Summary Report

2019/08/16

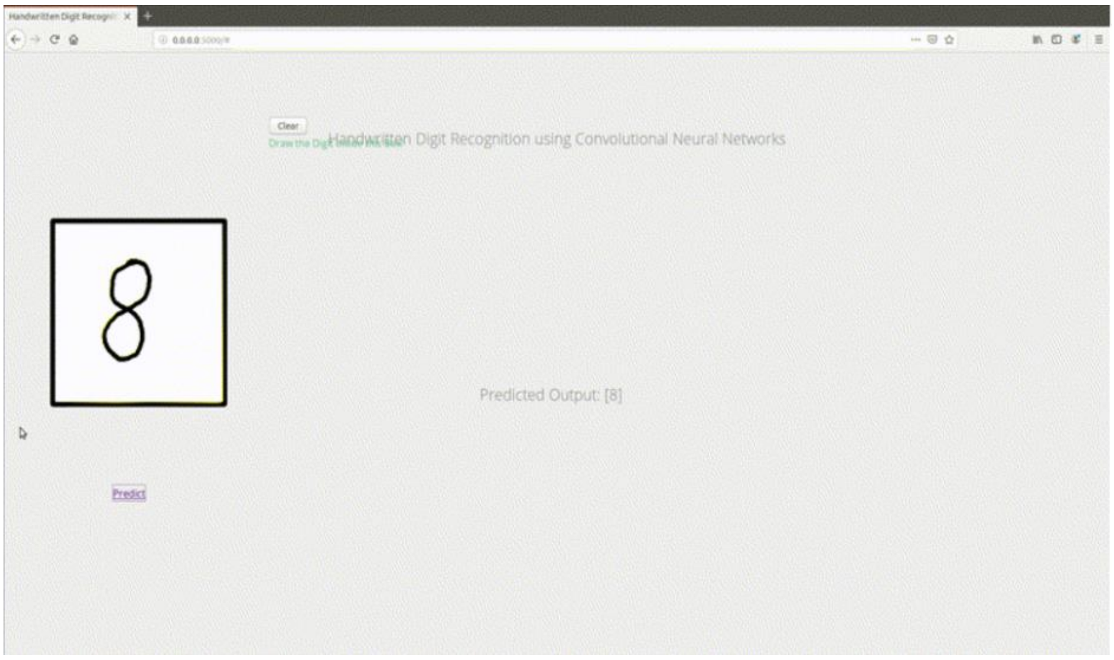
Zongduo Li

Abstract

This project is constructed by two docker Containers: an Application Docker Container and a Database Docker Container.

While the user submits ports in any browser, the Flask Router will link to the HTML5 file. In the HTML5 file, there is a canvas which user can use mouse to draw. The Router then saves the handwritten image and requests the prediction from the MNIST Keras model. After the result returns, the Router forwards the result to HTML5 and show it to user via browser and submits all two data to the Cassandra Database Container through the Docker Network Bridge.

The main job for this project is to recognize the user's handwritten digit. User can draw the digit in the canvas and press "predict it" button to start to predict the image, and it will return the result on the right side of the page. User can also press "clear it” button to clear their drawing if he is not satisfy with it. Each time, the prediction and date will be recorded in the Cassandra database.


Clear
Draw the Digit inside this Box!
Handwritten Digit Recognition using Convolutional Neural Networks
Predicted Output: [8]
Predict

1. Introduction

1.1 Research Status of Digital Recognition

Early researchers have achieved good results in the direction of digital recognition, such as using K-neighborhood classification method, SVM classification method, Boosting classification method and so on.

However, there are some shortcomings in these methods. For example, K-proximity method needs to load all training data sets into memory, and then use the digital image to be measured to sum the corresponding pixel difference with the training set. Finally, the prediction results are the smallest difference. Obviously, such a method is not reliable in normal picture accuracy, and the requirement for handwritten numerals to be tested is also very high. At present, the best recognition model should be CNN based on in-depth learning. The most typical example is LeNet-5. The United States first used it commercially to identify handwritten numbers on bank cheques [2]. It can be seen that handwritten numeral recognition based on in-depth learning is quite reliable in accuracy.

1.2 The Development and Present Situation of Deep Learning

The development of machine learning can be roughly divided into two stages. It originates from shallow learning and develops from deep learning, which is an important branch of machine learning.
In the late 1980s, the application of back propagation algorithm brought hope to machine learning, but also set off a boom of machine learning based on statistical model. Through practice, people have successfully found that back propagation algorithm can make an artificial neural network model learn certain rules of statistics in a large number of labeled training samples, and then predict unlabeled things. The network model in this stage is called shallow model because it contains only one hidden layer. The shallow model has some bottlenecks in the number of parameters, computing units and feature expression. In the 1990s, various shallow learning models have been proposed, such as support vector machine (SVM), Boosting, maximum entropy method and so on. Compared with the neural network model at that time, these models have a certain improvement in efficiency and accuracy [4].

Until 2006, Geoffrey Hinton, a professor at the University of Toronto, Canada, and his student Geoffrey Hinton, a leader in machine learning, published an overview article on "deep learning" in Ruslan Salakhutdinov's Science, which officially opened the wave of deep learning. The flagship event of in-depth learning is that Alex Krizhevsky and Ilya Sutskever, Ph.D. students of Geoff Hinton in 2012, used in-depth learning to get the first recognition result on ImageNet in the competition of image classification, and they are ahead of the Google team in recognition accuracy with in-depth learning. The Google team here is not a normal team, but a team led by Andrew Ng, the leader in machine learning, and Jeff Dean, the God of Google, with hardware and data resources beyond the reach of other teams. What defeats this top team is only two "hairy children" who have been studying in depth for a short time. At the same time, it also attracts large-scale investment in deep learning from industry. Google acquired Hinton's DNN startup and invited Hinton to join Google; LeCun joined Facebook and became director of AI Labs; Baidu set up its own Deep Learning Institute and invited Wu Enda, who was formerly in charge of Google Brain. The rapid development of in-depth learning is astonishing. In the early 2016 game between Google's AlphaGo system and Weiqi world champion Li Shishi, AlphaGo won the game with a big score of 4:1. It can be seen that the third wave of AI is also closely related to in-depth learning. The most classical models in deep learning are fully connected neural network, convolutional neural network CNN and cyclic neural network RNN. Another very important technology is deep reinforcement learning technology, which AlphaGo uses.

The success of in-depth learning is attributed to three factors: big data, big model and big calculation. At the same time, these three directions are also the hotspots of current research. Benefiting from the improvement of computing power and the emergence of large data, in-depth learning can deepen the level of model to the level of computing load of hundreds of millions of neurons under current conditions. GPU's support for in-depth learning computing also lowers the threshold for in-depth learning research, so that more junior researchers can step into this field, inject fresh blood into this field, constantly put forward new ideas, and broaden the new application direction.

The essence of in-depth learning is to improve the accuracy of classification or prediction of similar data by constructing machine learning model of multi-layer hidden layer and a large number of training data, and constantly adjusting parameters in training to find features that can reflect the characteristics of data sets. "Deep model" is the means, while "feature learning" is the purpose. Unlike traditional shallow learning, in-depth learning emphasizes the number of hidden layers besides input and output layers, usually with more than two hidden nodes. In addition, in-depth learning highlights the importance of feature learning. By layer-by-layer feature transformation, the feature representation of samples in the original space is transformed into a new feature space, which makes classification or prediction easier.

Convolutional neural network algorithm is one of the most important algorithms in deep learning algorithm. At present, it is mainly used in image classification, image segmentation, target detection and other related computer vision fields. The team that won the ImageNet championship in the image classification competition in 2012 mentioned above is using the improved convolution neural network to achieve a qualitative leap in recognition accuracy.

1.3 The Significance of Research

As a universal symbol in the world, the Arabic numerals transcend the boundaries of the country, culture and nation, and are widely used around us. The number of categories is appropriate, only 10 categories, which is convenient for the evaluation and testing of research methods. Research on handwritten numeral recognition method based on in-depth learning is helpful to the understanding of in-depth learning, and has great theoretical and practical value. The experience of handwritten numeral recognition method can also be extended to other character recognition problems, such as the recognition of English letters.

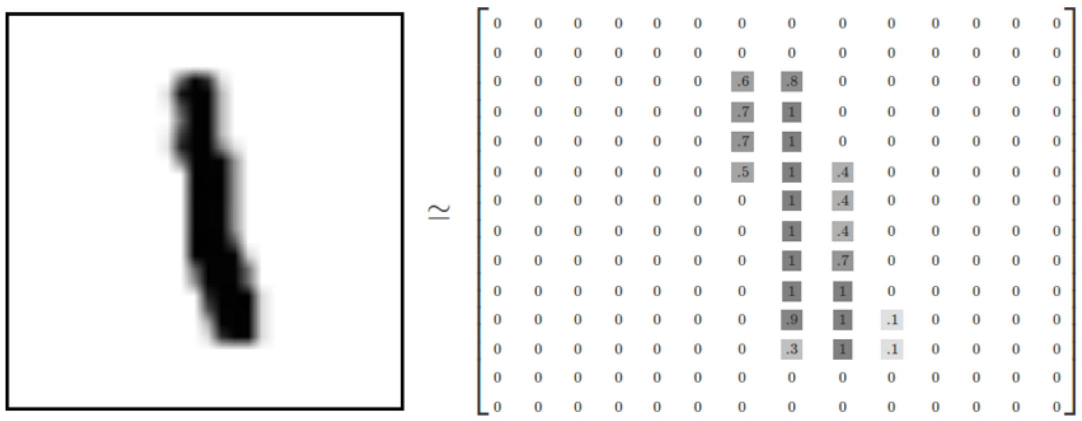
1.4 MNIST

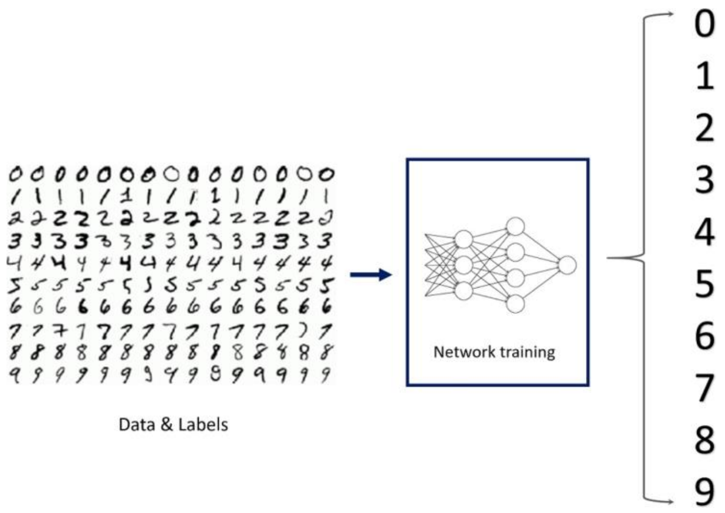
The MNIST dataset consists of handwritten digit images (from 0 to 9) which include 60,000 examples of training set and 10,000 examples of testing set. It is a subset of a larger set available from NIST. However, the official training set of 60,000 examples is divided into an actual training set of 50,000 examples and 10,000 validation examples. Additionally all digit images are black and white and sized-normalized and with a fix size image of 28 x 28 pixels. In general, each digit image is represented by a value between 255 and 0, such that 255 is black and 0 is white and anything in between is a different degree of grey.

This dataset is combined by two of NIST's database: Special Database 1 and Special Database 3. These two databases includes digits hand-written by high school students and employees of the United States

Census Bureau.

MNIST dataset is widely used in machine learning and image processing. On the other hand, this dataset is used by researchers to test and compare their research results with others.





2. Project Knowledge Review

2.1 Neural network

Python, as the leading language of AI era, has a very good ecological environment. NumPy, the library of numerical algorithms, is basically concise and efficient for matrix and set operations. In order to analyze the next Python code smoothly, I selected one code to focus on, slightly modified (the number of the first two layers of neurons) can run alone.

```
1 import numpy as np
2 sizes = [8, 15, 10]
3 biases = [np.random.randn(y, 1) for y in sizes[1:]]
4 weights = [np.random.randn(y, x) for x, y in zip(sizes[:-1], sizes[1:])]
```

Line 1: Import numpy and enable NP as an alias.

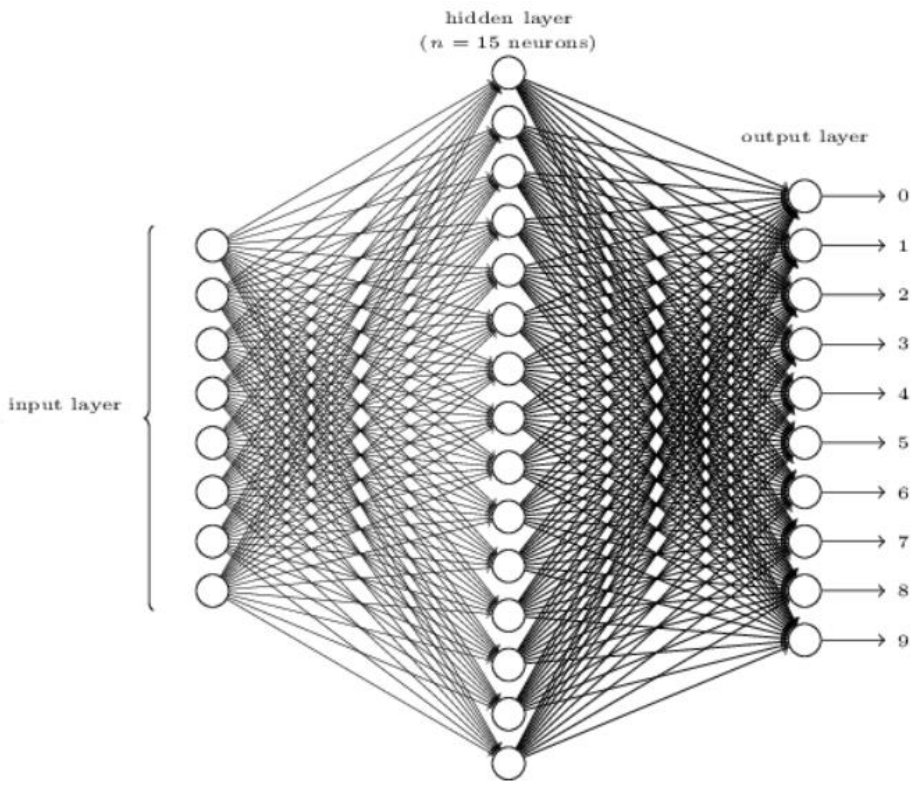
Line 2: An array definition that contains three elements.

Line 3:
Look at sizes [1:], which represents a subarray of sizes. It contains elements from subscript 1 of the original array to the last element of the original array. Its value can be calculated as [15, 10].
Secondly, NumPy's random number generation method, random. randn, which generates random numbers, conforms to the standard normal distribution with the mean value of 0 and the standard deviation of 1.
The parameters of random.randn method, which describe the shape of the tensor generated, such as random.randn(2,3) will generate a tensor with rank 2 and shape[2,3], are a matrix [[-2.17399771, 0.20546498, -1.2405749], [-0.36701965, 0.12564214, 0.10203605]. For tensors, please refer to the 2 TensorFlow kernel basis.
Line 3 shows the power of Python and NumPy as a whole: parameterization of method parameters, that is, when randn method is called, variable randn (y, 1) is passed in, while variable y traverses set sizes [1:], the effect is equivalent to [randn (15, 1), randn (10, 1)];

Line 4:
First look at sizes [:-1] to indicate that it contains elements from the first of the original array to the last of the original array (penultimate second), when sizes [:-1] is [8, 15];
The two parameters of the fourth line of randn are variables y and X. The zip method appears at this time, which limits the two variables to be self-increasing synchronously, and the effect is equivalent to [randn (15, 8), randn (10, 15)].

2.2 Matrix and Neural Network

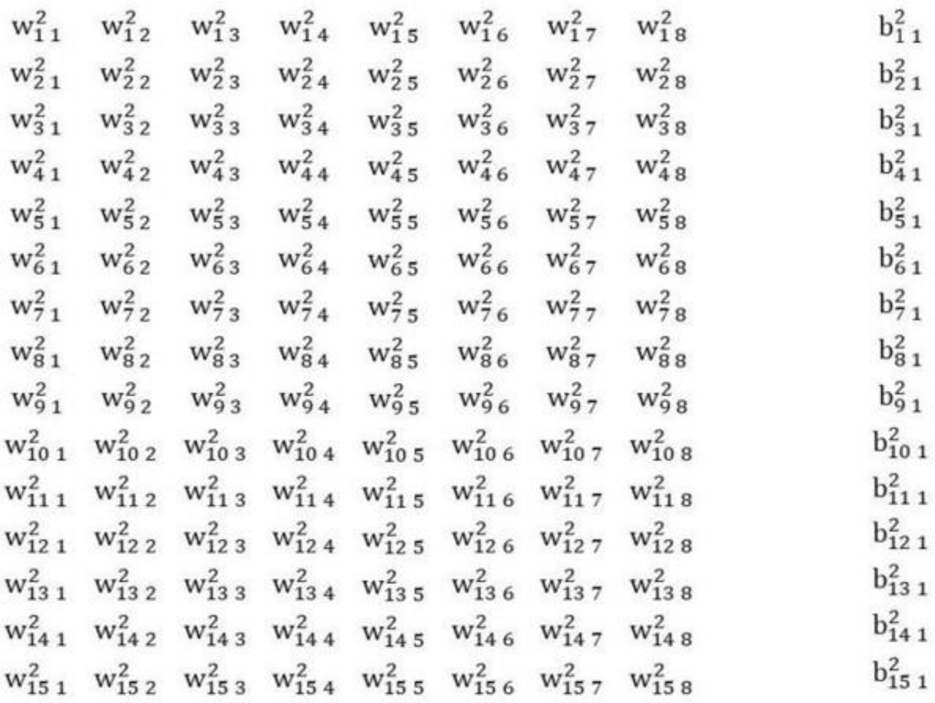
After analyzing the first four lines of code, we know how to define matrices efficiently, but what does it have to do with the construction of neural networks? The corresponding relationship between network structure and matrix structure is given below.



The above neural network structure can be described as: sizes = 8, 15, 10], 8 neurons in the first input layer, 15 neurons in the second hidden layer and 10 neurons in the third output layer.

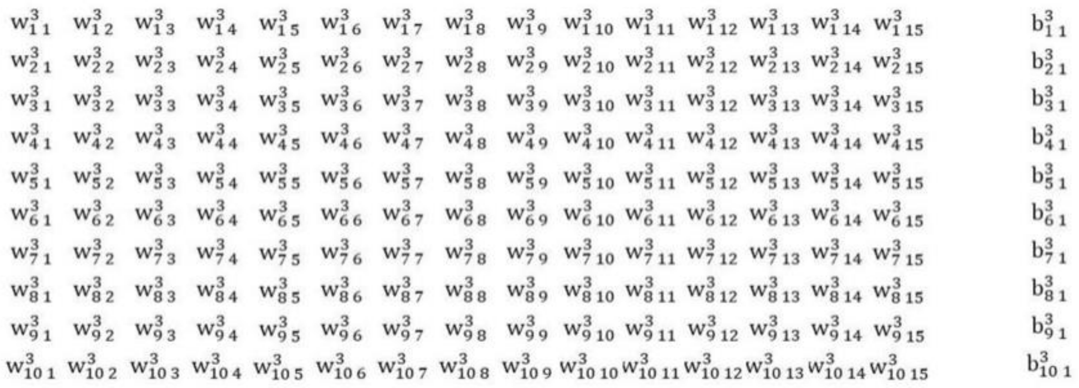
The first layer is the input layer, without weight and bias.

The weights and biases of the second layer are:



$$\begin{array}{cccccccccc}
w^2_{1\,1} & w^2_{1\,2} & w^2_{1\,3} & w^2_{1\,4} & w^2_{1\,5} & w^2_{1\,6} & w^2_{1\,7} & w^2_{1\,8} & & b^2_{1\,1} \\
w^2_{2\,1} & w^2_{2\,2} & w^2_{2\,3} & w^2_{2\,4} & w^2_{2\,5} & w^2_{2\,6} & w^2_{2\,7} & w^2_{2\,8} & & b^2_{2\,1} \\
w^2_{3\,1} & w^2_{3\,2} & w^2_{3\,3} & w^2_{3\,4} & w^2_{3\,5} & w^2_{3\,6} & w^2_{3\,7} & w^2_{3\,8} & & b^2_{3\,1} \\
w^2_{4\,1} & w^2_{4\,2} & w^2_{4\,3} & w^2_{4\,4} & w^2_{4\,5} & w^2_{4\,6} & w^2_{4\,7} & w^2_{4\,8} & & b^2_{4\,1} \\
w^2_{5\,1} & w^2_{5\,2} & w^2_{5\,3} & w^2_{5\,4} & w^2_{5\,5} & w^2_{5\,6} & w^2_{5\,7} & w^2_{5\,8} & & b^2_{5\,1} \\
w^2_{6\,1} & w^2_{6\,2} & w^2_{6\,3} & w^2_{6\,4} & w^2_{6\,5} & w^2_{6\,6} & w^2_{6\,7} & w^2_{6\,8} & & b^2_{6\,1} \\
w^2_{7\,1} & w^2_{7\,2} & w^2_{7\,3} & w^2_{7\,4} & w^2_{7\,5} & w^2_{7\,6} & w^2_{7\,7} & w^2_{7\,8} & & b^2_{7\,1} \\
w^2_{8\,1} & w^2_{8\,2} & w^2_{8\,3} & w^2_{8\,4} & w^2_{8\,5} & w^2_{8\,6} & w^2_{8\,7} & w^2_{8\,8} & & b^2_{8\,1} \\
w^2_{9\,1} & w^2_{9\,2} & w^2_{9\,3} & w^2_{9\,4} & w^2_{9\,5} & w^2_{9\,6} & w^2_{9\,7} & w^2_{9\,8} & & b^2_{9\,1} \\
w^2_{10\,1} & w^2_{10\,2} & w^2_{10\,3} & w^2_{10\,4} & w^2_{10\,5} & w^2_{10\,6} & w^2_{10\,7} & w^2_{10\,8} & & b^2_{10\,1} \\
w^2_{11\,1} & w^2_{11\,2} & w^2_{11\,3} & w^2_{11\,4} & w^2_{11\,5} & w^2_{11\,6} & w^2_{11\,7} & w^2_{11\,8} & & b^2_{11\,1} \\
w^2_{12\,1} & w^2_{12\,2} & w^2_{12\,3} & w^2_{12\,4} & w^2_{12\,5} & w^2_{12\,6} & w^2_{12\,7} & w^2_{12\,8} & & b^2_{12\,1} \\
w^2_{13\,1} & w^2_{13\,2} & w^2_{13\,3} & w^2_{13\,4} & w^2_{13\,5} & w^2_{13\,6} & w^2_{13\,7} & w^2_{13\,8} & & b^2_{13\,1} \\
w^2_{14\,1} & w^2_{14\,2} & w^2_{14\,3} & w^2_{14\,4} & w^2_{14\,5} & w^2_{14\,6} & w^2_{14\,7} & w^2_{14\,8} & & b^2_{14\,1} \\
w^2_{15\,1} & w^2_{15\,2} & w^2_{15\,3} & w^2_{15\,4} & w^2_{15\,5} & w^2_{15\,6} & w^2_{15\,7} & w^2_{15\,8} & & b^2_{15\,1}
\end{array}$$

The weights and biases of the third layer are:



$$\begin{array}{ccccccccccccccccc}
w^3_{1\,1} & w^3_{1\,2} & w^3_{1\,3} & w^3_{1\,4} & w^3_{1\,5} & w^3_{1\,6} & w^3_{1\,7} & w^3_{1\,8} & w^3_{1\,9} & w^3_{1\,10} & w^3_{1\,11} & w^3_{1\,12} & w^3_{1\,13} & w^3_{1\,14} & w^3_{1\,15} & & b^3_{1\,1} \\
w^3_{2\,1} & w^3_{2\,2} & w^3_{2\,3} & w^3_{2\,4} & w^3_{2\,5} & w^3_{2\,6} & w^3_{2\,7} & w^3_{2\,8} & w^3_{2\,9} & w^3_{2\,10} & w^3_{2\,11} & w^3_{2\,12} & w^3_{2\,13} & w^3_{2\,14} & w^3_{2\,15} & & b^3_{2\,1} \\
w^3_{3\,1} & w^3_{3\,2} & w^3_{3\,3} & w^3_{3\,4} & w^3_{3\,5} & w^3_{3\,6} & w^3_{3\,7} & w^3_{3\,8} & w^3_{3\,9} & w^3_{3\,10} & w^3_{3\,11} & w^3_{3\,12} & w^3_{3\,13} & w^3_{3\,14} & w^3_{3\,15} & & b^3_{3\,1} \\
w^3_{4\,1} & w^3_{4\,2} & w^3_{4\,3} & w^3_{4\,4} & w^3_{4\,5} & w^3_{4\,6} & w^3_{4\,7} & w^3_{4\,8} & w^3_{4\,9} & w^3_{4\,10} & w^3_{4\,11} & w^3_{4\,12} & w^3_{4\,13} & w^3_{4\,14} & w^3_{4\,15} & & b^3_{4\,1} \\
w^3_{5\,1} & w^3_{5\,2} & w^3_{5\,3} & w^3_{5\,4} & w^3_{5\,5} & w^3_{5\,6} & w^3_{5\,7} & w^3_{5\,8} & w^3_{5\,9} & w^3_{5\,10} & w^3_{5\,11} & w^3_{5\,12} & w^3_{5\,13} & w^3_{5\,14} & w^3_{5\,15} & & b^3_{5\,1} \\
w^3_{6\,1} & w^3_{6\,2} & w^3_{6\,3} & w^3_{6\,4} & w^3_{6\,5} & w^3_{6\,6} & w^3_{6\,7} & w^3_{6\,8} & w^3_{6\,9} & w^3_{6\,10} & w^3_{6\,11} & w^3_{6\,12} & w^3_{6\,13} & w^3_{6\,14} & w^3_{6\,15} & & b^3_{6\,1} \\
w^3_{7\,1} & w^3_{7\,2} & w^3_{7\,3} & w^3_{7\,4} & w^3_{7\,5} & w^3_{7\,6} & w^3_{7\,7} & w^3_{7\,8} & w^3_{7\,9} & w^3_{7\,10} & w^3_{7\,11} & w^3_{7\,12} & w^3_{7\,13} & w^3_{7\,14} & w^3_{7\,15} & & b^3_{7\,1} \\
w^3_{8\,1} & w^3_{8\,2} & w^3_{8\,3} & w^3_{8\,4} & w^3_{8\,5} & w^3_{8\,6} & w^3_{8\,7} & w^3_{8\,8} & w^3_{8\,9} & w^3_{8\,10} & w^3_{8\,11} & w^3_{8\,12} & w^3_{8\,13} & w^3_{8\,14} & w^3_{8\,15} & & b^3_{8\,1} \\
w^3_{9\,1} & w^3_{9\,2} & w^3_{9\,3} & w^3_{9\,4} & w^3_{9\,5} & w^3_{9\,6} & w^3_{9\,7} & w^3_{9\,8} & w^3_{9\,9} & w^3_{9\,10} & w^3_{9\,11} & w^3_{9\,12} & w^3_{9\,13} & w^3_{9\,14} & w^3_{9\,15} & & b^3_{9\,1} \\
w^3_{10\,1} & w^3_{10\,2} & w^3_{10\,3} & w^3_{10\,4} & w^3_{10\,5} & w^3_{10\,6} & w^3_{10\,7} & w^3_{10\,8} & w^3_{10\,9} & w^3_{10\,10} & w^3_{10\,11} & w^3_{10\,12} & w^3_{10\,13} & w^3_{10\,14} & w^3_{10\,15} & & b^3_{10\,1}
\end{array}$$

Looking back at line 3, it's equivalent to [randn (15, 1), randn (10, 1)]. It's equivalent to putting two layers of bias matrices together in a network:

```
3 biases = [np.random.randn(y, 1) for y in sizes[1:]]
```

Looking back at the fourth line of code, it's equivalent to [randn (15, 8), randn (10, 15)], which is equivalent to putting the two layers of weight matrix together in the network:

```
4 weights = [np.random.randn(y, x) for x, y in zip(sizes[:-1], sizes[1:])]
```

The four matrices themselves represent the neural network model that we want to construct. The elements in them constitute all the learnable parameters of the neural network (excluding hyperparameters). When you understand the mapping relationship between the neural network and the matrix group, you can imagine the layers of data flowing through the network until the final output form.

2.3 TensorFlow

“TensorFlow is an open-source machine learning library for research and production. TensorFlow offers APIs for beginners and experts to develop for desktop, mobile, web and cloud.”

In this project, Keras with backend TensorFlow to construct the MNIST model so that it can classify the user's handwritten digit.

This MNIST model gets to 99.29% test accuracy after 4 epochs.

2.4 Convolution Neural Network

Convolutional neural network (CNN) is a kind of artificial neural network. It is a special way of image recognition and belongs to a very effective network with forward feedback.
Conventional neural networks can not adapt to all images well. For example, in the training set of CIFAR-10, the size of the image is only 32*32*3 (32 wide, 32 high and 3 color channels). Then the number of neurons passing through the first hidden layer after the input layer will reach 3072. When the hidden layer rises from one layer to two layers, three layers or more, the sum of the weights and input products of each

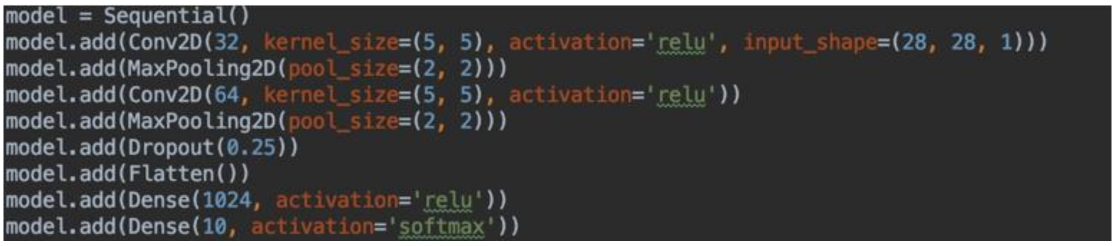
neuron in the hidden layer behind will expand beyond imagination. At the same time, in addition to the low efficiency, a large number of parameters will lead to over-fitting. Unlike conventional neural networks, the neurons in each layer of convolutional neural networks are arranged in three dimensions: width, height and depth (where the depth is not the number of layers of network structure), which is a three-dimensional structure. In the final output layer of convolution network, the data of three-dimensional structure will be converted into one-dimensional classification value in depth direction.

The main purpose of convolution neural network is to recognize two-dimensional graphics. Its network structure is highly invariant to translation, scaling, tilt or other forms of deformation. Convolutional neural network (CNN) is an efficient recognition algorithm with rapid development in recent years. Its application scope is not only limited to the field of image recognition, but also applied to face recognition, character recognition and other directions.

In this project, we use convolution to build our Keras Model.

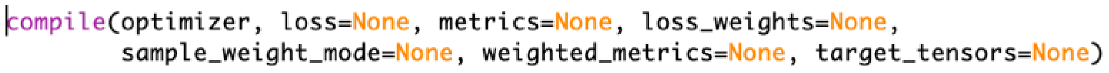
Thus, in general, we have the following process.

IMAGE —> CONVONLUTION —> MAX POOLING —> CONVOLUTION —> MAX POOLING —> FULLY CONNECTED —> FULLY CONNECTED

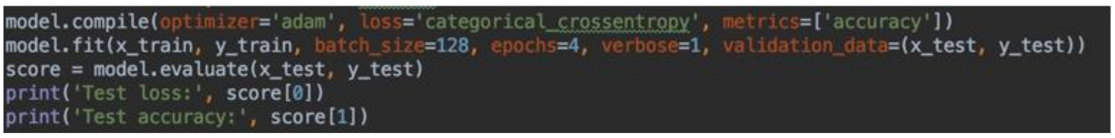


```
model = Sequential()
model.add(Conv2D(32, kernel_size=(5, 5), activation='relu', input_shape=(28, 28, 1)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, kernel_size=(5, 5), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(1024, activation='relu'))
model.add(Dense(10, activation='softmax'))
```

CLASSIFIER



```
compile(optimizer, loss=None, metrics=None, loss_weights=None,
        sample_weight_mode=None, weighted_metrics=None, target_tensors=None)
```



```python
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(x_train, y_train, batch_size=128, epochs=4, verbose=1, validation_data=(x_test, y_test))
score = model.evaluate(x_test, y_test)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

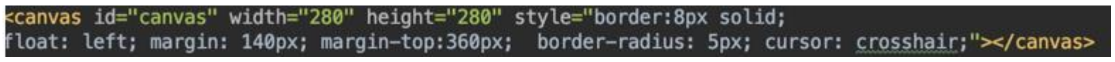
Then we compile this model by model.compile(), Here we use the optimizer Adam.

After running this python file, we get 99.29% test accuracy which is pretty good.

2.5 Flask

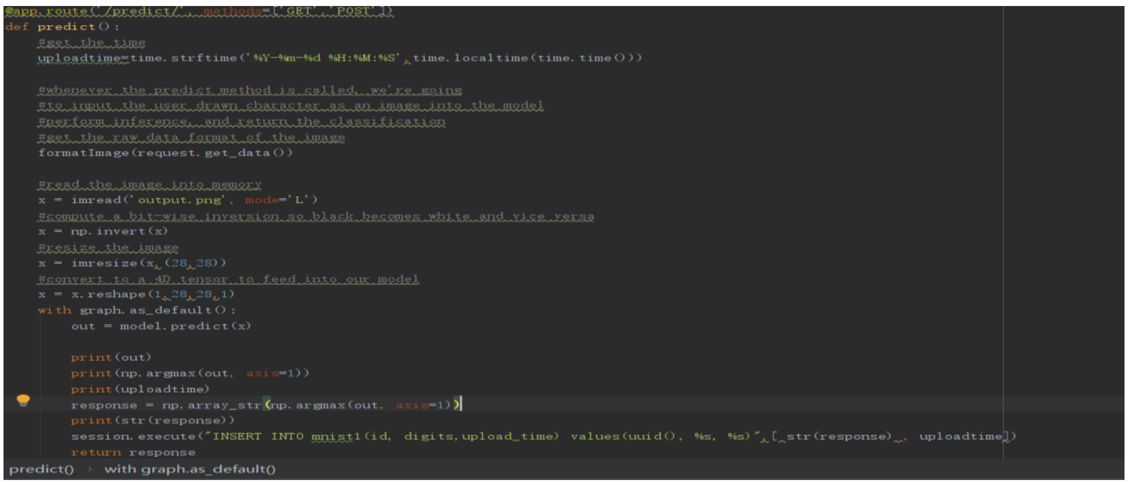
“Flask is lightweight WSGI web application framework.” Flaks is an excellent micro framework that really makes it enjoyable to work with this back-end web applications.

In my project, the Flask Router will link to the HTML5 file (index.html). In the HTML5 file, there is a canvas which user can use mouse to draw.



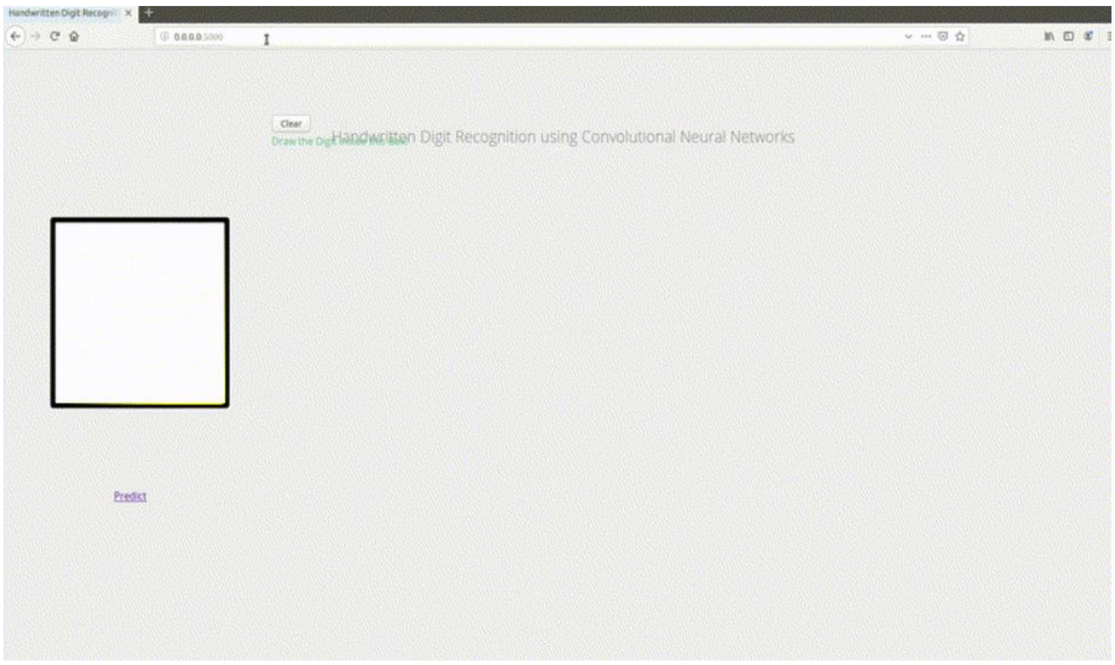
```html
<canvas id="canvas" width="280" height="280" style="border:8px solid;
float: left; margin: 140px; margin-top:360px;  border-radius: 5px; cursor: crosshair;"></canvas>
```

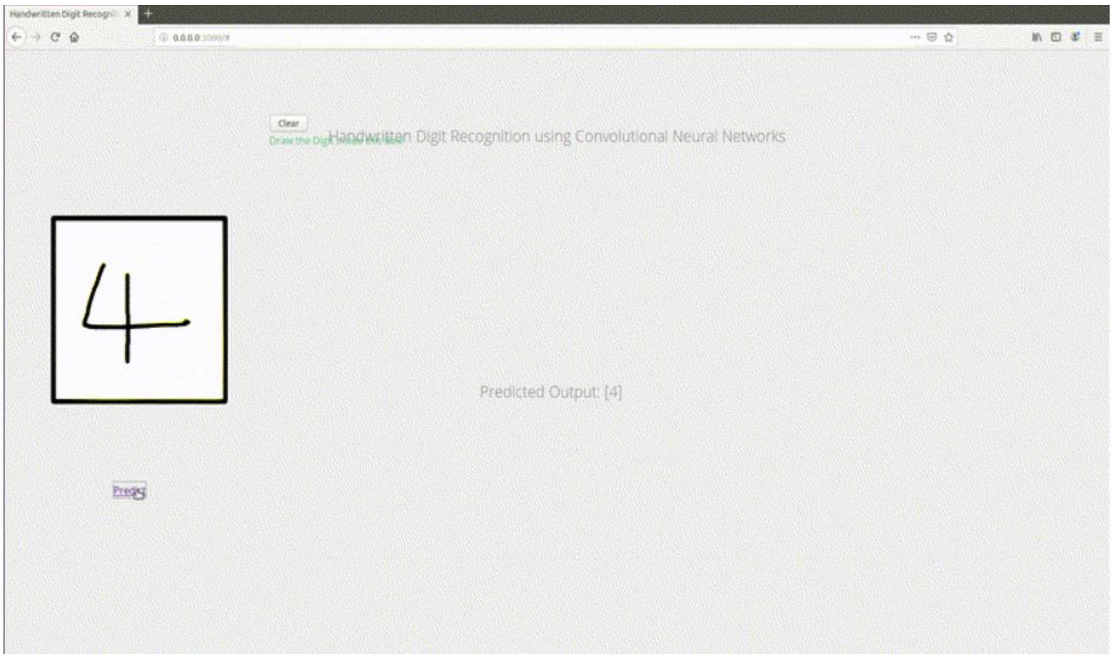
The Router then saves the handwritten image and requests the prediction from the MNIST Keras model. After the result returns, the Router forwards the result. to HTML5 and show it to user via browser and submits all two data to the Cassandra Database Container.



```python
@app.route('/predict/', methods=['GET', 'POST'])
def predict():
    #get the time
    uploadtime=time.strftime('%Y-%m-%d %H:%M:%S',time.localtime(time.time()))

    #whenever the predict method is called, we're going
    #to input the user drawn character as an image into the model
    #perform inference, and return the classification
    #get the raw data format of the image
    formatImage(request.get_data())

    #read the image into memory
    x = imread('output.png', mode='L')
    #compute a bit-wise inversion so black becomes white and vice versa
    x = np.invert(x)
    #resize the image
    x = imresize(x,(28,28))
    #convert to a 4D tensor to feed into our model
    x = x.reshape(1,28,28,1)
    with graph.as_default():
        out = model.predict(x)

        print(out)
        print(np.argmax(out, axis=1))
        print(uploadtime)
        response = np.array_str(np.argmax(out, axis=1))
        print(str(response))
        session.execute("INSERT INTO mnist1(id, digits,upload_time) values(uuid(), %s, %s)",[str(response), uploadtime])
        return response
predict()  ›  with graph.as_default()
```

This will be the interface of our application. The predicted result will be showed to user on the right side of the page


Handwritten Digit Recogni
0.0.0.0:5000
Clear
Draw the Digit
Handwritten Digit Recognition using Convolutional Neural Networks
Predict


Handwritten Digit Recogni
0.0.0.0:5000/#
Clear
Draw the Digit
Handwritten Digit Recognition using Convolutional Neural Networks
Predicted Output: [4]
Predict

2.6 Docker

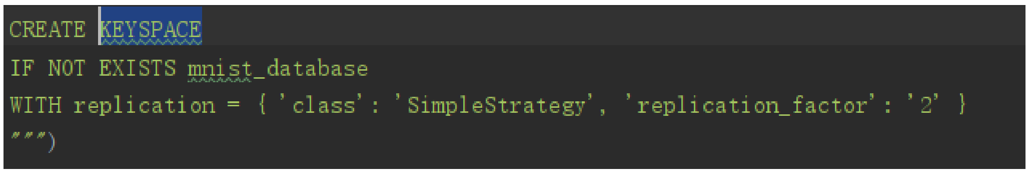
Docker is mainly a software development platform and a kind of virtualization technology that makes it easy for us to develop and deploy apps inside of neatly packaged virtual containerized environments.

This project is constructed by two docker Containers: an Application Docker Container and a Database Docker Container. Both the Application Docker Container and Database Docker Container are connected by Docker Network Bridge, so that they can communicate with each other.

2.7 Cassandra

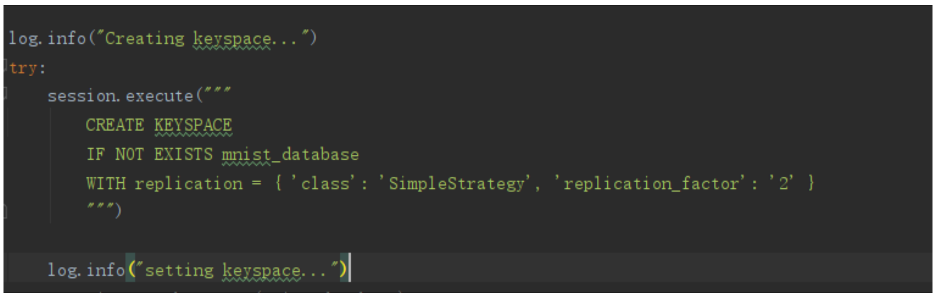
Cassandra is a free open source no sequel database. It stores that values in the form of key value pairs. Cassandra in highly robust because it has a masterless replication so basically Cassandra works in the principle of clustering.

We can Strat the Cassandra by creating a KEYSPACE.
In the project, we use Cassandra-driver, which may have a little different with the above attachment.
First we need to give the input of the variable KEYSPACE.



```
CREATE KEYSPACE
IF NOT EXISTS mnist_database
WITH replication = { 'class': 'SimpleStrategy', 'replication_factor': '2' }
""")
```

In the project, we name KEYSPACE “mnist_database”. Then similar to the work we did in the terminal, we will assign the details of this KEYSPACE.



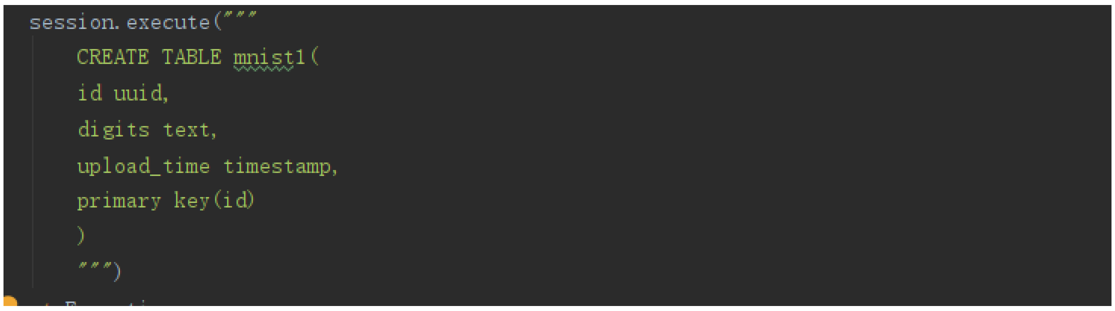
```
log.info("Creating keyspace...")
try:
    session.execute("""
        CREATE KEYSPACE
        IF NOT EXISTS mnist_database
        WITH replication = { 'class': 'SimpleStrategy', 'replication_factor': '2' }
        """)

    log.info("setting keyspace...")
```

Then let's start to create the table inside this KEYSPACE. We can also create the table through terminal.

We must tell Cassandra which variable is PRIMARY KEY, since it specifies which

node will hold a particular table row.

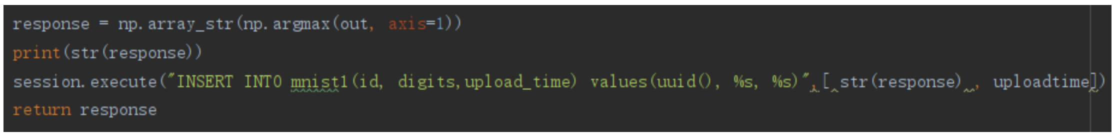
In the project we create the table by using Cassandra-driver/



```
session.execute("""
    CREATE TABLE mnist1(
    id uuid,
    digits text,
    upload_time timestamp,
    primary key(id)
    )
    """)
```

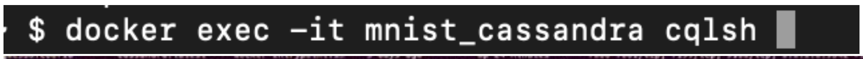
The last thing we need to do is inserting the data into our table.

```
Insert into keyspace_name.table_name(
     ColumnName1,
     ColumnName2,
     ColumnName3 . . . .)
values (
     Column1Value,
     Column2Value,
     Column3Value . . . .)
```

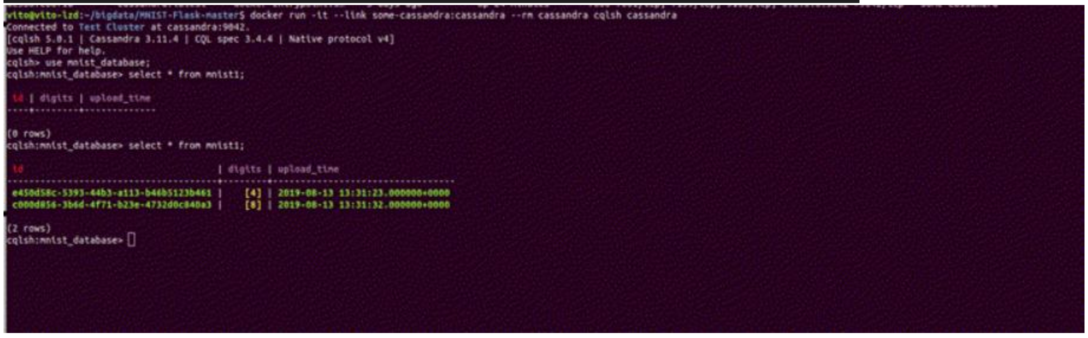


```
response = np.array_str(np.argmax(out, axis=1))
print(str(response))
session.execute("INSERT INTO mnist1(id, digits,upload_time) values(uuid(), %s, %s)",[str(response), uploadtime])
return response
```

Then we can use the following command to call cqlsh, and take a look at our Cassandra.



```
$ docker exec -it mnist_cassandra cqlsh
```



```
vito@vito-lzd:~/bigdata/MNIST-Flask-master$ docker run -it --link some-cassandra:cassandra --rm cassandra cqlsh cassandra
Connected to Test Cluster at cassandra:9042.
[cqlsh 5.0.1 | Cassandra 3.11.4 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
cqlsh> use mnist_database;
cqlsh:mnist_database> select * from mnist1;

 id | digits | upload_time
----+--------+-------------

(0 rows)
cqlsh:mnist_database> select * from mnist1;

 id                                   | digits | upload_time
--------------------------------------+--------+---------------------------------
 e450d58c-5393-44b3-a113-b46b5123b461 |    [4] | 2019-08-13 13:31:23.000000+0000
 c000d856-3b6d-4f71-b23e-4732d0c840a3 |    [8] | 2019-08-13 13:31:32.000000+0000

(2 rows)
cqlsh:mnist_database> 
```

In Cassandra, it will record two data (predicated) result and data into table mnist1.

3. Summary

I’m so honored to have the opportunity to participate this project. Unlike the usual projects or assignments I took in the university, this project is harder and more challenging. Most of the background of this project such as docker, Javascript, flask, mnist I have never seen before. I need to do lots of research so that I can understand and even use this new knowledge. We always do group work in the university, it is my first time to solo a whole project. I feel confidence and proud of myself I can finish this project.

Because of this project, I can have a interest in the machine learning. I can take a brief look of how neural network work. Even though, mnist is a patch in the machine learning field, I’m confident to learn more knowledge about machine learning in the following time.

At last, I would like to thank a lot to prof Zhang for teaching us the background and knowledge about this project and even more about big data, and giving me this opportunity to participate in this project and have access to machine learning.