Project Report

Chenjie Wu

The Ohio State University

Contents:

1. Introduction

As the information technology develops, the amount data need to be stored and process is skyrocketing, and there are lots of new technologies emerges to help deal with those problems, such as NoSQL and containerization. In 2005, Roger Mougalas, the director of market research at O'Reilly Media, creates the term “Big Data” for the first time. [1] It simply means a set of data which is so large and structurally complex, or even unstructured, that is difficult or inefficient to be processed by traditional techniques such as relational database systems. Basically, 4 characteristics distinguish big data from the other genres: Velocity, Volume, Variety, and Veracity, which means the high quality of data. Data centers use various of new techniques such as Spark to increase the computational speed and IO capacity of the whole cluster. Also, new forms of database systems such as Cassandra are able to store and process non-traditional data, such as images or video streams, better, while greatly improving the efficiency of horizontally scaling the cluster, which is more cost effective.

In this project, an application of recognizing images of single handwritten number is designed using several frontier technologies in the field of big data, including Cassandra, a NoSQL database system, and Docker, which is used to create, distribute and run applications conveniently on different platforms, from servers to developers’ devices. The purpose of this project is not only to familiarize us with the specific techniques, but also to give us more insight of big data’s future.

2. Tools

a. Docker

Docker use OS-level virtualization technologies to help build, delivering and execute the applications in various of platforms. It has characteristics of light-weight (sometimes smaller than several kilobytes), portable (able to run on different platforms) and self-sufficient (contains packages, configuration files, frameworks and application codes). The developers bundle their applications, dependent libraries and configuration files into containers, which is written into disk drives as images. In some degree, it resembles traditional virtual machines like VMWare and VirtualBox in their effects. However, a docker container replaces hypervisors in virtual machines with Docker engine, and it removes guest layer of operating system, which tremendously reduce its size and starting time. Because of that, the developers can easily distribute their images by uploading them to docker registry, from which clients pull the images and create containers from them to run the applications.

Docker uses layered file system to provide convenience for modifying the existing images. Docker image is composed of multiple read-only layers. When a container is created using this image, a read-only container layer is added to the top of the system. Asymmetrically, when the container was stopped, a “stop container” layer was added instead of simply deleting container layer.

To ease the usage of the application and provide more conveniences for the users, in this project, the Cassandra database system would be pulled from docker registry. Also, the whole application would be bundled into a container along with dependencies.

b. Flask

Flask is an open-source, lightweight web framework written in Python. It is based on REST web service style, which was designed to fit the HTTP protocol. It helps individual developers to develop several web applications. In this project, Flask would be used to receive picture input from user in command line and return a numerical prediction of that picture.

c. Cassandra

Traditional, relational database such as MySQL and DB2 has been used for a long time. In those databases, data are stored in form of tables, which is highly organized but, at the same time, restrictive. Its ACID compliance (atomic, consistent, isolated and durable) leads to its large occupation of computing resources. What's more, when doing table joining and splitting operations, the data can be fragile.

Although relational databases are highly normalized and schematic, modern commercial applications often prefer speed and availability, which is not relational databases' advantage, rather than consistency. Moreover, in the epoch of big data, more and more data are non-relational, meaning that they cannot be easily fit into SQL. Non-relational database (NoSQL) appear as a remedy for these puzzles. NoSQL distribute nodes of data across different machines, and it is advantageous in horizontally scaling (adding more machines rather than upgrade machines) compared to SQL. The 4 major forms of NoSQL enable efficient distribution, storage and approaching of image, video stream and other non-relational data.

As a representation of key-value database, Cassandra is implemented by Java and stores data in forms of keys and their corresponding values. It focuses on availability (speed) and partitional tolerance (reliability) rather than consistency, making it extremely suitable for commercial applications. Notable applications are observed on Grubhub's and Uber's data storage. Also, its IO capacity is almost linear with its extent of horizontal scale, which enables cost-effective enlargements of database.

In this project, the prediction value is not only be returned to command line, but also be written to Cassandra database system by codes using python's Cassandra API.

d. TensorFlow

TensorFlow is an open-source deep learning framework developed by Google Brain Team. It is widely used both for research and production purposes. In this project, TensorFlow is used to import MNIST data, training, saving, restoring models and make predictions.

e. MNIST

MNIST is the abbreviation of "Modified National Institute of Standards and Technology database". It is sometimes called "the Hello World of machine learning". MNIST database contains 50000 training images and 10000 testing images. Each of them is in 28*28 pixels and represents a written number. Along with each image is a one-hot encoded label representing the number corresponding to the image. In this project, MNIST database and its official sample code were used as a basis of training the prediction model.

3. Implementation

Firstly, before publishing the project, the model is already trained in model_training.py for 100000 times and saved to /models/model1/handwritten1:

```
…
train_step.run(feed_dict={x: batch[0], y_1hot: batch[1]})
…
address = '/Users/wuericchenjie/Python3Projects/THE_Project/models/model1/handwritten1'
save.save(sess, address)
```

On MacOS, user upload an image of handwritten number onto localhost:

```
$ curl -X POST -F "image=@number_2.jpeg" "http://0.0.0.0:8000/upload"
```

This snippet of code handles the incoming image, and save it to the project folder:

```
img=request.files['image']
saved_file_path = 'inputGraph.png'
img.save (saved_file_path)
```

Then, image is resized, greyscaled, digitalized, vectorized and flattened to match the format, using process_image () method:

```
normalized_pv = process_image(saved_file_path)
```

Then, perdict () method is called to use existing model to perdict image:

```
result = perdict (normalized_pv)
```

Information of each upload and perdiction is logged into cassandra database:

```
cqlsm.insert_data(session, file_name, current_time, result)
```

Numerical prediction result is returned to command line:

```
return result
```

4. result

Code is unable to run…

5. conclusion

In conclusion, the application is able to train the model and produce the result, although the accuracy needs further improvements.

By the way, this project not only points direction for me to learn how to implement RESTful service using Flask, how to use docker to enable cross-platform distribution of application, how

to use models to make prediction with TensorFlow, and how to use Cassandra database in docker, etc., but also provides me with deep insights of future trend of big data. Before attending this project, I had almost no knowledge of Python, and various of notions were completely new to me: RESTful API, HTTP methods, neural networks, containerization, etc.. Big data was also a brand-new concept for me. Going through these six weeks of online lecture, self-learning, project planning and implementation, they are not strange to me anymore.

The most astonishing moment for me was when I hear Mr. Zhang says that a container can be as small as several KBs. Though I had never used a virtual machine, I saw my friends installing them on their MacBooks just to run SolidWorks. A bare virtual machine already occupied tens of gigabytes of disk spaces and several gigabytes of memory. I got even more excited when I heard that a container can be distributed and run anywhere as long as docker is installed, because I had struggled to install OSU CSE components and dependencies on my different devices.

In addition, this project helped me better comprehend how the skills and knowledge our colleges teach differ to the trends in industry. For instance, this is the first time I learnt that JSON, which can be parsed by JavaScript, is used more and more as a structured file format. In comparison, my teacher is still teaching us tree structure using XML and the non-native, hard-to-use XMLTree class in OSU CSE components of java. Also, I learnt how NoSQL database works, and how to use it along with containerization technologies, while my school do not offer such a course.

[1] Rijmenam, Mark. “A Short History Of Big Data”. Jan, 2019. [online] Available: https://datafloq.com/read/big-data-history/239 [Accessed Oct. 14, 2019]