



Dharmsinh Desai University, Nadiad

Faculty of Technology, Department of Computer Engineering

B.Tech. CE Semester – VI

Subject: System Design Project

Project Title:

Handwritten Digit Recognition

Submitted By:

Dhamecha Mayur CE027 17CEUBG023

Godhani Vivek CE036 17CEUOG113

Jakasaniya Dharm CE048 17CEUON043

Guided By:

Prof. Hariom A. Pandya

Assistant Professor, CE Dept.

Dharamsinh Desai UniverSity, Nadiad.



Dharmsinh Desai University, Nadiad

Faculty of Technology, Department of Computer Engineering

CERTIFICATE

This is to certify that System Design project entitled “Handwritten Digits Recognition” is the bonafied report of work carried out by

- 1) **Dhamecha Mayur** **CE027** **17CEUBG023**
- 2) **Godhani Vivek** **CE036** **17CEUOG113**
- 3) **Jakasaniya Dharm** **CE048** **17CEUON043**

Of Department of Computer Engineering, Semester VI, academic year 2020-2021,
under our supervision and guidance.

Guide

Prof. Hariom A. Pandya
Assistant Professor of
Department of Computer
Engineering, Dharmsinh Desai
University, Nadiad.

HOD

Dr. C. K. Bhensdadia
Head of the Department of
Department of Computer
Engineering, Dharmsinh Desai
University, Nadiad.

Content

1.Abstract.....	4
2.Introduction.....	5
3.software requirement specification.....	6
4.Design.....	7
4.1 Usecase diagram.....	7
4.2 Class diagram.....	8
4.3 Sequence diagram.....	9
4.4 Activity diagram.....	10
5. Tools/Technologies.....	12
6. Implementation Details.....	13
7. Testing.....	14
8. Screens shots	15
9. Conclusion	20
10. Limitation & future Implementation Details.....	21
10.1 Limitations.....	21
10.2 Future implementations.....	21
12. Bibliography.....	22

1. Abstract

Our Project “Handwritten Digits Recognition” provides the platform for user to he can identify the digits that written by hand. So for that user need to scan that photo and send to server and there is service which identify the digits and response to the user.

2. Introduction

This application can identify 0 to 9 digits. Therefore user need to write any number from 0 to 9 in paper after user need to install the application on his phone. After he could select that from gallery or capture it from camera. after he need to send it to our server which is created in local machine. Machine can recognize that image using Machine Learning. And Respond back to the user So user can see the result.

3. Software Requirement Specifications

1.) Functionality

R.1 : Upload Image

Description : User can upload the image for identify

Input : Image selection

Output : Image uploaded successfully

R.2 : Crop Image

Description : User can crop image for better detection

Input : Image

Output : Crop Image

R.2 : Detect Number

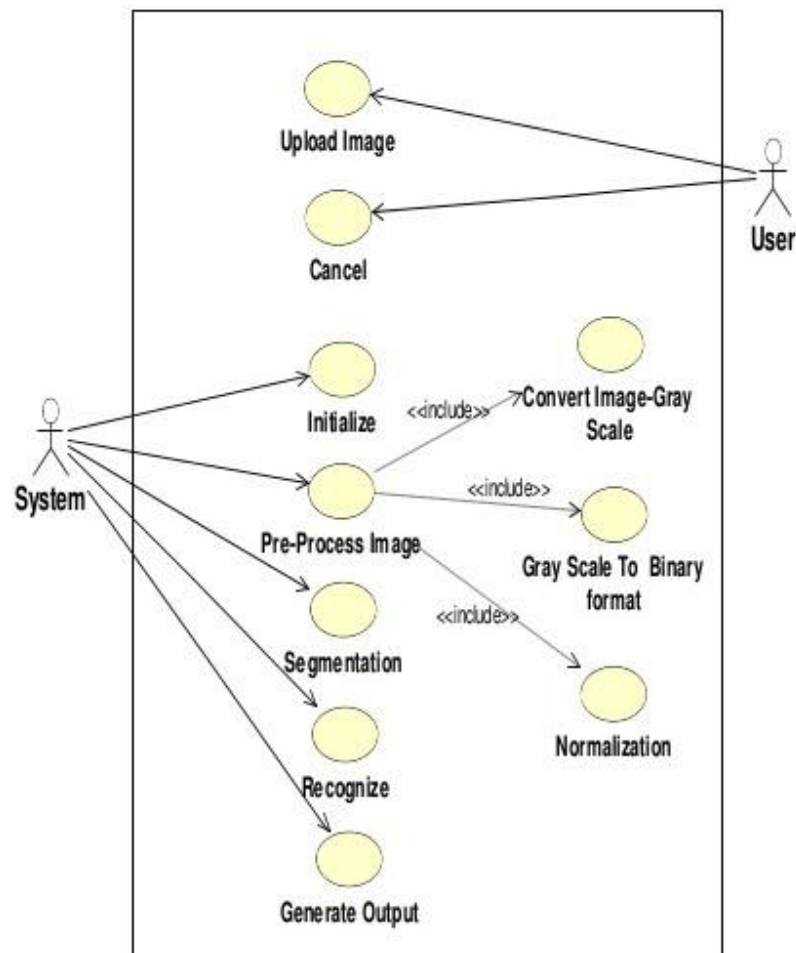
Description : Detect the number which is in image

Input : Image

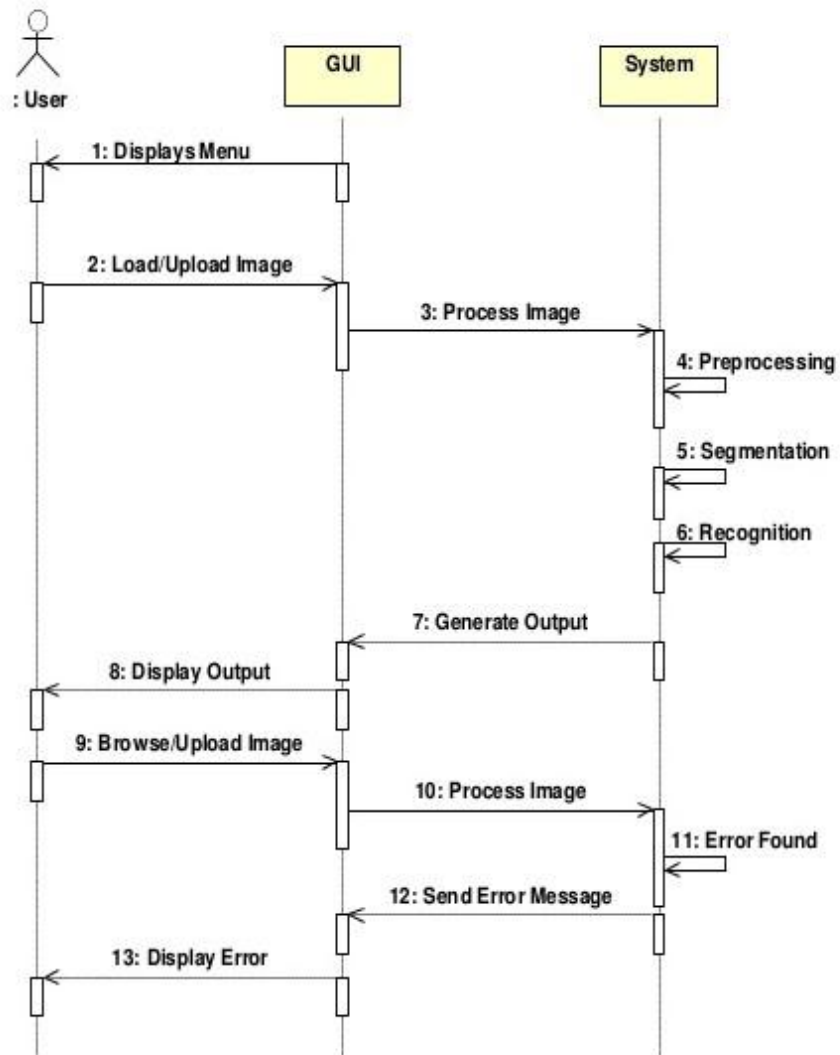
Output : Detected number

4.Design

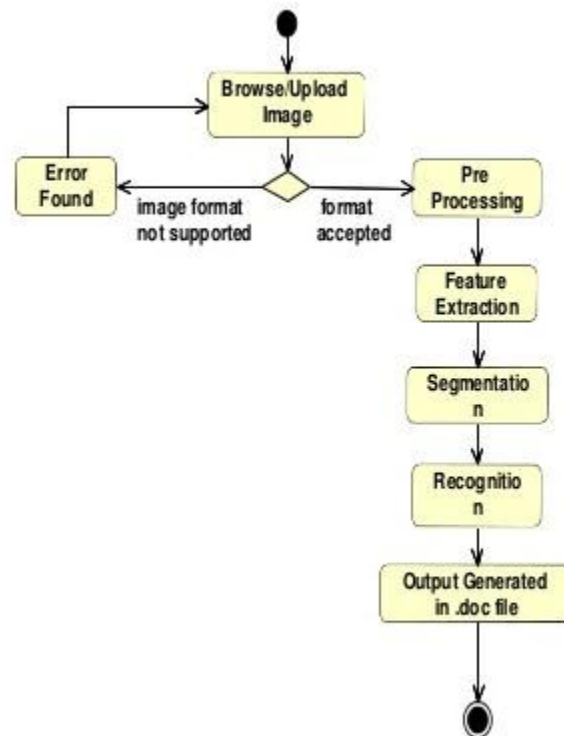
4.1 : Usecase Diagram



4.3 : Sequnce Diagram



4.4 : Activity Diagram



4.6: Data Dictonry

What is Manist Dataset?

Samples provided from MNIST (Modified National Institute of Standards and Technology) dataset includes handwritten digits total of 70,000 images consisting of 60,000 examples in training set and 10,000 examples in testing set, both with labeled images from 10 digits (0 to 9). This is a small segment form the wide set from NIST where size was normalized to fit a 20*20 pixel box and not altering the aspect ratio. Handwritten digits are images in the form of 28*28 gray scale intensities of images representing an image along with the first column to be a label (0 to 9) for every image. The same has opted for the case of the testing set as 10,000 images with a label of 0 to 9.

Yann Lecun, Corinna Cortes, and Christopher Burges developed this MNIST dataset for evaluating and improving machine learning models on the handwritten digit classification problem. The MNIST dataset was developed from the special dataset from NIST with special database 3 (United States Census Bureau employees) and special database 1 (high school students) which consist with the binary images of handwritten digits. Earlier SD-3 (special database -3) was considered as training and SD-1 (special database -1) as testing set with easier recognizing level of SD-3. Therefore to keep it challenging, disjoint and fair among different learning classifiers, NIST dataset was mixed up. Division of the MNIST took place by 30,000 samples from SD-3 and 30,000 samples from SD-1 with 250 writers approx. and 5,000 samples from SD-3 and remaining 5,000 samples from SD-1 to form a different test set. Images of digits were taken from various scanned digits, normalized in size and justify as centered. This makes it an excellent dataset for evaluating models and allowing the machine learning aspirant to focus on deep learning and machine learning with very little data cleaning.

Talking about the newer or more modified version which is similar to the standard MNIST, an EMNIST or Extended MNIST have been emerged out in the year 2017 with the samples of 2, 40,000 images in training set along with increment to 40,000 images in the testing set consisting of handwritten digits.



5. Tools/Technologies

Technologies:

Android Studio

XML

Java

Flask

Python

Mnist (Dataset)

Tools:

Visual Studio Code

Android Studio

Anaconda

Platforms:

Service run on laptop (Which provide a service to Android app)

Application app run On Application Phone (Which consume a service)

6. Implementation Detail

6.1: Modules

- **Home**

It is for insert image directly from gallery or camera.

6.2: Functions

- **Add Image**

It is the basic function for Add image that user wants search.

- **Process Image**

It is the function for process image for identify details that user added previously.

- **Display data**

It display the data of which is process by server and response back to the user

7. Testing

7.1: Unit Testing

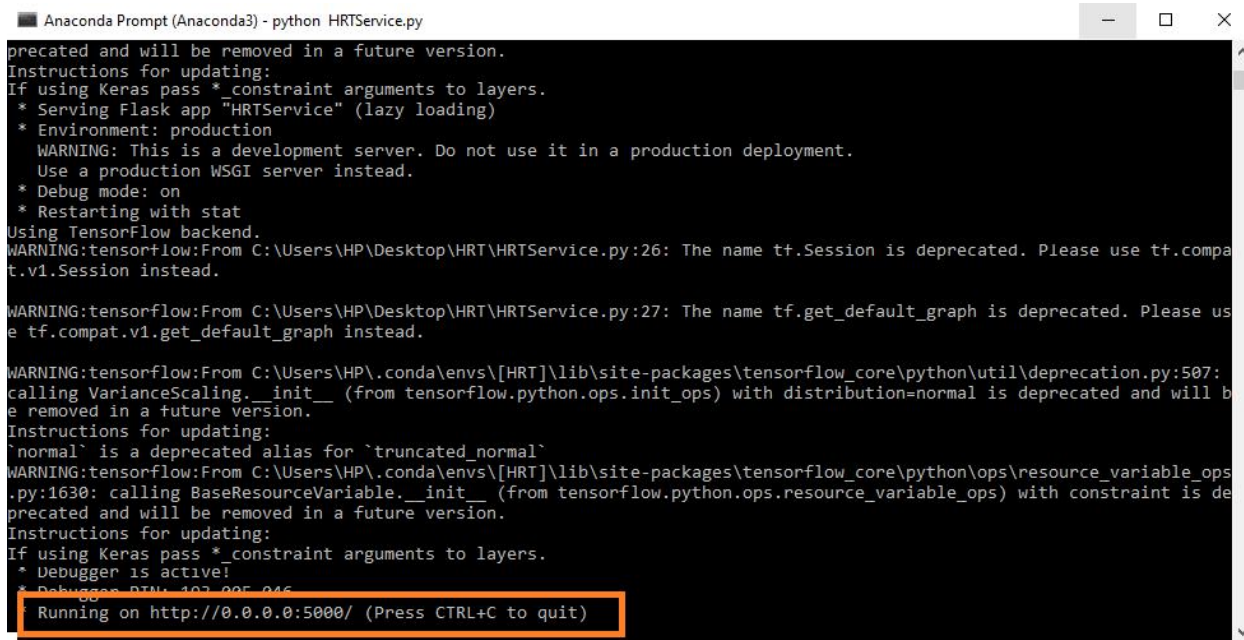
Android Studio Code provides better way of testing. After combining all application. We can generate android application. Application can be tested using unit testing.

7.2: Integration Testing

After combining application android application needed to be tested. Whole android application can be tested after entering the different types of data. Application also tested added as different images. We solved some issues and break of the application.

8. Screen shots

1.service Start



```

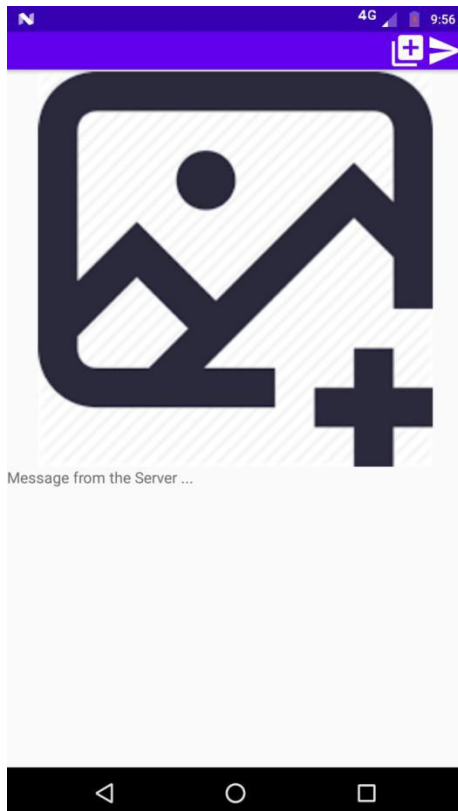
Anaconda Prompt (Anaconda3) - python HRTService.py
preated and will be removed in a future version.
Instructions for updating:
If using Keras pass *constraint arguments to layers.
* Serving Flask app "HRTService" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
Using TensorFlow backend.
WARNING:tensorflow:From C:\Users\HP\Desktop\HRT\HRTService.py:26: The name tf.Session is deprecated. Please use tf.compat.v1.Session instead.

WARNING:tensorflow:From C:\Users\HP\Desktop\HRT\HRTService.py:27: The name tf.get_default_graph is deprecated. Please use tf.compat.v1.get_default_graph instead.

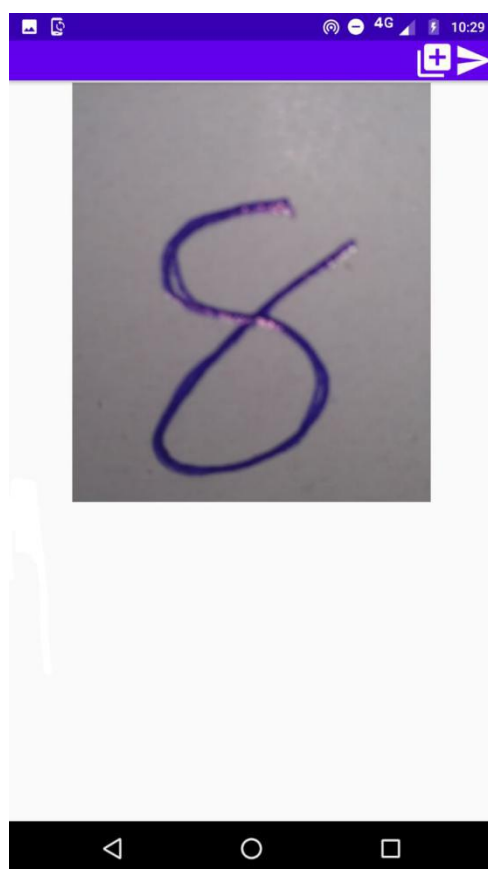
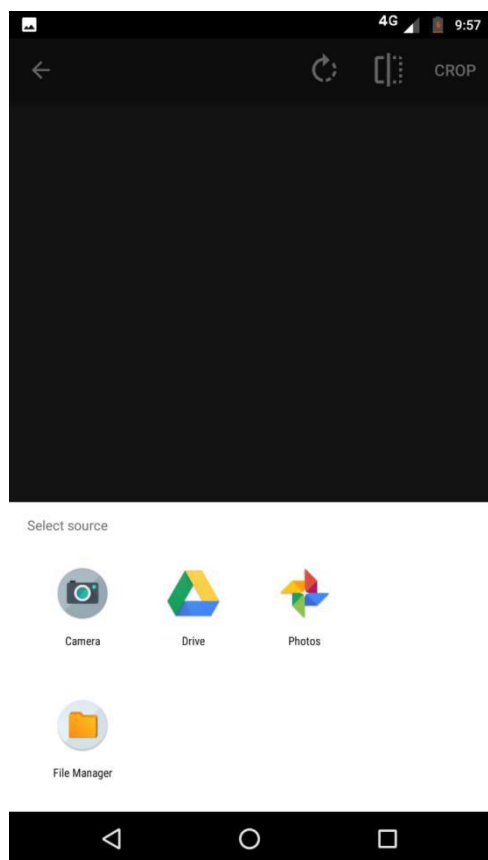
WARNING:tensorflow:From C:\Users\HP\.conda\envs\[HRT]\lib\site-packages\tensorflow_core\python\util\deprecation.py:507: calling VarianceScaling.__init__ (from tensorflow.python.ops.init_ops) with distribution=normal is deprecated and will be removed in a future version.
Instructions for updating:
`normal` is a deprecated alias for `truncated_normal`
WARNING:tensorflow:From C:\Users\HP\.conda\envs\[HRT]\lib\site-packages\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *constraint arguments to layers.
* Debugger is active!
* Debugger PIN: 107-005-046
Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)

```

2. Application start



3. Image send and detection process for image of 8




```
Anaconda Prompt (Anaconda3) - python HRTService.py

Instructions for updating:
If using Keras pass *_constraint arguments to layers.
* Debugger is active!
* Debugger PIN: 193-005-046
* Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)

Received image File name : predict.jpg

Output : 202

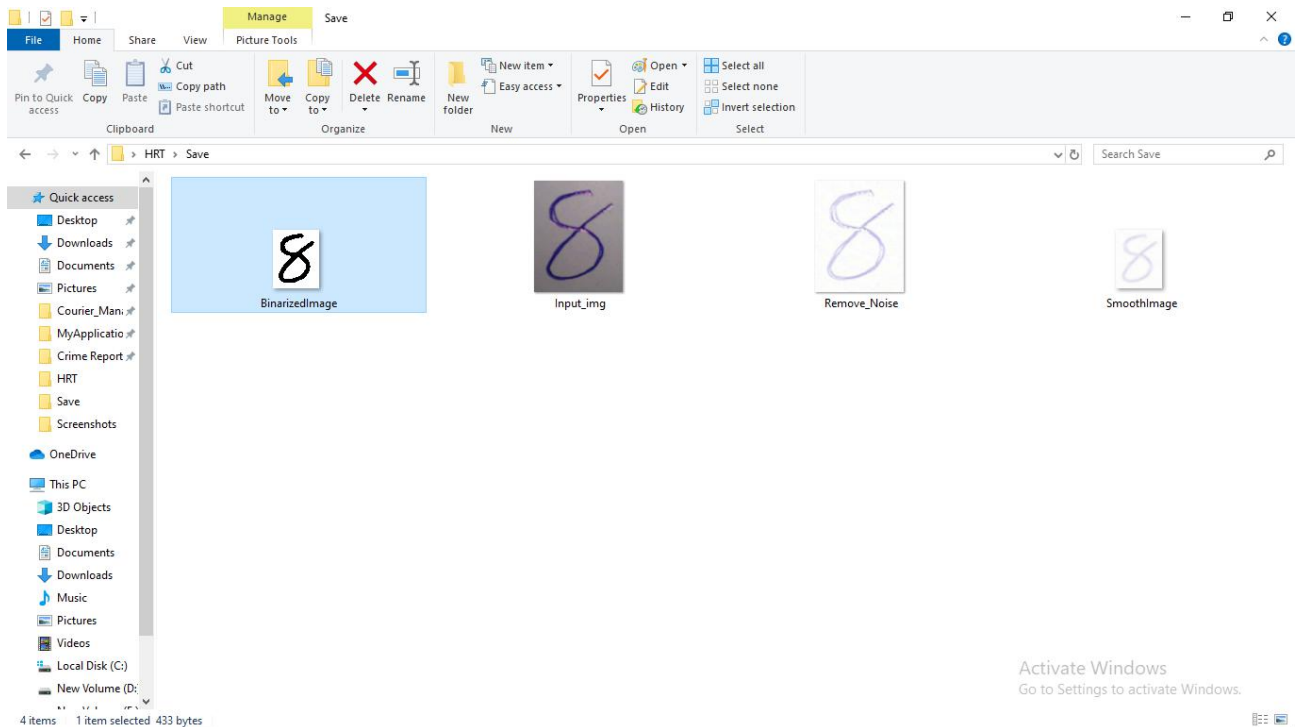
192.168.43.1 - - [24/Apr/2020 22:08:28] "[37mPOST / HTTP/1.1[0m" 200 -
Received image File name : predict.jpg

Output : 8

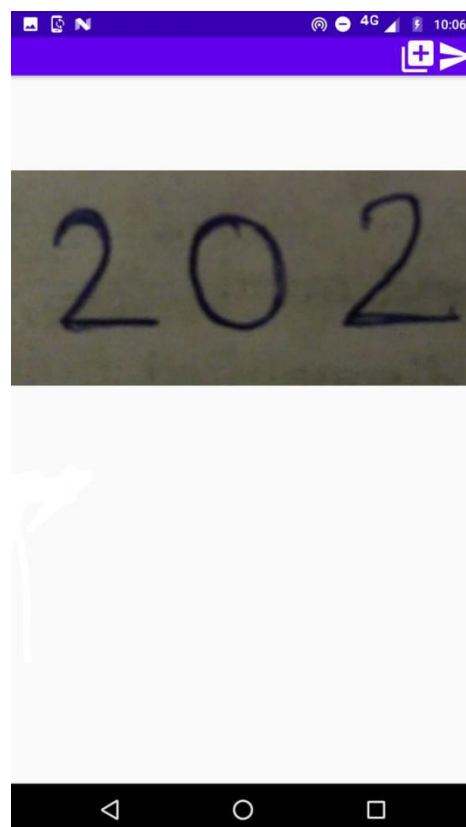
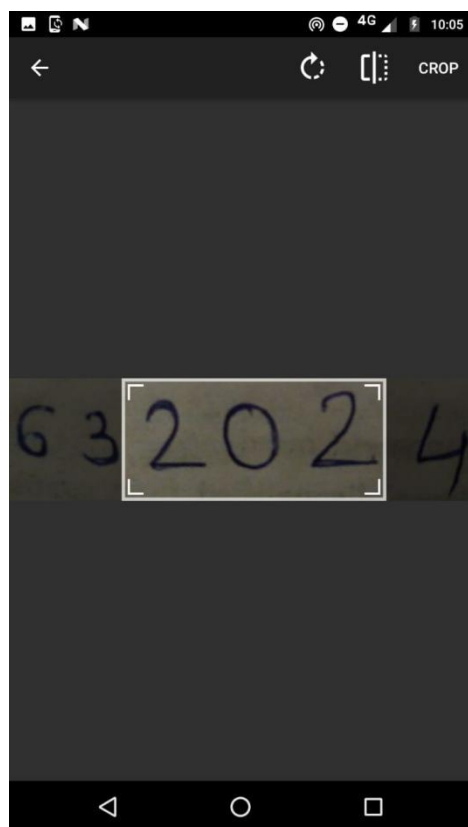
192.168.43.1 - - [24/Apr/2020 22:14:57] "[37mPOST / HTTP/1.1[0m" 200 -
Received image File name : predict.jpg

Output : 8

192.168.43.1 - - [24/Apr/2020 22:15:19] "[37mPOST / HTTP/1.1[0m" 200 -
```



4. Image send and detection process for image of 202



```

Anaconda Prompt (Anaconda3) - python HRTService.py

* Restarting with stat
Using TensorFlow backend.
WARNING:tensorflow:From C:\Users\HP\Desktop\HRT\HRTService.py:26: The name tf.Session is deprecated. Please use tf.compat.v1.Session instead.

WARNING:tensorflow:From C:\Users\HP\Desktop\HRT\HRTService.py:27: The name tf.get_default_graph is deprecated. Please use tf.compat.v1.get_default_graph instead.

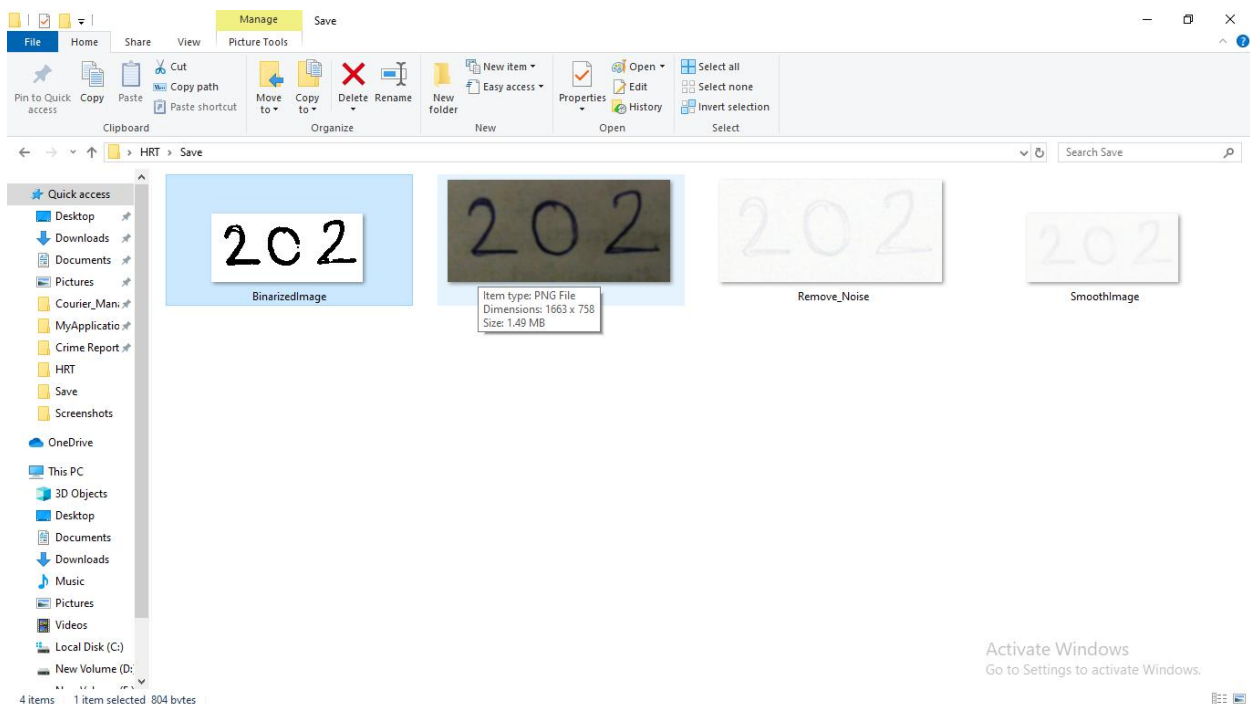
WARNING:tensorflow:From C:\Users\HP\.conda\envs\[HRT]\lib\site-packages\tensorflow_core\python\util\deprecation.py:507: calling VarianceScaling.__init__ (from tensorflow.python.ops.init_ops) with distribution=normal is deprecated and will be removed in a future version.
Instructions for updating:
`normal` is a deprecated alias for `truncated_normal`
WARNING:tensorflow:From C:\Users\HP\.conda\envs\[HRT]\lib\site-packages\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
* Debugger is active!
* Debugger PIN: 193-005-046
* Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)

Received image File name : predict.jpg

Output : 202

192.168.43.1 - - [24/Apr/2020 22:08:28] "[37mPOST / HTTP/1.1[0m" 200 -

```



9. Conclusion

Handwritten digits recognition provide platform to use machine as human brain. The human can identify the digits which is written in paper. This application provide better way to identify digits which is very useful in many machine learning project.

10. Limitations and Future Enhancements

10.1: Limitations:

We have only implemented for 0 to 9 digits which is present in mnist dataset. If we have hindi dataset or gujrati dataset then we can also indentify hindi or gujrati digits. So that we have not implemented. If digit is greater than 9 than it is also not identify by our project.

10.2 Future Enhancements:

We will extend our project for hindi and gujarati dataset so we can indentify the hindi or gujrati digits. We will extend our project for character indentify also. If we join the charater then it become word which also need to indentify our project. That we need to implement in our project.

11. Reference / Bibliography

❖ Following links and websites were referred during the development of this project.

- <https://www.google.co.in>
- <https://anaconda.org/anaconda>
- <http://www.stackoverflow.com>
- <https://developer.android.com/studio>