MNIST-PROJECT

REPORT

Zhang Kehan

2019.8.17

Contents

ABSTRACT

This project is a hand-writing digit prediction project written by Python. This project's main recognition algorithm is Convolutional Neural Networks, CNN. It is able to recognize hand-writing single digits. And the accuracy of the trained model is 99.3%.

The whole project is deployed in docker container, and is able to communicate with Cassandra container. The web page is based on FLASK,a light-weight web application framework. This project use a mnist model that has been trained beforehand, whose accuracy is up to 99.3%. This project follows the RESTful.

Users can upload images of a hand-writing digit to the website by clicking the "choose file" button, and click the "submit" button, then the prediction of the hand-writing digit will be shown on the web page. In addition, the record of this prediction will be saved to Cassandra, one prediction record include the time of prediction, the name of the uploaded file and the predict result.

Keywords: Python, Mnist, CNN, Docker, Cassandra, Flask, RESTful,

1 INTRODUCTION

This python program is a project that use the mnist model that have been trained to help users to recognize the images that they upload to the server.

2 FUNCTIONAL REQUIREMENT

This project have to implement the following requirements:

2.1 display on web

User uploads a image of any format to the server, the server will be able to recognize the digit in the image and return the result of recognition on the HTML page.

2.2 use flask

The program is based on flask, the lite web program framework.

2.3 record results in cassandra

The result(including time of recognition, name of uploaded file and the result of recognition) should be recorded in Cassandra.

2.4 use docker container and enable communication between containers

The whole main program should be deployed and run in docker and is able to communicate with Cassandra.

3 REALIZATION

3.1 train and save model

use 20000 images from MNIST_DATA to train the model. After access the model, we know that the accuracy of this model is up to 99.3%. saved models are as followed.

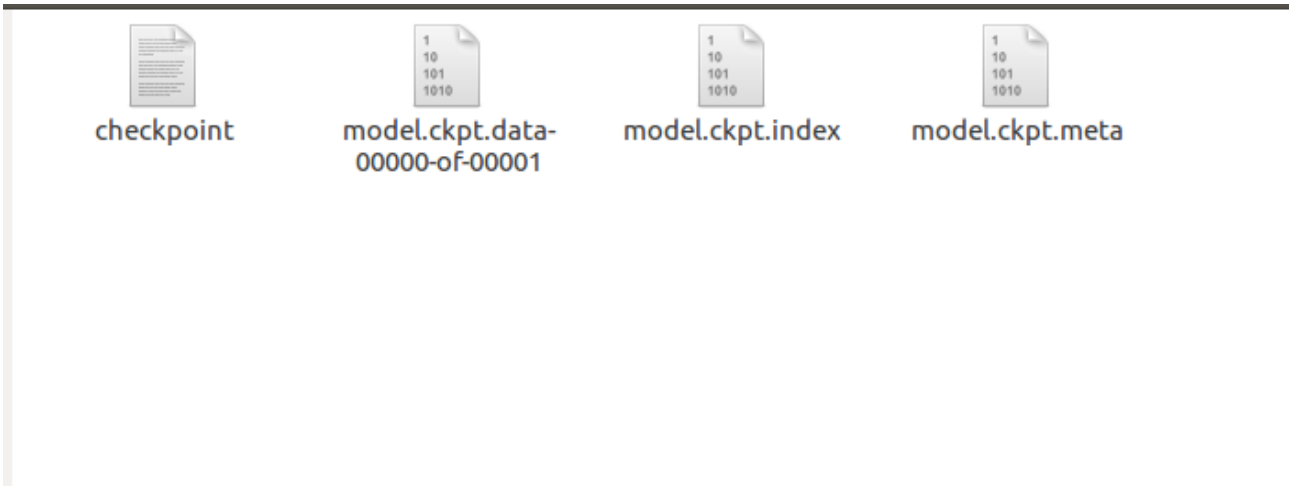

Figure 1 Trained and saved model

3.2 manage images

There are several procedures that have to be done to manage the image uploaded by the user. First, I convert RGB image into gray

image, and then I convert the image to 28 x 28 pixels. The processed images are as followed. And the main code is shown below:

```
#find the path of current file
basepath = os.path.dirname(__file__)

#find the path of the file that is used to save images
upload_path = os.path.join(basepath, 'upload_file',secure_filename(f.filename))

#convert the relative path to abusolute path
upload_path = os.path.abspath(upload_path)

#save the image to specific path
f.save(upload_path)

#read the images
im = Image.open(upload_path)

#convert the image to grayscale image
im = im.convert('L')

#sharp the image
im=ImageEnhance.Sharpness(im).enhance(3)

#convert the image to 28*28 Pixels
im=im.resize((28,28))

#save the standardlized image to specific path
im.save(upload_path)
```

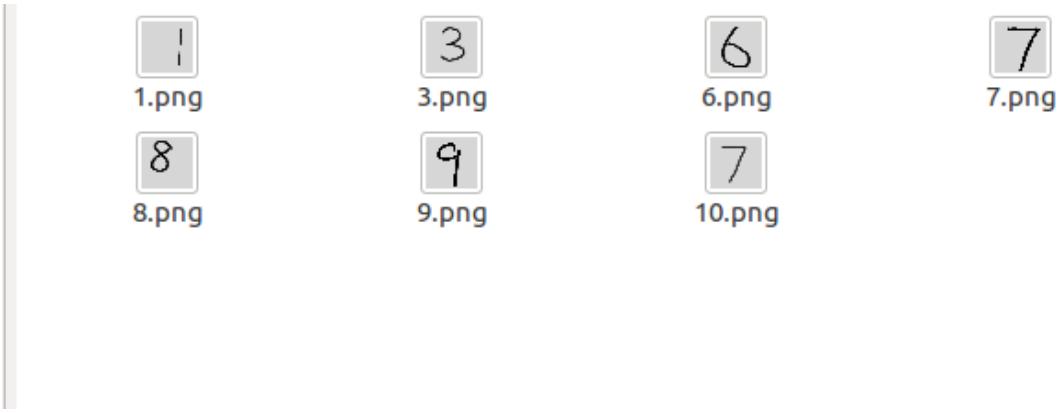

Figure 2 Converted and saved images

3.3 implement web page with FLASK

Type the url as 0.0.0.0:5000/upload

Figure 3 The web page of the program

3.4 use model

Press the “select file” button and find the image you want to recognize and then Press the “submit” button to execute the predict process, the result will be displayed in the flame.

Figure 4 Display the result on the web page

3.5 deploy the whole program into container

use the dockerfile to make program images

```
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
testproject         latest              54e618fa0f31        10 hours ago
966MB
```

Figure 5 make image of the program

3.6 connect to cassandra

```
cqlsh:mnistkeyspace> select * from predictrecord;

 time                     | filename | result
--------------------------+----------+--------
 2019-08-17 15:23:59 |    9.png |      9

(1 rows)
```

Figure 6 Check the record in Cassandra

4 PROBLEMS AND SOLUTIONS

During the accomplishment of the project, I encounter several problems and under the help of my teacher ZHANG FAN and blogs on the Internet, I solved them and finished the project successfully. Problems and solutions are as followed.

4.1 problem of use the model

error:Key Variable_*** not found in checkpoint when use the model

solution1:First, I checked if the name of variables are correspond to the name of those that in the saved model

solution2:Then, I find that replace 'tf.get_variable()' with 'tf.Variable()' can solve the problem. Because when we use tf.Variable(), if the system inspect a conflict in naming, it will handle the conflict automatically.

4.2 problem of the image's lacking fidelity

solution:sharpening the image using 'im=ImageEnhance.Sharpness(im).enhance(3)'

4.3 problem of showing the html template

error:templateNotFound when use flask

solution: create ‘templates’ file folder, and put the html file into the ‘templates’ file folder. Because when execute ’render_template('handwriting_webpage.html')’ , the program will find the html file in the default file folder,which is ‘templates’.

4.4 problems of communication between containers

error:unable to connect to any server
solution1: unify the port that is exposed by the container and the port that is run in the program.
solution2: run the two container in the same network
solution3: specify the ip of cassandra and force the program to moniter the ip.

Notice: There is a problem impede me for a long time. After we run the cassandra container, we should wait the cassandra container to be fully started for seconds, and then run the program container. Otherwise, two container will not be able to communicate with each other.