

**FACULTY OF SCIENCE**

**SCHOOL OF COMPUTER SCIENCE**

**2020-2021**

**A**

**PROJECT REPORT**

**ON**

**Handwritten Digit** **Recognizer**

**BY**

**Prabhakar Maity Hrishikesh Kulkarni**

**Deepak Pawade**

**FOR MINI PROJECT – TRIMESTER VII OF**

**BACHELOR OF SCIENCE (COMPUTER SCIENCE)**

**MIT WORLD PEACE UNIVERSITY**



**SCHOOL OF COMPUTER SCIENCE**

Certificate

This is to certify that**, PRABHAKAR MAITY, HRISHIKESH KULKARANI, DEEPAK PAWADE, AKSHAY AGARWAL** studentsof B.Sc.(Computer Science) in **Trimester** **VII** has successfully / partially completed Industrial Training at **MINI PROJECT** at **MITWPU** in partial fulfilment of B.Sc. Computer Science under MIT World Peace University, for the academic year 2020-2021.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Internal Guide Head of the School

Date: \_\_\_/\_\_\_/\_\_\_\_\_\_

External Examiners:

1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ACKNOWLEDGEMENT:**

**DECLARATION:**

**Pune** **Signature of the Candidate**

**Date:**

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Contents** | **Page No.** |
| **Chapter 1** | **INTRODUCTION** |  |
| 1.1 Existing System |  |
| 1.2 Problem Definition- Need of Computerization |  |
| **Chapter 2** | **PROPOSED SYSTEM** |  |
| 2.1 Proposed System |  |
| 2.2 Objective of the System |  |
| 2.3 User Requirements |  |
| 2.4 Operating Environment – Hardware and Software |  |
|  |  |
| **Chapter 3** | **ANALYSIS AND DESIGN** |  |
| Module List |  |
| 3.1 Training and Testing the model |  |
| 3.2 System diagrams (DFDs) |  |
| 3.4 Screen Shots of GUI |  |
| 3.5 Tools and Requirements |  |
| **Chapter 4** | **USER MANUAL** |  |
| 4.1 User Manual |  |
|  |  |
| **Chapter 5** | **CONCLUSION** |  |
| 5.1 Future Enhancement |  |
| 5.2 Disadvantages |  |
| 5.3 Conclusion |  |

Introduction

**Existing System**

* Handwriting recognition of characters has been around since the 1980s. The task of handwritten digit recognition, using a classifier, has extraordinary significance and use such as online digit recognition on PC tablets, recognize zip codes on mail, etc.
* An ever-increasing number of individuals use pictures to transmit data. It is additionally main-stream to separate critical data from pictures.
* Image Recognition is an imperative research area for its generally used applications.
* In general, the field of pattern recognition, one of the difficult undertakings, is the precise computerized recognition of human handwriting.
* This is a very difficult issue because there is an extensive diversity in handwriting from an individual to another individual.
* This difference does not make any issues to people, yet, anyway it is increasingly hard to instruct computers to interpret general handwriting.

**Problem Definition: Need of Computerization**

* 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.
* Digit recognition system is the working of a machine to train itself or recognizing the digits from different sources like emails, bank cheque, papers, images, etc. and in different real-world scenarios for online handwriting recognition on computer tablets or system, recognize number plates of vehicles, processing bank cheque amounts, numeric entries in forms filled up by hand (say — tax forms) and so on.
* The digit recognition model uses large datasets in order to recognize digits from distinctive sources.

**Proposed System**

* The system will be trained to recognize the handwritten digit drawn by the user on a canvas built within an interactive window.
* We are going to implement a handwritten digit recognition app using the MNIST dataset.
* We will be using a special type of deep neural network that is **Convolutional Neural Networks**.

Objectives

* To achieve an accuracy of more than 90% while recognizing the digits drawn by the user.
* To introduce different data sets to train the model to achieve other than MINST dataset to push the accuracy even further.

User Requirements

* Windows 10 enabled PC with a mouse to draw over the canvas.
* Certain python libraries mentioned below to be able to run the GUI.

Operating Environment – Software & Hardware

(minimum requirements)

**Software:**

* Windows 10
* Python 3.6.5
* Keras, KTinker, Pillow, TensorFlow, pywin32, win32gui.

**Hardware:**

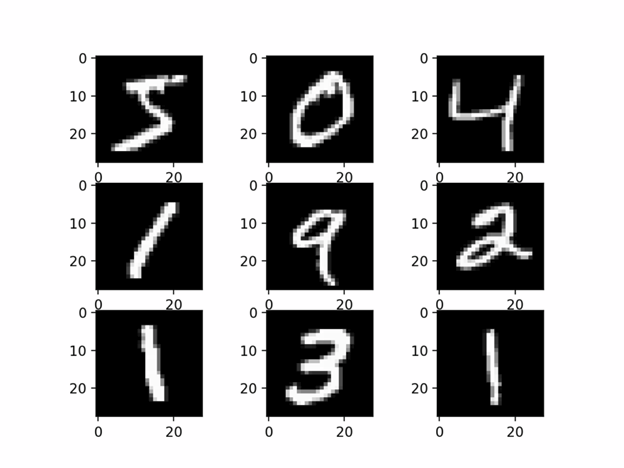
* Processor: Any x32 or x64 CPU with 2 or more cores.
* Primary Memory: 4GB
* Secondary Memory: 15GB

**Analysis & Design**

The project involves training and testing the model using MINST dataset and making a GUI that uses the trained model to identify the drawn digit by the user over the canvas.

Training

### The MNIST dataset:

****

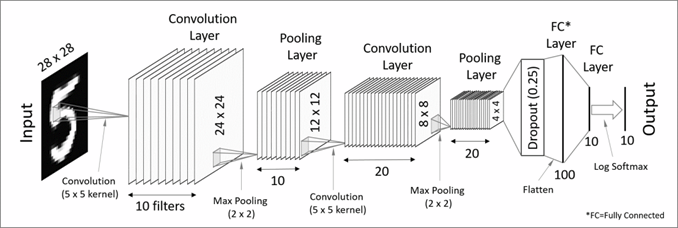
The [MNIST dataset](http://yann.lecun.com/exdb/mnist/) contains 60,000 training images of handwritten digits from zero to nine and 10,000 images for testing. So, the MNIST dataset has 10 different classes. The handwritten digits images are represented as a 28×28 matrix where each cell contains grayscale pixel value.

Preprocessing:

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)

Creating the model:

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 offer fitting of the model. We will then compile the model with the Adadelta optimizer.

****

***CNN architecture in MNIST dataset***

Algorithm:

### To see the performing steps for a system to predict, we can define algorithms as –

### Break the image into small image tiles — Similar to sliding window, we can pass sliding window over the entire large image and each result is saved as separate, as a segment of large image as tiny picture tile.

### Feeding each tiny tile into the smaller size neural network — we rarely initialize the parameters with the same values and if not so, then we mark that tile as interesting.

### Save the results from each small tile into a new array — we would not like to misplace the index of the original file. So we place the results in a grid of the same arrangement as an original image.

### Down-sampling — to reduce the size of a newer array, down-sampling is used by max-pooling.

### 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.**

Testing

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 around 99% accuracy.

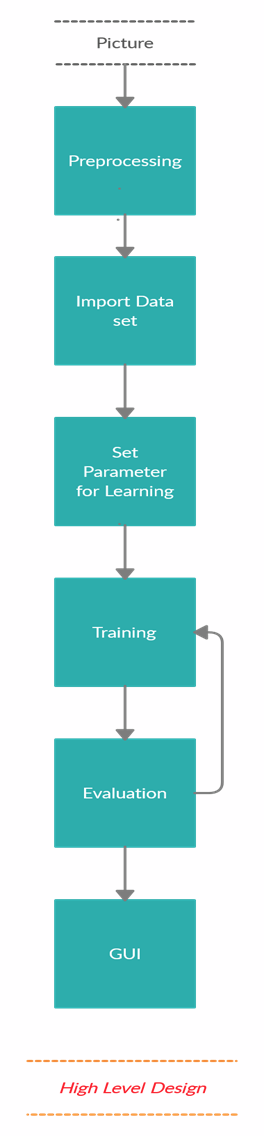
### 

### Create GUI to predict digits

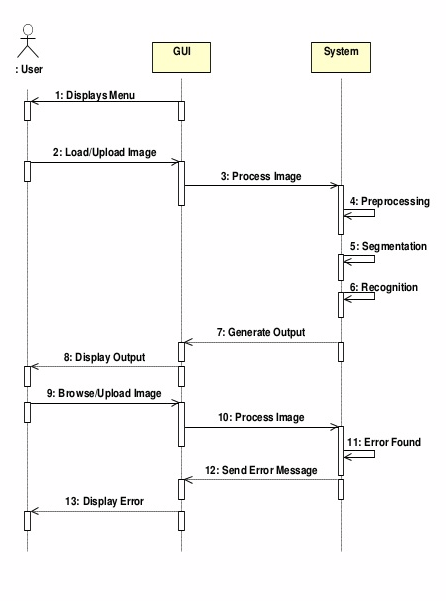
we build an interactive window to draw digits on canvas and with a button, we can recognize the digit. The Tkinter library comes in the Python standard library. We have created a function predict\_digit() that takes the image as input and then uses the trained model to predict the digit.

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.

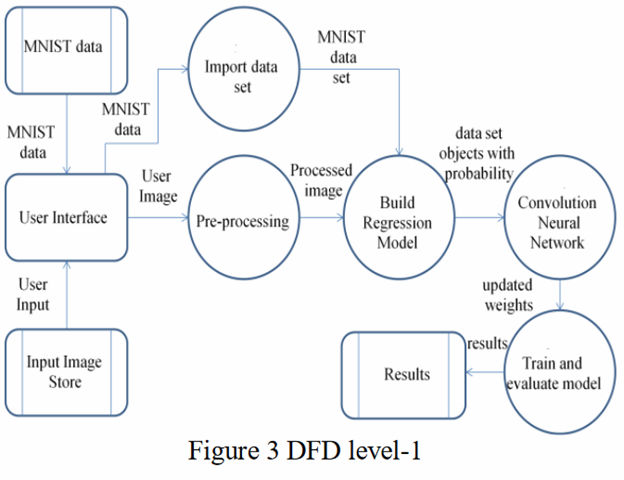
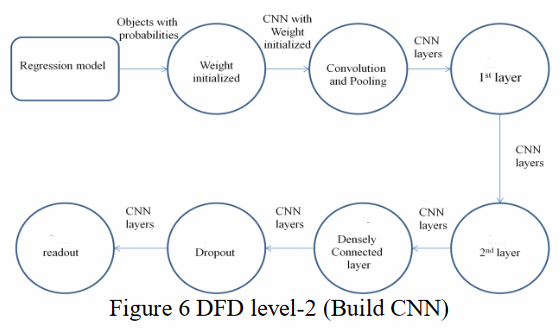
System Diagrams

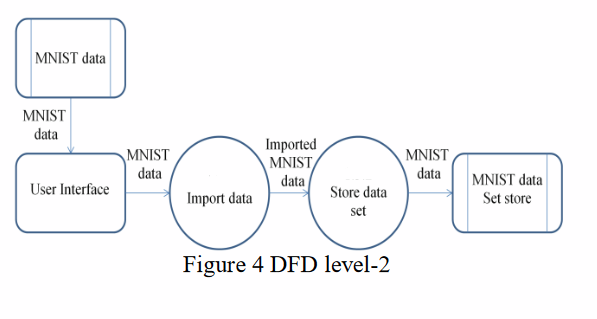
****

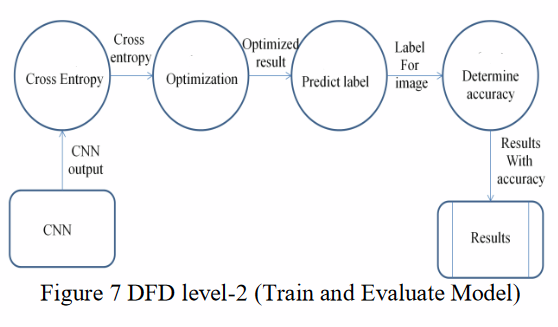
High Level Diagram



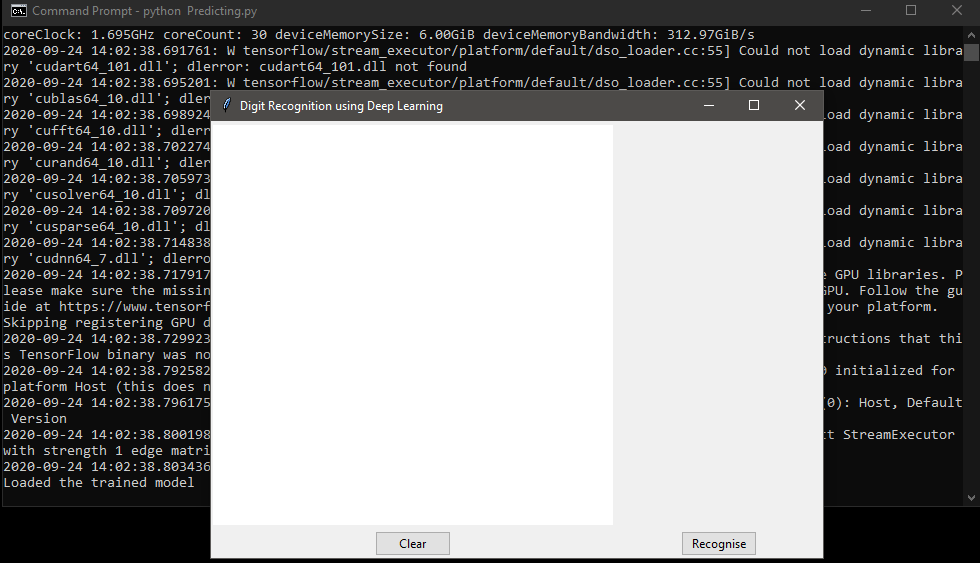
Sequence Diagram

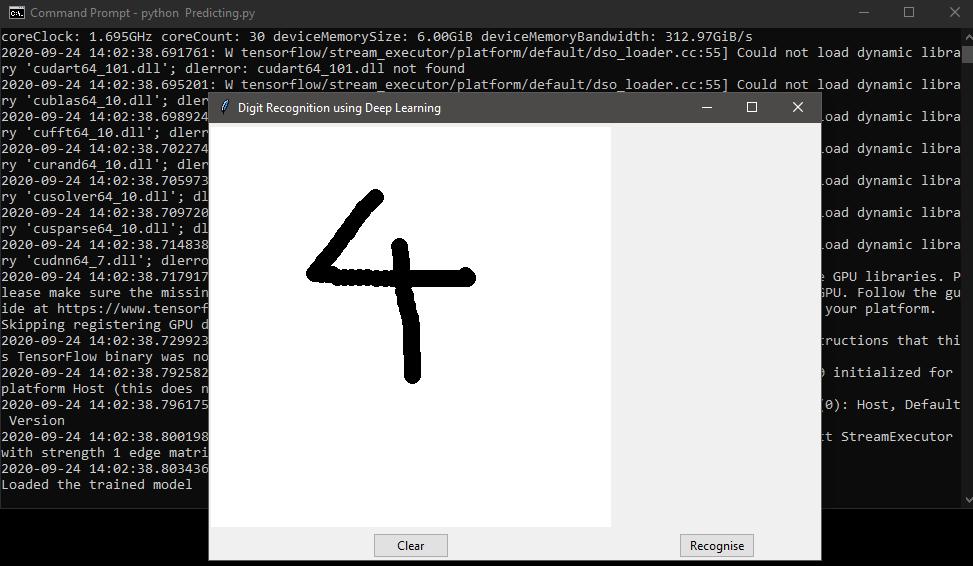
****

****

****

Screenshots of the GUI





Resources and Tools required/Used

Software:

* **Python**(Programming Language)
* **IDE**: IDLE/Spyder/Jupyter
* **win32gui** :Python extensions for Microsoft Windows’ Provides access to much of the Win32 API, the ability to create and use COM objects, and the Pythonwin environment.
* **pywin32**: dependency for win32gui.
* **Pillow**: image managing library.
* **Keras**: Keras contains numerous implementations of commonly used neural-network building blocks
* **TensorFlow** is a [free](https://en.wikipedia.org/wiki/Free_software) and [open-source](https://en.wikipedia.org/wiki/Open-source_software) [software library](https://en.wikipedia.org/wiki/Library_(computing)) for [dataflow](https://en.wikipedia.org/wiki/Dataflow_programming) and [differentiable programming](https://en.wikipedia.org/wiki/Differentiable_programming) across a range of tasks.
* **Windows 10 x64**

Hardware:

* CPU : AMD Ryzen 5 2600
* GPU : Nvida Geforce RTX 2060
* RAM : 16GB 3000MHz DDR4
* Secondary Storage: 240 GB SSD

User Manual

|  |  |  |
| --- | --- | --- |
| Version | features | ETA |
| 1.0 | Digit recognizer | ??/10/2020 |
| 2.0 | Expression recognizer | ??/12/2020 |

1.Releases

2.Instructions:

1. Using mouse or trackpad, draw the digit on the canvas of the GUI.
2. After drawing, clicking on the ‘Recognize’ button will display the result along with the probability that it guessed with.
3. Click ‘Reset’ button to reset the canvas.

3.Q&A:

* Can the accuracy be increased ?

Yes, by training the model on more unknown data sets, the accuracy can be increased. But, finding more data sets is challenging.

Conclusion

**Future Enhancements**

1. Train the machine to be able to recognize simple equation/expressions\*\*.

2. Extend the model to work on NIST dataset.

3. Increase the accuracy further by implementing more number of hidden layers and/or epochs

4. Develop a similar system to identify alphabets.

(\*\*Though it requires a model of itself to segregate each character and is kind of simple than detection. But the main focus is digit detection and this approach is to be considered after the desired accuracy has been achieved in digit recognition.)

**Disadvantages**

The Handwritten Digit Recognizer system is not 100% accurate and it never could be. The state of the art similar system is still 99% accurate.

**Final Conclusion**

The ultimate aim of the system is to show the usefulness of Deep-learning in digitalization of undocumented data.

This technology proves to be a boon for us in increasing efficiency and decreasing time for various complex workloads.

References

Dataset:

*The MNIST Database of handwritten digits* - [Yann LeCun](http://yann.lecun.com), Courant Institute, NYU & [Corinna Cortes](http://homepage.mac.com/corinnacortes/), Google Labs, New York & [Christopher J.C. Burges](http://research.microsoft.com/en-us/people/cburges/), Microsoft Research, Redmond.-<http://yann.lecun.com/exdb/mnist/>

Guidance:

*Neural Networks and Deep Learning*, By [Michael Nielsen](http://michaelnielsen.org) / Dec 2019 -<http://neuralnetworksanddeeplearning.com/>

### *Deep Learning* - Ian Goodfellow and Yoshua Bengio and Aaron Courville-<http://www.deeplearningbook.org/>