Class Assignment (CA) 1 Report

on

DIGIT CLASSIFICATION USING DEEP LEARNING

Submitted in partial fulfillment of the requirements for the award of degree

in the department of

Computer Science and Engineering

LOVELY PROFESSIONAL UNIVERSITY PHAGWARA, PUNJAB

Submitted by:

Priyanshu Saxena 11915058

K Murali Krishna 11907747

Bipasha Das 11904046

Section: KM048

Date of submission: 27 November 2021

Contents

S. No.	Topic name	Page No.
1	Abstract	3
2	Introduction	4
3	Importance of Handwritten Digit Recognition	5
4	Approach to the project	6 - 9
5	Hardware and Software Requirement	10
6	Main Goal and Application	11
7	Future Scope	12
8	Conclusion	13
9	References	14

ABSTRACT

Digit Classification has always been a major project of research, and this report consists the total working done by the group on this project using neural networks and deep learning. This project hasn't been just limited to the use of Jupyter Notebook and Algorithm Classification, the use of Tkinter library of Python has made it possible for us to make this project more user-intractable and easy to understand by the user.

The overall working is done on the mnist dataset that is found in Keras dataset. The MNIST database of handwritten digits, has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image.

The image size is of 28x28 pixels, which can be shown in the Jupyter Notebook by using the matplotlib library, and the image can be seen in both color-encoding and binary (rgb numbering). The proposed framework involves three primary parts, image pre-processing, feature extraction and classification. This study strives to improve the recognition accuracy by more than 99% in handwritten digit recognition. As will be seen, pre-processing and feature extraction play crucial roles in this experiment to reach the highest accuracy.

INTRODUCTION

This project aims to recognize the handwritten digits by using tools from Machine Learning and Deep Learning to train the classifiers, so it produces a high recognition performance. Furthermore, the use of tools from Computer Vision and Tkinter is explored to investigate the effect of the selection of classifiers, features, and image preprocessing on the entire error rate. The dataset used for the application is a MNIST dataset containing 60,000 training and 10,000 testing images originally, which are 28 x 28 grayscale (0-255), labelled and bitmap format. It is an excellent database for machine learning and pattern recognition methods while needing minimal efforts in preprocessing and formatting. There are many features in this data, so it has many dimensions.

THE IMPORTANCE OF HANDWRITTEN DIGIT RECOGNITION

More and more people are focusing on the use of the personal computer rather than acquiring excellent handwriting skills. The one reason is that the internet and applications are becoming more intelligent than before. Additionally, the poor quality or illegible handwriting is the main reason for inaccurate handwritten character recognition.

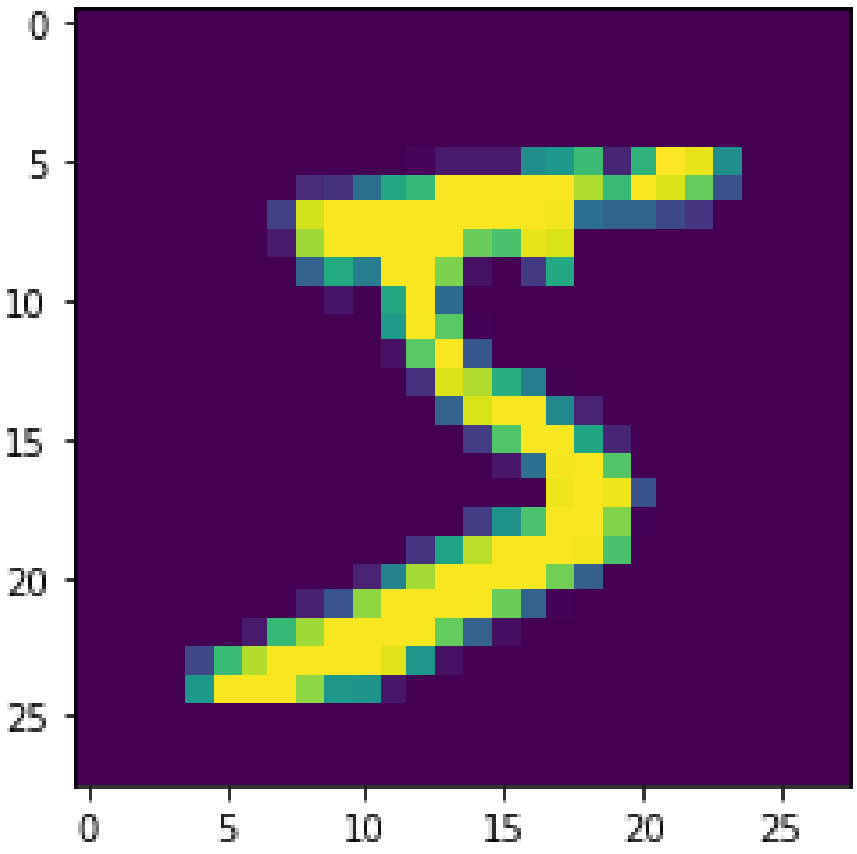
Typically, handwriting experts use sophisticated classification models to analyze printed or handwritten character images. As part of this process, they extract features from the samples which include slants, orientation and the center alignment of the letters. Offline digital recognition has many practical applications. For instance, the handwritten sample is analyzed and recognized by the handwriting expert to identify the zip code in an address written or printed on an envelope (Hanmandlu & Murthy, 2007). As a result, the benefits of applying this system at the post office are enormous. The system can realize the automatic sorting of millions of emails, thus reducing the human burden and speeding up the whole process (Mane & Kulkarni, 2018).

APPROACH TO THE PROJECT

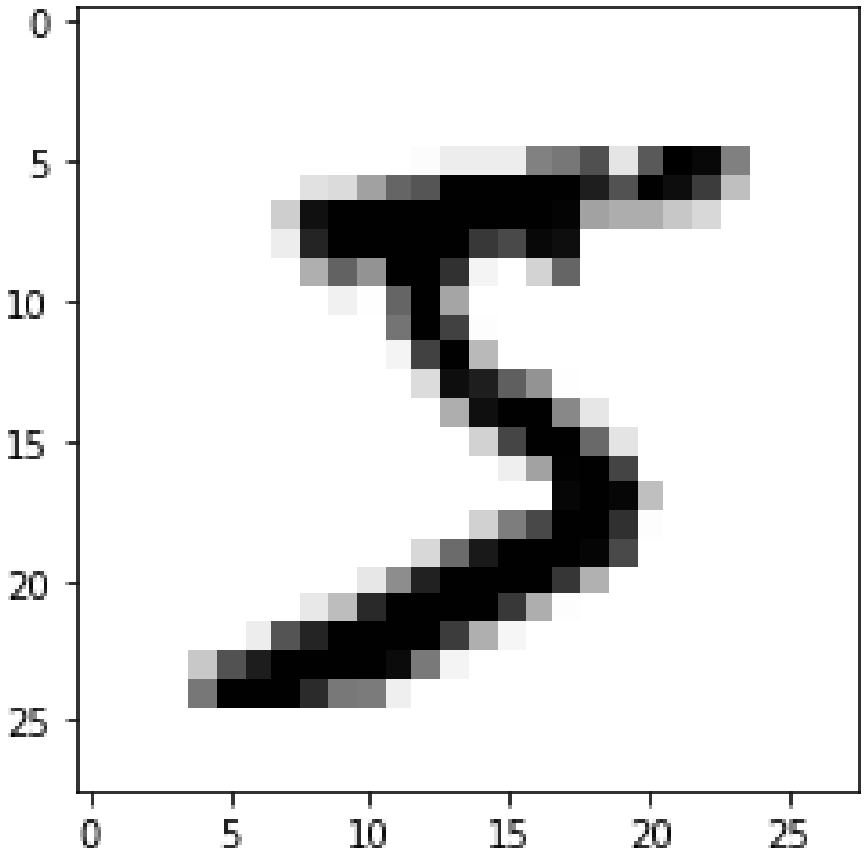
As it is known that the digit classification requires a dataset which is hand-written and not generated by computer, hence to get that dataset, we need Tensorflow library included with Keras which further include a bunch of datasets from which we'll be using MNIST dataset.

MNIST dataset contains 60k training samples and 10k testing samples, and it contains 28x28 pixels of images purely hand-written. Python doesn't have any library to directly show these images, hence we'll be taking help from the matplotlib library which will plot all the 28x28 pixels on a graph.

When plotted, the graph will look something like this:



This image is of number 5, plotted on a 28 by 28 graph. But we don't require a color encoded graph, for training purpose, we'll be converting this image into binary format, i.e., in 0-255 form. After conversion it will look something like this:



The images are in 0-255 form, which can be hectic for the deep learning model to process, hence we normalize it, and by normalization we mean that we convert all the values to lie in between 0 and 1. To do this we divide all the values by 255, which give us all the values in the range of 0 to 1. After normalization, we need to process, we convert the 28 by 28 grid into a 784 single layer grid, which will be fed to the 784 input neurons.

The input layers would be having 784 neurons, and we can choose the hidden layers as per our choice, and for this we are choosing 2 layers. And for the output layer, we take the number of neurons as per the number of digits that is 10, ranging from 0 to 9.

Now as the model has been made, we start testing it on the testing dataset as per the set epochs. The accuracy differs in between 97-99%. After the testing on the test dataset, we can use it on real-

time handwritten digits, and for this we require the use of open-cv, and open-cv will read the image from the system and convert it to the data as per requirement, which will in turn help us to predict the written digit. The digits for which the model was checked were 1, 2, 3, 5, 7 and 8. The model predicted wrong for the digit 2 but stood out right for the remaining digits.

The two hidden layers use relu as the activation function while the output layer uses softmax as the activation function.

The code has been shared below:

```
import tensorflow as tf
import tensorflow.keras as keras
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np
import cv2 as cv
```

```
(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()
```

```
x_train = keras.utils.normalize(x_train, axis=1)
x_test = keras.utils.normalize(x_test, axis=1)
```

```
model = keras.models.Sequential()
model.add(keras.layers.Flatten(input_shape = (28, 28)))
```

```
model.add(keras.layers.Dense(units=128, activation=tf.nn.relu))
model.add(keras.layers.Dense(units=128, activation=tf.nn.relu))
```

```
model.add(keras.layers.Dense(units=10, activation=tf.nn.softmax))
```

```
model.compile(optimizer='adam',loss='sparse_categorical_crossentropy',metrics=['accuracy'])
```

```
model.fit(x_train, y_train, epochs=5)
```

```
loss, accuracy = model.evaluate(x_test, y_test)
print("Loss: ", loss)
print("Accuracy: ", accuracy)
```

```
for x in range(1,7):
    img = cv.imread(f'{x}.png')[:,:,0]
    img = np.array([img])
```

```
prediction = model.predict(img)
print(f'The predicted result is: {np.argmax(prediction)}')
plt.imshow(img[0], cmap=plt.cm.binary)
plt.show()
```

HARDWARE AND SOFTWARE REQUIREMENTS

SOFTWARE REQUIREMENT

- Jupyter Notebook

HARDWARE REQUIREMENT

- CPU: 2 x 64-bit 2.8 GHz 8.00 GT/s CPUs
- RAM: 32 GB (or 16 GB of 1600 MHz DDR3 RAM)
- Storage: 300 GB.
- Additional space recommended if the repository will be used to store packages built by the customer. With an empty repository, a base install requires 2 GB.
- Internet access to download the files from Anaconda.org or a USB drive containing all of the files you need with alternate instructions for air gapped installations.

MAIN GOAL AND APPLICATION

- Handwritten Digit Recognition is used to recognize the Digits which are written by hand.
- A handwritten digit recognition system is used to visualize artificial neural networks.
- It is already widely used the automatic processing of bank cheques, postal addresses, in mobile phones etc.
- The handwritten digit recognition using convolutional neural network has proved to be of a fairly good efficiency.
- It works better than any other algorithm, including artificial neural networks.

FUTURE SCOPE

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

CONCLUSION

In this report, three approaches of neural network have been presented to recognize the handwritten digits. Among the all three approaches Back-propagation is the most successful in the recognition process. Back-propagation is fast and efficient as compared to the other two approaches and gives quick convergence on satisfactory local minima in case of error. It has a simple implementation and computing time is reduced if the weights chosen are small at the beginning. In future we are expecting to explore the Back-propagation algorithm to make recognition of digits more fast and efficient and improve the overall performance.

REFERENCES

- Google
- Kaggle
- YouTube
- StackOverFlow
- GitHub