Project Report

Chendu Li

2019.08.18

Prof. Fan Zhang

Description:

This project is under the instruction provided by Dr. Fan Zhang from the MIT LIGO Lab. This project can help users to recognize digital numbers from handwriting number pictures, using the TensorFlow training model and MNIST database.

Content:

1. Lecture Content:

I learned to use some basic applications of systems and tools from Dr. Fan's lecture. Some of them are relating and necessary to the project, and others are widely used in the area of Big Data. Applications and tools include the followings:

- **RESTful API:** RESTful Web Service uses the HTTP method to execute operations and URLs to access various resources including XML and JSON. An API is called the application program interface, which was designed to utilize the HTTP protocol better.
- **NoSQL**: NoSQL is designed for a non-relational database, such as a graph, key-value, JSON, and Column Family. Cassandra is one of the most useful NoSQL languages. In the CAP Theorem, Cassandra is mainly focused on availability and partition tolerance. Furthermore, Cassandra can boost performance in proportion with the increase in the numbers of hardware to overcome the diminishing utility problem.
- **Container Technology**: Docker is the container technology I learned and used in this project. It is an open platform to build, ship, and run distributed applications. Docker was constructed with a layered filesystem, which makes the user capable of transmitting information by pushing and pulling layers from the Docker Registry efficiently. During the lecture, I learned the basic examples of the Docker CLI, and Docker Daemon, which is one of the components of Docker. Docker can create an environment that helps users to transmit

Docker Images and download other images to users' Docker Containers without install of number of applications.

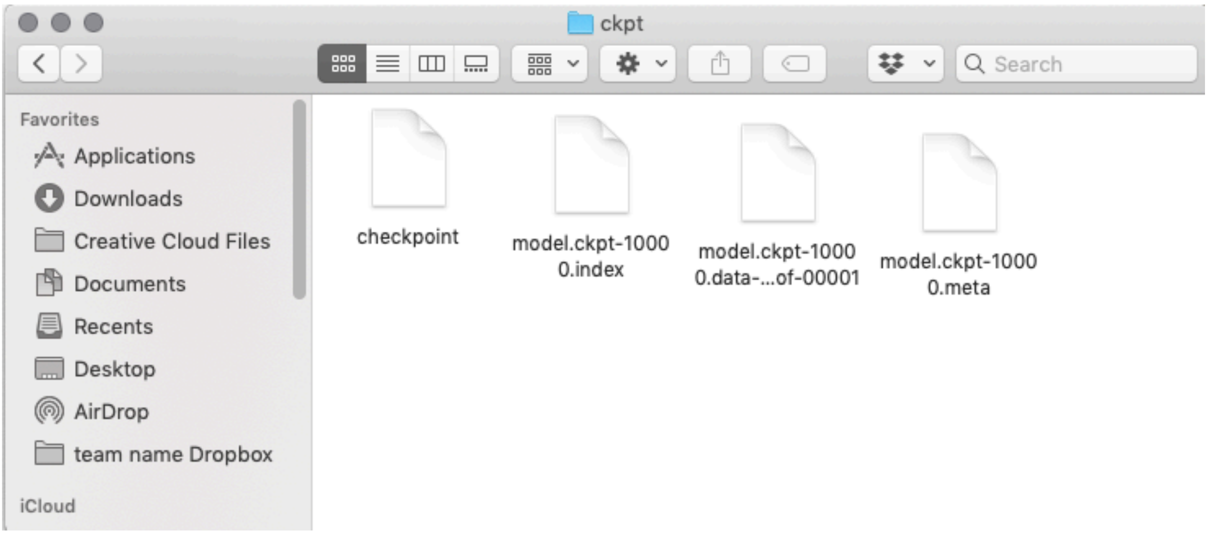
- **Flask:** Flask is a microframework written in Python. While using Docker, I also applied Flask to the project coding to help me access its extensive functions, including routing and HTTP Method.
- **Spark:** Spark improved the shuffle stage in Hadoop that increased the processing speed tremendously while supporting a wide range of languages and software. Spark can be recognized as a resilient distributed dataset. It can deal with different parallel operations in one cluster with high speed.
- **GitHub:** For this project, I created my GitHub Account to update my process and upload my works. The following is the project GitHub Link: https://github.com/lichendu/BigData-Project

2. Project:

This project is about using the TensorFlow training model and MNIST to help users to recognize a certain digit number from a picture which contains a hand-writing number. The users can use the curl- X POST command to upload their test picture (which should be in 28px* 28px) to the localhost URL address. Then, the project will use Flask to handle the picture upload requests and will run the prediction for the digit number it contains. MNIST is a large database contains more than 60,000 training images. Applying TensorFlow to the model, users can receive high accuracy digital number outputs from their uploaded pictures.

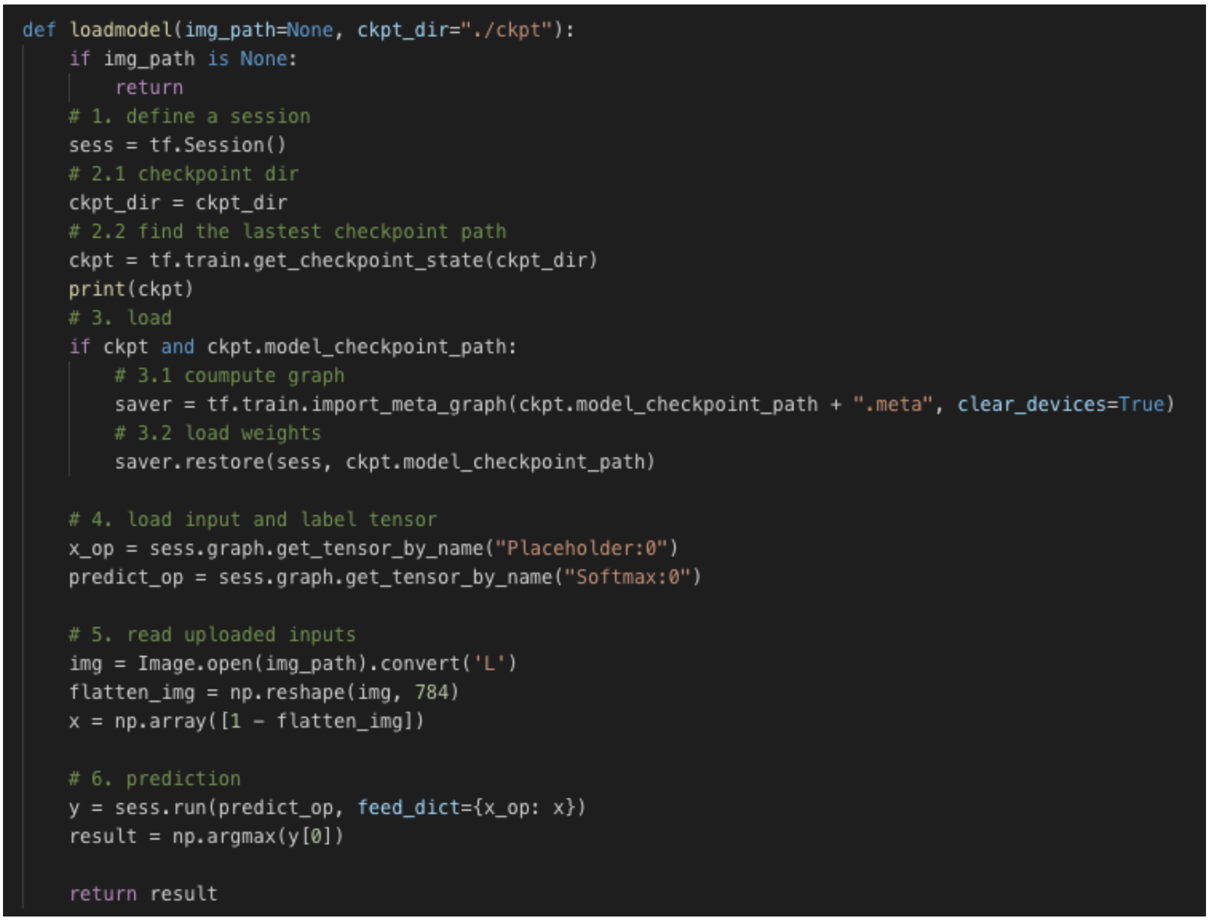
Step 1: Save and use TensorFlow MNIST Model:

The first part of the project is to save the MNIST training model as a checkpoint file in my local Mac. The following screenshot is the CKPT file used in my project.



Step 2: Design the Prediction Function

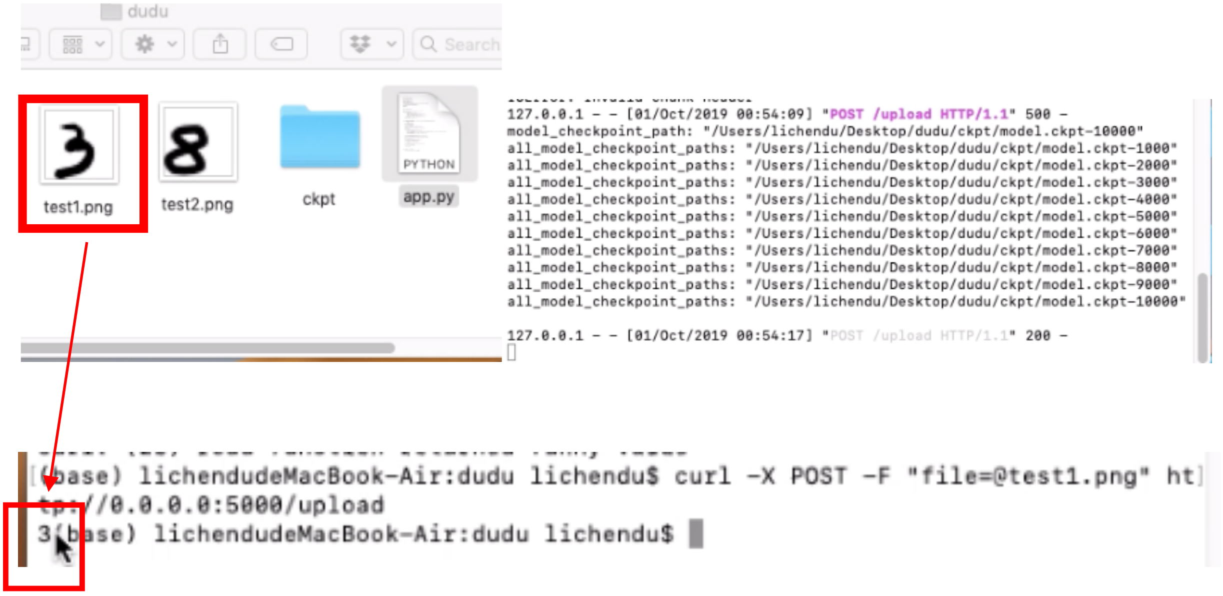
The second part is to design a function called load_model() to predict the digital number using the Tensorflow training model and MNIST database



```
def loadmodel(img_path=None, ckpt_dir="./ckpt"):
    if img_path is None:
        return
    # 1. define a session
    sess = tf.Session()
    # 2.1 checkpoint dir
    ckpt_dir = ckpt_dir
    # 2.2 find the lastest checkpoint path
    ckpt = tf.train.get_checkpoint_state(ckpt_dir)
    print(ckpt)
    # 3. load
    if ckpt and ckpt.model_checkpoint_path:
        # 3.1 coumpute graph
        saver = tf.train.import_meta_graph(ckpt.model_checkpoint_path + ".meta", clear_devices=True)
        # 3.2 load weights
        saver.restore(sess, ckpt.model_checkpoint_path)

    # 4. load input and label tensor
    x_op = sess.graph.get_tensor_by_name("Placeholder:0")
    predict_op = sess.graph.get_tensor_by_name("Softmax:0")

    # 5. read uploaded inputs
    img = Image.open(img_path).convert('L')
    flatten_img = np.reshape(img, 784)
    x = np.array([1 - flatten_img])

    # 6. prediction
    y = sess.run(predict_op, feed_dict={x_op: x})
    result = np.argmax(y[0])

    return result
```

Step 3: Upload picture to Complete

Run app.py in Mac terminal first, and then use curl – x post to upload input test1.png to localhost: 5000. We can see the output is 3. For more running process details, please see the video.mov in my GitHub link



```
127.0.0.1 - - [01/Oct/2019 00:54:09] "POST /upload HTTP/1.1" 500 -
model_checkpoint_path: "/Users/lichendu/Desktop/dudu/ckpt/model.ckpt-10000"
all_model_checkpoint_paths: "/Users/lichendu/Desktop/dudu/ckpt/model.ckpt-1000"
all_model_checkpoint_paths: "/Users/lichendu/Desktop/dudu/ckpt/model.ckpt-2000"
all_model_checkpoint_paths: "/Users/lichendu/Desktop/dudu/ckpt/model.ckpt-3000"
all_model_checkpoint_paths: "/Users/lichendu/Desktop/dudu/ckpt/model.ckpt-4000"
all_model_checkpoint_paths: "/Users/lichendu/Desktop/dudu/ckpt/model.ckpt-5000"
all_model_checkpoint_paths: "/Users/lichendu/Desktop/dudu/ckpt/model.ckpt-6000"
all_model_checkpoint_paths: "/Users/lichendu/Desktop/dudu/ckpt/model.ckpt-7000"
all_model_checkpoint_paths: "/Users/lichendu/Desktop/dudu/ckpt/model.ckpt-8000"
all_model_checkpoint_paths: "/Users/lichendu/Desktop/dudu/ckpt/model.ckpt-9000"
all_model_checkpoint_paths: "/Users/lichendu/Desktop/dudu/ckpt/model.ckpt-10000"

127.0.0.1 - - [01/Oct/2019 00:54:17] "POST /upload HTTP/1.1" 200 -
```

```
(base) lichendudeMacBook-Air:dudu lichendu$ curl -X POST -F "file=@test1.png" ht
tp://0.0.0.0:5000/upload
3(base) lichendudeMacBook-Air:dudu lichendu$
```

Conclusion:

I obtained a brand-new understanding of Big data from this project and Dr. Fan. This project taught me to use the Shell language and Mac terminal. It was the first time that I learned to use different kinds of tools, like Anaconda, Docker, Flask, etc.

It was an unprecedented experience for me to write my code. While I was working on the project, I found the database of MNIST was impressive and interesting. I found the use of Docker makes the coding process easier and more efficiently. I also found there are more interesting tools and systems are waiting for me to explore. This project not only taught me how to use computing and coding applications but also evoked my curious in this Big data trend.

In this project, I got to know the current trends of Big Data and the relating industries. From Dr. Fan’s lecture, I was introduced to the Hadoop MapReduce and its application in Google. I found that the way of MapReduce helped people to process data efficiently was interesting. The lectures gave me a new perspective on how data are utilized, stored and visualized. Applying MapReduce, Google File System, and Bigtable, Google is the most successful company in this Big data world. In other words, Data can help companies and people solve problems, and it is the system and those tools I learned from this project that helped the world move forward. Therefore, I would like to pursue further understanding in the area of Big data.

In conclusion, this project gave me extensive learning experience in all kinds of systems, applications, and tools. But more importantly, this project confirmed my interest and determination in pursuing this area.