is declared. After fitting the model is saved so that it doesn't have to be trained each and every time.

## Experiments

The images are given as input. The images are then resized to find the space better. Using OpenCV library the images are changed into grayscale, then dilated and Gaussian blur are added so that the images could be separated easily and are more readable. After that, the images are separated into letters in different bounding boxes using Contours and ROI. Finally, each images are fed into the model and the output is obtained.

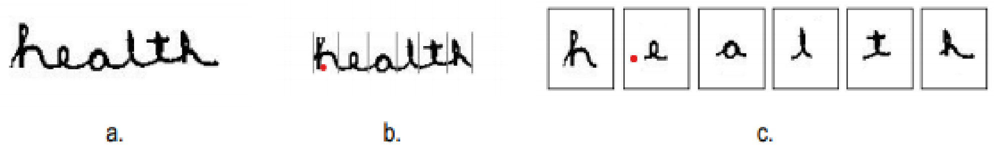
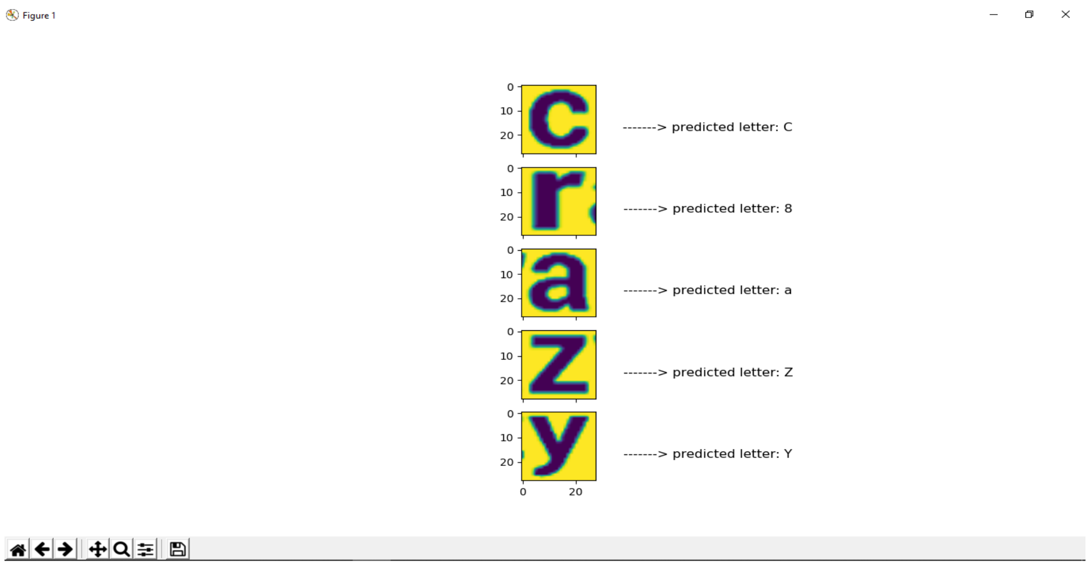


Figure 1: Segmentation of input image



```
-------> predicted letter: C
-------> predicted letter: 8
-------> predicted letter: a
-------> predicted letter: Z
-------> predicted letter: Y
```

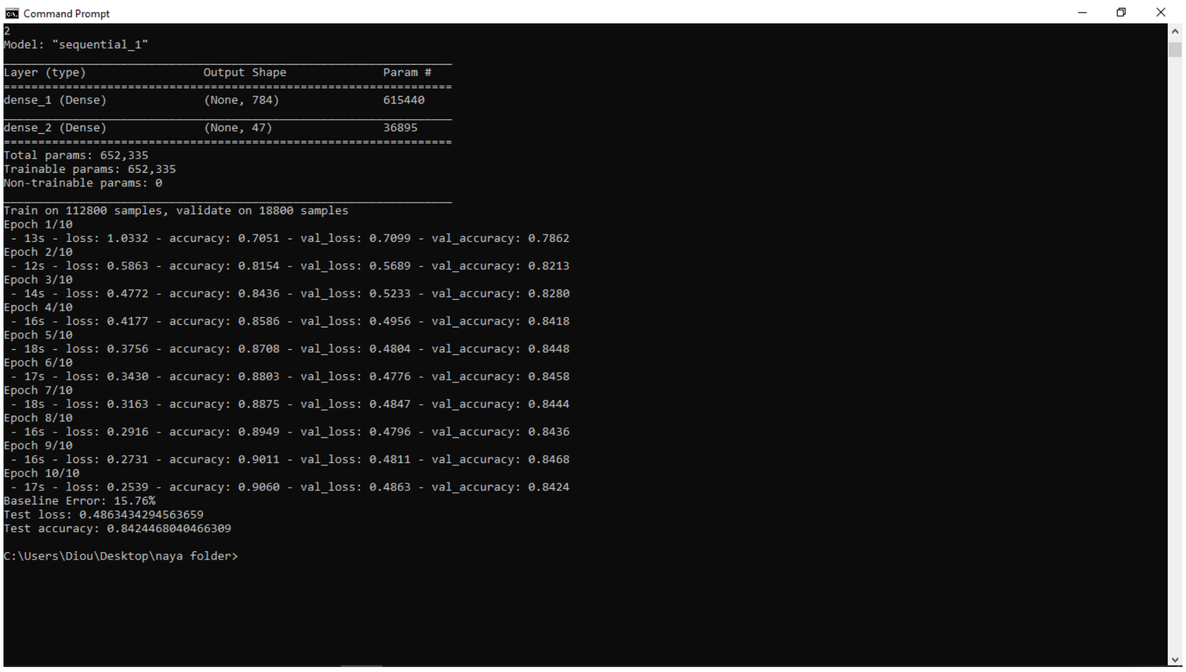
Figure 2: Output of our model

## Evaluation of Results

Logarithmic loss is used as the loss function (called categorical crossentropy in Keras).

The test-loss of the model is 0.486.

The test-accuracy of the model is 0.842.



```
2
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 784)               615440
_________________________________________________________________
dense_2 (Dense)              (None, 47)                36895
=================================================================
Total params: 652,335
Trainable params: 652,335
Non-trainable params: 0
_________________________________________________________________
Train on 112800 samples, validate on 18800 samples
Epoch 1/10
 - 13s - loss: 1.0332 - accuracy: 0.7051 - val_loss: 0.7099 - val_accuracy: 0.7862
Epoch 2/10
 - 12s - loss: 0.5863 - accuracy: 0.8154 - val_loss: 0.5689 - val_accuracy: 0.8213
Epoch 3/10
 - 14s - loss: 0.4772 - accuracy: 0.8436 - val_loss: 0.5233 - val_accuracy: 0.8280
Epoch 4/10
 - 16s - loss: 0.4177 - accuracy: 0.8586 - val_loss: 0.4956 - val_accuracy: 0.8418
Epoch 5/10
 - 18s - loss: 0.3756 - accuracy: 0.8708 - val_loss: 0.4804 - val_accuracy: 0.8448
Epoch 6/10
 - 17s - loss: 0.3430 - accuracy: 0.8803 - val_loss: 0.4776 - val_accuracy: 0.8458
Epoch 7/10
 - 18s - loss: 0.3163 - accuracy: 0.8875 - val_loss: 0.4847 - val_accuracy: 0.8444
Epoch 8/10
 - 16s - loss: 0.2916 - accuracy: 0.8949 - val_loss: 0.4796 - val_accuracy: 0.8436
Epoch 9/10
 - 16s - loss: 0.2731 - accuracy: 0.9011 - val_loss: 0.4811 - val_accuracy: 0.8468
Epoch 10/10
 - 17s - loss: 0.2539 - accuracy: 0.9060 - val_loss: 0.4863 - val_accuracy: 0.8424
Baseline Error: 15.76%
Test loss: 0.4863434294563659
Test accuracy: 0.8424468040466309

C:\Users\Diou\Desktop\naya folder>
```

Figure: Training of the data (test for accuracy and loss)

## Discussion on Results

In this project, a neural network in Keras was created to classify the handwritten words.

The accuracy of the model is 0.842. Still there are some opportunities for improvement.

Although the result was quiet good, but it could have been much better.

The following can be done for improvement and better results:

- Increasing the size of datasets
- Increasing the number of layers in the model
- Removing noise from the images
- Use of multiple algorithms

## Contributions of each group member

The contributions of all the group members is mentioned in the table below.

| Team Members | Contribution and work division |
|---|---|
| Amir Manandhar | Research about the dataset and neural networks.<br>Preprocessing the data. |
| Subash Hada | Model training using training dataset.<br>Testing of the dataset and accuracy calculation. |
| Saurav Maskey | Segmentation of the input image and feeding to the model. |

## Conclusion and Future Extensions to the Project

Finally, a simple neural network was created using Keras which predict the words given in an image. Most of the objectives were met during the process. We got to learn much about machine learning and different neural networks. We learned about training and testing of the data. We also learned how to classify custom handwritten words which were not a part of our dataset.

As future extensions to the project, we will improve the accuracy of our model by adding more dataset. Our current dataset contains only 131,600 images. We plan on adding a dataset containing about 6 lacs images in total. Also we will try to change our model and create a total of 5 layers which would increase our accuracy significantly.