2019.3

Nan Zhang MNIST Report

1. Introduction

The purpose of the project is to identify the number in a hand-written picture posted by a user. The program is running in a container rather than local device, which saves the storage space and promotes portability. The interaction between the users and the interface is also monitored and recorded. The project applies some tools and knowledge in the big data field. Mnist from tensorflow is used to realize the function.

2. Project Steps

install ubuntu 16.04

Install python

Spyder

Docker

Local train mnist

Cassandra

Docker build

Dockerfile and requirment

Upload docker minist\_deep run to train the docker environment change home to root

Run app.py (test past, change home to root

Connect to docker

Curl upload pic

identify numbers

Terminal Code:

docker run --name nanz-cassandra -p 9042:9042 -d cassandra

docker exec -it nanz-cassandra cqlsh

docker run --link nanz-cassandra:cassandra -p 4000:80 mnist

local: Dockerfile.txt和requirements.txt目录中docker buid -t python3-tensorflow

docker run -it --name python3-tensorflow --link nanz-cassandra:cassandra -p 4000:80 python3-tensorflow

docker inspect -f '{{.ID}}' python3-tensorflow

docker run -it --link nanz-cassandra:cassandra --rm cassandra cqlsh cassandra

describe keyspaces;

use mykeyspace;

select \* from nanztable;

docker images

sudo docker exec -it container ID bash

docker start nanz-cassandra

docker exec -it nanz-cassandra cqlsh(use nankeyspace;

Select \* from nantable;

sudo docker exec -it

curl localhost:4000/upload -F "file=@picname“

curl 0.0.0.0:4000 -F "file=@url"(can also use the browser and open 0.0.0.0:4000/html

3. Achievements

My local test does not pass but I guess the main purpose of this project is achieved beyond my expectation. Everything was new to me before, but now I am proud what I know.

I got a data platform testing position at a fintech company even though I am an accounting background student and my company is currently using DOCKER!! And I am familiar with Oracle and MongoDB now. I could not imagine it before. The main reason of me getting the job is this program and python I learnt at UIUC. I started this job on April 10th and now I have finished and pushed 3 demands.

During March and April, I finished two UIUC data anlytics course project(each takes 4 weeks with 4 or 5 people in the group): The classical EY P-card auditing using python and Excel pivot table (the main purpose is to compare the two) and the Chicago Yellow Cab big data analysis using Python and Tableau as team leader and I do most of both projects using ubuntu. I guess there is still long way to go for me to learn more about Linux.

I am clearer with my career path. I just got admitted to UTD for Econ PhD program. Now I know exactly where I am and where I want to be as an accounting background student familiar with database, programming tools and github.

4. about the course

**Class 1:**

Install Anaconda: conda install or pip install

Docker: Dockerfile and Requirement

Github: pull and push, download

Linux: Bash, Regular Expression

[1] <https://github.com/fanzhang15/pythoncourse>

[2] [https://try.github.io](https://try.github.io/)

[3] <https://www.tensorflow.org/versions/r1.4/get_started/mnist/beginners>

[4] <https://github.com/tensorflow/tensorflow/blob/r1.4/tensorflow/examples/tutorials/mnist/mnist_softmax.py>

[5] <https://github.com/tensorflow/tensorflow/blob/r1.4/tensorflow/examples/tutorials/mnist/mnist_deep.py>

**Class 2:**

Docker: Docker Tutorial <https://docs.docker.com/get-started/>

Flask: I know it is the micro-framework now but I am still a little confused what Flask is for

<http://flask.pocoo.org/docs/0.12/quickstart/#a-minimal-application>

Data type: Json、XML、CSV、Yaml

RESTful API: Post、Get and Delete <https://blog.miguelgrinberg.com/post/designing-a-restful-api-with-python-and-flask>

curl –XPOST:

<https://blog.csdn.net/huaweizte123/article/details/79672479>

**Class 3:**

NoSQL (Cassandra): active Cassandra in Docker and CQLSH

SQL Basics: SELECT \*, WHERE

Write to Cassandra: primary key, result, time stamps

Spark: RDD  
Consistency, Partition tolerance, Availability of Database

[1] <https://hub.docker.com/_/cassandra/>

[2] <http://abiasforaction.net/a-practical-introduction-to-cassandra-query-language/>

[3] docker pull [registry.docker-cn.com/library/cassandra:latest](http://registry.docker-cn.com/library/cassandra:latest)

[4] <https://spark.apache.org/>

[5] <https://spark.apache.org/docs/2.2.0/rdd-programming-guide.html>

[6] docker run --name fzhang-cassandra -p 9042:9042 -d cassandra:latest

[7] 运行docker ps，

（0.0.0.0:9042->9042/tcp）。

fanzhang@MacBook-Pro:~/Documents/projects/TestPython$ docker ps

CONTAINER ID        IMAGE COMMAND                  CREATED STATUS PORTS                                                       NAMES

728946c1ee0b        cassandra:latest "docker-entrypoint.s…"   39 seconds ago Up 24 seconds 7000-7001/tcp, 7199/tcp, 9160/tcp, 0.0.0.0:9042->9042/tcp   fzhang-cassandra

[8] import logging

log = logging.getLogger()

log.setLevel('INFO')

handler = logging.StreamHandler()

handler.setFormatter(logging.Formatter("%(asctime)s [%(levelname)s] %(name)s: %(message)s"))

log.addHandler(handler)

#from cassandra.cluster import Cluster

#from cassandra import ConsistencyLevel

from cassandra.cluster import Cluster

from cassandra.query import SimpleStatement

KEYSPACE = "mykeyspace"

def createKeySpace():

  cluster = Cluster(contact\_points=['127.0.0.1'],port=9042)

  session = cluster.connect()

[log.info](http://log.info/)("Creating keyspace...")

  try:

      session.execute("""

          CREATE KEYSPACE %s

          WITH replication = { 'class': 'SimpleStrategy', 'replication\_factor': '2' }

          """ % KEYSPACE)

[log.info](http://log.info/)("setting keyspace...")

      session.set\_keyspace(KEYSPACE)

[log.info](http://log.info/)("creating table...")

      session.execute("""

          CREATE TABLE mytable (

              mykey text,

              col1 text,

              col2 text,

              PRIMARY KEY (mykey, col1)

          )

          """)

  except Exception as e:

      log.error("Unable to create keyspace")

      log.error(e)

createKeySpace();

**Class 4:**

Amcharts, d3, Playground

Cloud Native Landscape

Spark RDD, Spark batch processing (actual demonstration of pi), Spark stream data processing (actual demonstration of network\_wordcount)

[1] <https://spark.apache.org/>

[2] <https://www.amcharts.com/>

[3] <https://d3js.org/>

[[](https://d3js.org/)4] <http://playground.tensorflow.org/>

[5] <http://redmonk.com/fryan/files/2017/04/CloudNativeLandscape_v0.9.4.jpg>

My conquest is the Sea of Stars.