University of Melbourne

COMP90024 Assignment 2 Report

Investigation of the livability of Melbourne based on education and housing among centre, west and east Melbourne

Wei Ge - 1074198
Han Wang - 1041260
YanBei Jiang - 1087029
Yiwen Zhang - 1002781
Zening Zhang - 1078374

Team 03

An Assignment submitted for the UNIMELB:

COMP90024 Cluster and Cloud Computing

May 16, 2022

Contents

1 Introduction

Melbourne is one of the most famous cities in the world. Melbourne was the world's most liveable city for eight years in a row. However, in 2012, Melbourne is "only" ninth. Although it is a good ranking, some problems may happen in Melbourne. Twitter is one of the most popular social applications. There are many people in Melbourne who use Twitter. In this project, we will develop a cloud based solution to analyze twitter data.

In section 2, we will introduce the general system design and the architecture of the whole system. Also, we will simply explain the technology stack as well as the advantages and disadvantages of them.

In section 3, we will discuss the deployment, which will be divided into three parts. The first part is about MRC, UniMelb Research Cloud, which we used as the cloud system in our project. The second part is the allocation of instances. We have four instances to work together with different roles. The third part will explain the Ansible auto-deployment. This is the method we used for automation deployment.

We will discuss how we collect data and process them. We have three kinds of data, the data from Aurin, 10 GB tweets supplied by school, and tweets we get from Twitter by tweepy. Data from Aurin will be used as the professional data to illustrate the house price and education level. 10 GB tweets and tweets from tweepy will be used to analyze people's sentiment. For the data processing, we will use NLP, Natural Language Processing to do it.

Section 5 will introduce the method we used to store data and extract data. To store data, we will use couchDB as the database to store it. And we use MapReduce to extract data for more convenient use.

Section 6 provides a specific user guide of the system capabilities, while section 7 will explain the application scenario. Different scenarios will explain different observations.

Section 8 will show the error handling we used in our project. And section 9 will discuss some challenges we meet during the process of the project as well as some creativity for this project.

2 System Design and Architecture

2.1 Whole system architecture

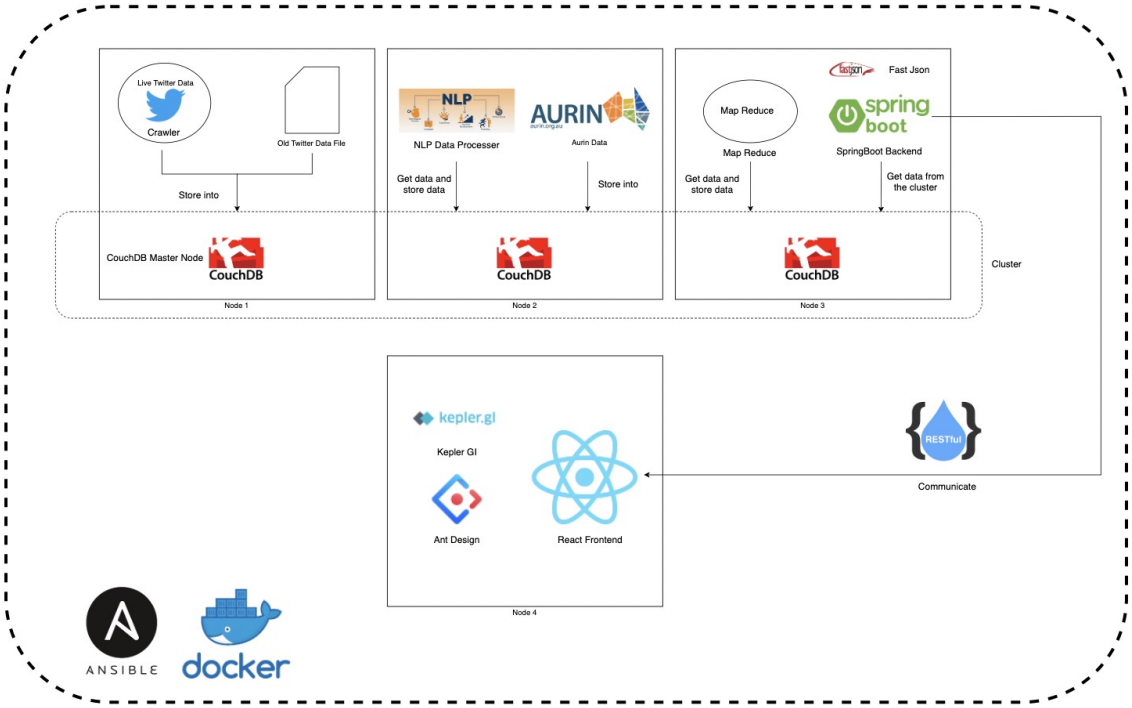


Figure 1: Whole system architecture.

There are four nodes in this cloud application, as shown in the Figure 1

In Node1, the CouchDB master is positioned. While only the master can accept data and share it inside the database cluster, the crawler and large file reader are deployed on the same node to communicate with the master without network transmission. This design can reduce the cost and delay of the web because the crawler and large file reader will store data very frequently.

In Node2, the NLP Data Processor is allocated. Because the data processor needs to process the endless data through a complex algorithm, this node is its proper node. Moreover, the Aurin Data is small and already be processed when downloaded, thus the Aurin data is read on this node. In addition, all the readers run under the multithread.

In Node3, the MapReduce Runner and Backend are deployed. As our design, MapReduce will be called per hour to count the processed data and store the result into the cache database for Backend to use. This means MapReduce will not keep running every time with a large load. While this application's Backend is also not complex enough to cause high load, they are positioned together. Furthermore, the Backend will use multithread to handle requests from the Frontend.

In Node4, the Frontend is built. It owns the whole node because this project will exhibit lots of diagrams and complex maps for each visitor. As the number of visitors increases, the Frontend's load will be high.

2.2 Backend

It should be easy to develop and deploy to be a suitable backend for the cloud computing application. From that, this project is built by Java in Spring Boot frame while it contains several advantages.

Firstly, Java is popular enough to be understood by the whole team, and the powerful tools simplify the coding part in Spring Boot frame. The comments of Spring Boot motives the development to be fast and straightforward, even for our teammates who are without backend development experience. Moreover, instead of installing the complex environment, the Maven provides a convenient way to import and integrate the third-party libraries. For the deployment, the Spring Boot contains the tomcat server internally to prevent the developer from spending lots of time setting up the server on the cloud node. When the application needs to be deployed on the cloud, the whole project and required tools can be directly packaged by the Maven, followed by positioning only one .jar file on the cloud node.

However, there are several disadvantages to using Spring Boot. Although Spring Boot helps the developer avoid complex environment setup, the downloading time for java running environment will take a long time. Especially this project uses docker to run the backend, and each deployment will rebuild the running environment. However, this problem can be ignored because this project's backend does not contain complex logic. Therefore, basically, there is no bug occurs to causes the docker to redeploy.

2.3 Frontend

2.3.1 Kelper.gl

Kepler.gl is a data-agnostic, high-performance web-based application for visual exploration of large-scale geolocation data sets. The reason that we chose it as a tool to visuilze our Twtiier data since it can fit our need the best. It is a power tool that allows visualization of geo-data on the map, which can help us easily figure out the interesting features. And also, by providing the time stamps data, we can show some interesting animation about the twitter changes. However, due to some lack of information of Kepler.gl, it is a little bit hard to compute during the development and it costs a lot of computer resources to run this pack.

2.3.2 React.Js

React.Js is used for this project to help build the frontend website that shows the analysis results with diagrams and animations. By using react, we can easily separate the frontend and the backend. On the website, the user can easily see the data on the Kepler Gl and change the parameters on it to play around. And also, with libraries like "rsuite", "antd" and "reaviz", the data can be shown in the Pie chart, Bar Chart, and Multi Bar chart easily. React is a JavaScript library focused on creating declarative user interfaces (UIs) using a component-based concept. It's used for handling the view layer and can be used for web and mobile apps. It is easy to start with and there is a huge amount of third-party libraries to help make the development process easier. However, due to the updating of React.Js, it is easy to encounter problems like dependency problems. This brings many incapable warnings and errors during implementation, and sometimes not clear warning descriptions making the bug fixed periods becomes harder.

2.4 The communication between Backend and Frontend

The Restful design is applied in the communication between the frontend and backend. As one of the benefits of using Restful design, the required HTTP protocol can guarantee the information's security to be transferred. Moreover, the interaction under the Restful rule is stateless, the Server must respond to all kinds of requests from Client. While the database may be empty or lost when the nodes all crash, the Server's response may be different. Thus the stateless interaction can simplify the logic in both frontend and backend. In order to express the situation in response, the status code is used while code 200 means success and code 403 means no source can be used.

2.5 Database

To serve as the database for the cloud application, it should be reliable, fast, and easy to scale. In this project, the **CouchDB** is chosen because it contains several advantages.

Firstly, while this project focuses on tweets and aurin data, the relationships between each row can be ignored, and all the data are stored in the Json format, thus the no-SQL and document-based databases are suitable for this project.

As one of the document-based databases, the CouchDB provides the approach to visit the data through curl, there is no need to discover the third-party support to connect the CouchDB, which is user-friendly in cloud development because different cloud nodes may use many kinds of languages. Moreover, the automatic clusters' generator of CouchDB satisfies the data backup to prevent the data missing from sudden node's crash. Furthermore, the tweets' amount will be large and contains the duplicate value, the CouchDB can be easily scaled and avoid repetition because each row's id cannot be duplicated in it (The same tweet will contain the same id).

However, the CouchDB is not perfect. With the guarantee of availability, the consistency may be lost according to the CAP theorem. For this project, the lack of consistency means different users may see different data at the same time(the backend will visit different nodes in CouchDB cluster to get data). This problem can be tolerated.

2.6 Ansible

The goal of this project is to build our system on UniMelb Research Cloud(MRC), hence we used ansible to deploy our web application in an automatic way. Ansible is an open-source tool for application-deployment which can enable infrastructure as code. It is a simple but powerful automation for multi-platform computer support. Ansible is mainly focused on IT professionals, who use it to deploy application, update server and cloud provisioning. Because of the fact that Ansible does not need any agent software and additional security infrastructure, it is convenient to deploy. Section 3.3 shows more details.

2.7 Containerization (Docker)

In this project, we mainly used containerization technology, or more specifically, docker, to set up the environment and deploy the system.Today, containerization has become popular and widely used in software development. Before containerization, people used Virtual Machine(VM) to virtualise our hardware so that we can build multiple applications on it. However, VMs have some shortcomings, it creates a whole operating system whenever we have different environments and applications, this will definitely bring a lot of overhead and waste the disk space as discussed on COMP90024 lectures.

In contrast, containerization encapsulates each part of code and its dependencies into a seperated, isolated “box” which we called docker-compose as we shown in figure 1. Inside these boxes, there is no operating system and they share a single host OS. Therefore, containerization is more light-weight and less startup time.

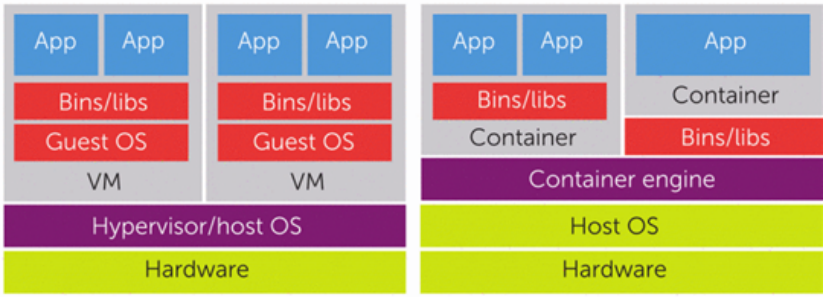


Figure 2: Virtualization architecture. The left is VM and the right is Container.

And also, migration could be another problem on VMs. As we package our code from different operating systems, such as Windows or MacOS and run it on the virtual machine which might be a Linux OS, this might cause the compatibility problem, namely code works on my own computer, but not on the MRC. Instead, containerization abstracts the application away from the Host OS and they set up the environment and install the dependencies by pulling an image from the DockerHub, so that we don’t need to worry about how to install some packages and compatibility problems, docker did all of the things for us. In our project, we also integrated docker inside Ansible, with just one line of code, we could deploy all of our applications on MRC easily.

3 Deployment

3.1 MRC

MRC(UniMelb Research Cloud), which is a kind of Cloud system, is used in our project. As we know, cloud is a convenient model for sharing a pool of configurable computing resources.MRC is the cloud computing infrastructure of University of Melbourne and the cloud infrastructure is implemented by OpenStack.

3.2 Instances Allocation

In this project, we used totally four instances to deploy our application and each of them had 100GB volume mounted. For security groups, we simply opened ports from 1 to 10000, this including port 22 for ssh, port 80 for http access, port 5984 for couchdb, port 8888 for backend and port 3000 for frontend. Figures below shows the details

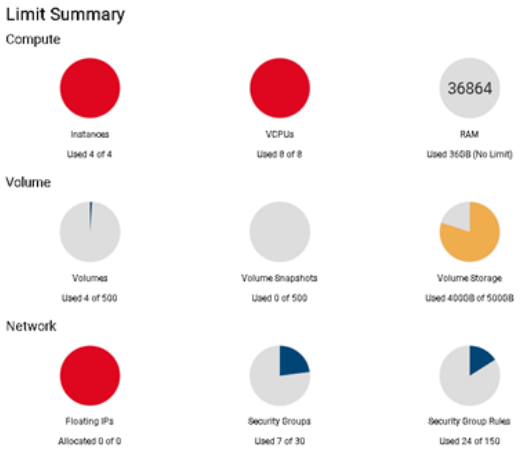


Figure 3: Virtualization architecture. The left is VM and the right is Container.

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone
instance-4	NeCTAR Ubuntu 20.04 LTS (Focal) amd64	172.26.133.159	uom.mse.2c9g	id_grp3	Active	melbourne-qh2-uom
instance-3	NeCTAR Ubuntu 20.04 LTS (Focal) amd64	172.26.131.98	uom.mse.2c9g	id_grp3	Active	melbourne-qh2-uom
instance-2	NeCTAR Ubuntu 20.04 LTS (Focal) amd64	172.26.134.178	uom.mse.2c9g	id_grp3	Active	melbourne-qh2-uom
instance-1	NeCTAR Ubuntu 20.04 LTS (Focal) amd64	172.26.133.54	uom.mse.2c9g	id_grp3	Active	melbourne-qh2-uom

Figure 4: Virtualization architecture. The left is VM and the right is Container.

For each of four instances, we balanced our workload to these nodes. Instance 1 is used as CouchDBMaster where it is responsible for reading and saving data from Twitter API and 10GB historical data. As we used Coudb cluster in our system, therefore instance 2 and 3 are couchdb slaves. Moreover, instance 2 is also used to do the NLP analysis on the tweets data. The NLP workload is quite high, so we plan to only do analysis on this node. Instance 3 is used to run the backend server and map-reduce. Instance 4 is used to only run the frontend since we need to ensure the response speed.

```
[CouchDBMaster]
172.26.133.54

[CouchDBSlave]
172.26.134.178
172.26.131.98

[Analysis]
172.26.134.178

[BackEnd]
172.26.131.98

[FrontEnd]
172.26.133.159
```

Figure 5: Virtualization architecture. The left is VM and the right is Container.

3.3 Ansible Auto-Deployment

Ansible is about automation, which requires a simple script to finish it. In our project, in order to achieve an automation deployment and four instances to use together, we choose Ansible as a tool for deployment and setup of the whole system. Ansible Playbook is used to define our deployment steps, overall, a total of 13 tasks are part of the ansible-playbook. As shown in figure 2, the full overview of the ansible-playbook tasks has been created for this project.

The tasks should be executed in the following steps:

3.3.1 Launch Instance

The first step is launching the instance on the MRC. This is done by first installing the related packages such as pip, openstacksdk to our localhost, so that we can communicate with MRC. And then we define the volume and security group that we need, followed by setting up the instance. This step involves four roles: *openstack-common, openstack-instance, openstack-security-group, openstack-volume.*

3.3.2 Install dependency

The second step is to install all the basic packages and plugins such as pip, setuptools to the instances so that they could install other packages. This corresponding role is *common*.

3.3.3 Mount volume

In this project, we need a lot of data to do analysis and this data is crucial to us, so large, persistent data storage is needed. Mounting volume to instances solves this problem and it is defined in the role

mount-volumes.

3.3.4 Configure docker

The fourth step is install and configure docker/docker compose so that we can use docker in the next few steps.

3.3.5 Git clone and git pull

We also integrated Git with ansible, in order to automatically update our code to the instance. The corresponding roles are *git-clone* and *git-pull*.

3.3.6 Setup CouchDB and setup cluster

Deployment and configuration of CouchDB and CouchDB-cluster and the roles are *couchdb-setup* and *couchdb-cluster-setup*

3.3.7 Setup Data analysis, Backend, Map-reduce, Frontend

We used docker to set up all the applications in this step including setup crawler(tweepy), process data(AURIN and natural language processing), backend deployment(Spring Boot), extraction data from couchdb(map reduce) and frontend deployment(React).
We use ansible to do all of the deployment automatically. The advantage is that Ansible is more lightweight and we do not need to set up an agent on the client side. When updating, just do one update on the operator. Batch task execution can be scripted and executed without being distributed to a remote location. And also, it is written in Python, maintenance is simpler, ruby syntax is too complicated. However, Windows cannot be used as the host, we have to use a Linux instance as a host.

4 Data Collection and Processing

4.1 Aurin data collection

Aurin(Australian Urban Research Infrastructure Network) is a national network where we can find some data in academic, government and private sectors. In our project, we use the data from Aurin to compare with the data from Twitter in order to establish the livability of Melbourne. We collect two kinds of data. The first one is shown in Figure 3, we get the education level of male and female in Great Melbourne and the rest of VIC. This data is published by GCCSA (Greater Capital City Statistical Areas) on "GCCSA-G46a Non-School Qualification-Level of Education by Age by Sex-Census 2016"

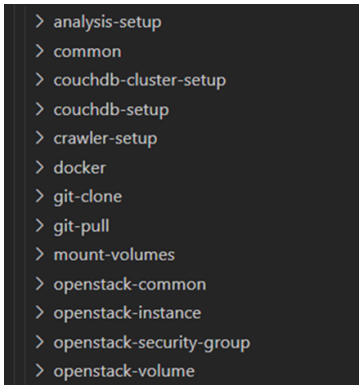


Figure 6: Data from Melbourne.

We also get the data for the house price including new house, other residential building and new residential building for Melbourne and rest of VIC, Sydney and rest of NSW, Brisbane and rest of QLD, Adelaide and rest of SA. We get this data from *dataset-AU-Govt-ABS-abs-building-approvals-gccsa-2017-18-gccsa-2016* which is published by ABS. For example, in Figure 4, is the data of the rest of QLD.

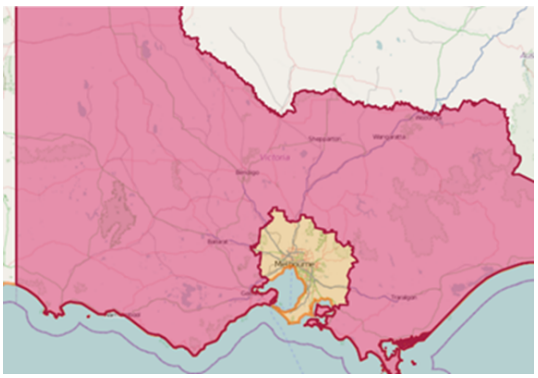


Figure 7: Data from QLD.

Figure 8: Data from QLD.

4.2 Tweepy API

According to the specification of this project, most of the data needed should be from the tweets. Thus, the Tweepy should be used to stream or search the data. To achieve this goal, the whole process of crawling data is separated into three-part and stored in the file named crawler:

- Tweepy application
- Filter setting
- Search and stream approach and saving

4.2.1 Tweepy application

Tweepy, as a tool from the official Twitter, is an open-source Python package that gives the user easy and convenient access to the Twitter API with python. However, the Tweepy has different levels with different endpoints. The essential level has 1 environment per project and the tweets crawled contain limited information which is not reaching our expectations for our further data analysis and NLP process. Thus, the next level, an elevated level with 3 environments per project and tweets crawled containing full information, is applied to have the maximum access and usage. In addition, to prevent any issue or limitation during crawling and project process. Another two Tweepy accounts were applied as standby accounts.

4.2.2 Filter setting

With the popularity of Twitter, billions of tweets are posted during the data. The tweet which is relative to the topic is only a few of them. Besides, the stream or searching rate has limitations. The especially in-stream operation, which is real-time crawling, therefore, the stream operation cannot crawl every single data posted every second. Thus, the filter should be set to restrict the stream operation to get the data needed. Fortunately, the searching and stream have a built-in function for users to filter tweets. In the filter setting, there is 7 parameter to control the data crawled. In the streaming process, track (keyword for each tweet), location (bounding box for tweet location) and language (language used for tweets) is selected. Similar to the filter in the stream, the search filter contains the same track, language and location, to ensure all the tweets are relative to the topic. And here are the sample for the search filter and stream filter.

```
"melbourne_city":
{"languages" :["en"],
    "track":["restaurant","cuisine","education","student","university","graduation","school","tuition","rent","house","apartment"],
    "locations":[144.913581,-37.839086,144.994266,-37.781922]
},
```

Figure 9: filter for stream.

```
"Melbourne_city":
{"languages" :"en",
    "track":"restaurant OR cuisine OR education OR student OR university OR graduation OR school OR tuition OR rent OR house OR apartment",
    "locations":"-37.815041,144.964979,5km"
},
```

Figure 10: filter for search.

4.2.3 Search and stream approach and saving

In the approach of search and stream, based on the limitation of location filter,**"Bounding boxes do not act as filters for other filter parameters. For example track=twitter&locations=-122.75,36.8,-121.75,37.8 would match any Tweets containing the term Twitter (even non-geo Tweets) OR coming from the San Francisco area."**

The streamer will crawl the non-geo tweets. Thus, to save the memory for the tweets database, we need to drop the tweets without the geo-location, which are the tweets that have a null value in the place attribute. Thus, the build-in function of the stream is disassembled. A condition to detect whether the place is null is added while the stream is filtering the data.

Besides, there is an expectation that the streaming function should run for 24 hours 7 days to get as many tweets as possible because the time to start streaming will cause losing data, and multiple logins and logout actions may cause the suspension of Twitter API policy. However, if we run the stream for a week and try to save the data to the database, the size of the temporary data in cache memory local will be extremely large and may cause a memory crash issue. Therefore, the saving function needs to be added to the filter function. To make sure, as long as a tweet passes the filter level, it will save to the database immediately. Thus, the cache memory overload issue will be solved, and we can do streaming and save synchronously.

Also, as there are many tweets posted in a day and stream cannot get all the tweets posted at the same time. To increase the efficiency and number of data, multithreading is deployed. Three threads are set to crawl the tweet at the same time. To ensure they can stream and search different data. Three different keys and tokens are used from three different Twitter accounts. (one account was limited during the project, then two keys and tokens from different environments in the same account are used). Thus, more data could be collected.

Besides, it is unknown whether the Twitter API policy will suspend our accounts while search and stream API is used at the same time. And the search part will run very fast. Hence the schedule library is used to schedule the search tool to run every 0 am at Melbourne time to search the tweets from 6 days ago. Then at that time, the streaming tool will disconnect and make sure there is no more API tool is using instead of the searching tool. And the streaming tool is set to restart every 1 am at Melbourne time. The reason why 0 am is picked to stop stream is because of the number of missing data. Most people sleep at midnight so the number of missing data will be the minimum for the whole day.

4.3 Data processing with (Natural Language Processing) NLP

After we retrieved the tweets data from Twitter API and 10GB historical Twitter Data, we did some NLP on it in order to get useful information on it. In the next few sections, more detailed processing steps are discussed.

4.3.1 Sentiment Analysis

In recent years, with the rise of social media, more and more people use social media such as Twitter to tell their friends what they are thinking and feeling through posting some tweets. For example, what's people's attitude towards the housing price before and after COVID-19, how about education? There is lots of information that we can retrieve from these tweet texts. In order to have a better understanding of them, firstly we did sentiment analysis on them, in other words, predicting authors' sentiment to be positive or negative or neutral based on the text. The package used is called

nltk which is one of the most popular NLP packages. Inside this package, there is a module called SentimentIntensityAnalyzer, where it can output the positive(pos), negative(neg), neutral(neu) scores by inputting the text into it. And then, we compared these three scores and selected the highest one to be our final result. One thing to note is that we found most of the tweets' sentiment are neutral, this is not what we want. To further deduce the discrepancy between the positive and negative sentiment, we want only tweets with a very high score of neutral to be classified as neutral, thus we increased the threshold to 0.8(the range is 0 to 1).

4.3.2 Subjective/Objective Analysis

This part is predicting the author's manner of speaking, namely whether it is subjective or objective. Subjective refers to the text influenced by personal feelings, while objective refers to the text judged by facts. In our project, for example, we might be interested whether people are objective or subjective towards the house price. We used a package called TextBlob to predict this and if the score > 0.5, we classified it as subjective, and vice versa.

4.3.3 Other Information Retrieval

We did some other information retrieval on tweet objects so that map reduce could easily get the relevant statistics. Since one of the scenarios of our system(See Section 8 for details) is comparing housing price/education level of different areas in melbourne. Therefore we added several additional fields to the json objects. For example, one field is education=0/1, which means whether this tweet contains education related keywords, and one field for housing price as well. The keywords list is shown below:

```
["education","teacher","professor","tutor","student","university","g
raduation","school","tuition","middle school","primary school","high
school","diploma","undergraduate","graduate","phd"]
```

Figure 11: Education related.

```
["house price",
"rent","house","apartment","property","home","accommodation","real
estate"]
```

Figure 12: Housing related.

4.4 Map-Reduce

Map-reduce is a programming paradigm for processing enormous sizes of data, which is commonly used in generating big data sets with a parallel, distributed algorithm on the cluster. The map-reduce file is JavaScript which allows the database to use the script to read each file in the database. In the project, the back end and front end cannot receive such a large size of data and process them, a process stage to process the data to a small volume and easy to read format. Thus, it is used to recombine and generate the result from the data processed by the NLP program.

4.4.1 Mapping stage

The expectation of emission is a dictionary only with key and value. Furthermore, the key is for grouping in the reduced stage, thus the key should keep the uniqueness for what kinds of grouping data you want. And the value is the part to do the calculation. Based on the data after NLP is processed, we expect to use the date, coordinate, and Melbourne area mapping by coordinate to do the comparison between the sentiment score and polarity score. Thus, the key should be the date, coordinate, and Melbourne area mapping by coordinate and the title, and the value should be the score for this tweet. Here is the sample for the mapping function:

Figure 13: mapping function

Then the output will be like:

key:[title, date, coordinate, Melbourne arear],value: sentiment or polarity score

4.4.2 Reduce stage

The reduced result is expected as the count which is to count how many tweets including the keywords and some of the sentiment score and polarity score.

The full sample for the map-reduce function will be:

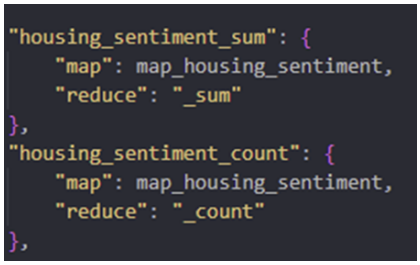


Figure 14: mapping-reduce function

sentiment and polarity of education and housing. After doing the map-reduce, the result will be formed as a dictionary and will be saved into the cache database for the backend to invoke.

4.4.3 Furthermore

To keep the data is always the newest, as the crawler and NLP are processing. The map-reduce is scheduled to run every hour to pass the newest data to the backend. To prevent the cause that there is issue or wrong data during NLP or crawling, and passed to the frontend, the previous version of the result will also be saved to do the frontend data backup. To save the memory for the cache database, it is also scheduled to delete a 24-hours ago map-reduce result file and map-reduce file by the index of the file. Hence the cache database will only contain the last 24 hours' map-reduce result.

5 Data Storing

5.0.1 How data be stored

There are three kinds of databases in this project, as shown below:

- The original database
- The processed database
- The cache database

cache_aurin	4.5 KB
cache_tweets	10.0 KB
old_tweets	2.7 GB
original_tweets	1.7 GB
processed_tweets	239.0 MB

Figure 15: information of couchdb.

The original database stores the original data from the source. The data discovered from calling twitter API or reading old data JSON file will be stored in the "original_tweets" and "old_tweets".

After getting the original data, the data will be processed followed by stored in the processed database, which is "processed_tweets" in the CouchDB.

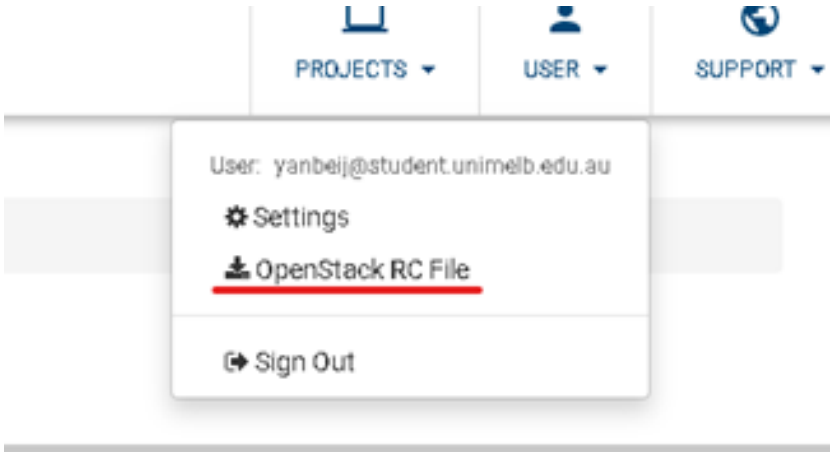
Moreover, to prevent the users from spending a lot of time to wait for the response, the Map-reduce script will be called per hour to get a result, then store the result into the cache database for the backend to quickly use. The "cache_tweets" and "cache_aurin" are responsible for that. In addition, each time the Map-reduce script stores the new result, it will remove the oldest document from the cache database to avoid the accumulation. In general, there are always 24 documents in the cache database. The reason for keeping not only the newest result is for backup and to avoid error (this will be discussed in the Error Handling part).

6 User guide

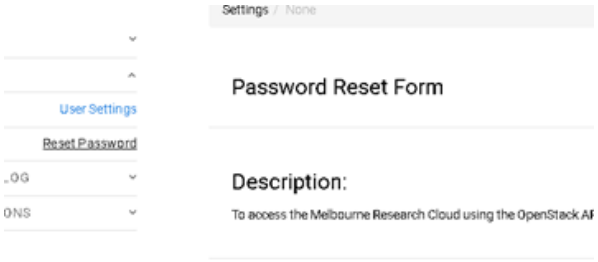
6.0.1 System Deployment

Since we used ansible to deploy the whole system, it is quite simple to deploy all the staff by following the next few steps

1. Make sure you have at least 4 instances available and at least 400GB disk space on MRC
2. Replace the file *"unimelb-COMP90024-2022-grp-3-openrc.sh"* with your own MRC configuration file by downloading it on the MRC homepage, as shown below



3. Modify the ansible configuration file at ./ansible/host_vars/nectar.yaml This file contains all the configuration information about setting up the system. There are a few configuration parameters that MUST be modified.
   (a) Generate your own MRC private key as shown below and save it to your own computer, change the field ssh_private_key_path to your own private key path
   https://dashboard.cloud.unimelb.edu.au/settings/reset-password/



   (b) You might change the volume mount or number of instances created if you have a larger capacity.

4. Run the following command

```
sudo ./launch_instance.sh
sudo ./install_dependency.sh
sudo ./deploy.sh
```

That's all the steps we need to do. Ultimately, you will start the application up!

6.1 End User Invocation

To start the webpage that we made, simply enter the following URL in the browser: http://172.26.133.159 The webpage has 3 main sections, homepage, Kepler map and data set. On the Homepage, the basic information about the assignments is shown and the information about the team is also shown. The table of the group shows some basic information like student id, name, emails and so on. This provides a basic overview of the group to the users.

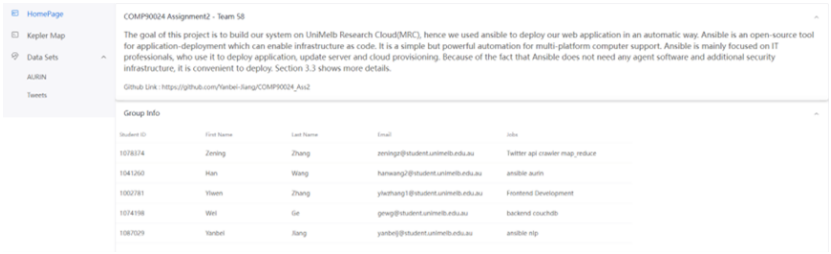


Figure 16: Frontend page

For the Kepler map section, once users click in, a map with the data set we provided will be shown. It will show the entire map and the user can zoom in or out to the section that will highlight on it.

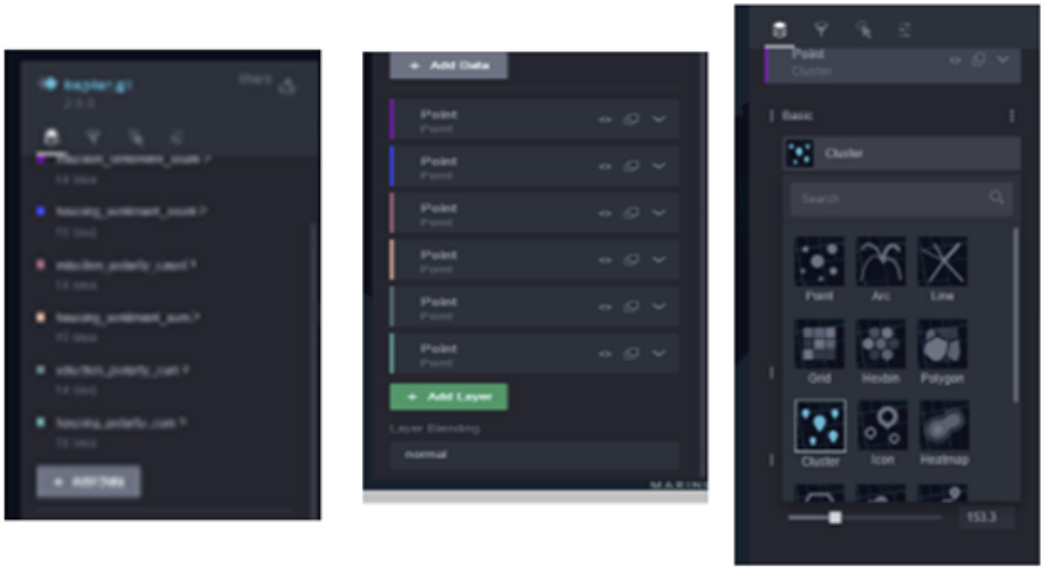


Figure 17: Frontend page

Users can click on the data set to see the data inside. The section below the "Add Data" button allows the users to change the appearance of the data. And click the filter button, and users can add filters to different sets of data. For example, they can add a filter on the Date to check the value change during the time.

Figure 18: Frontend page

In the Data Set section, it is split into two sections, Aurin and tweets. Each section has some diagrams to help visualize the result of our calculations based on the data. It is easy to understand the effect of housing and education on the livability of Melbourne. The users can put the mouse on the diagram, and a box of data information will be shown for the users to have a clear understanding of the diagram.

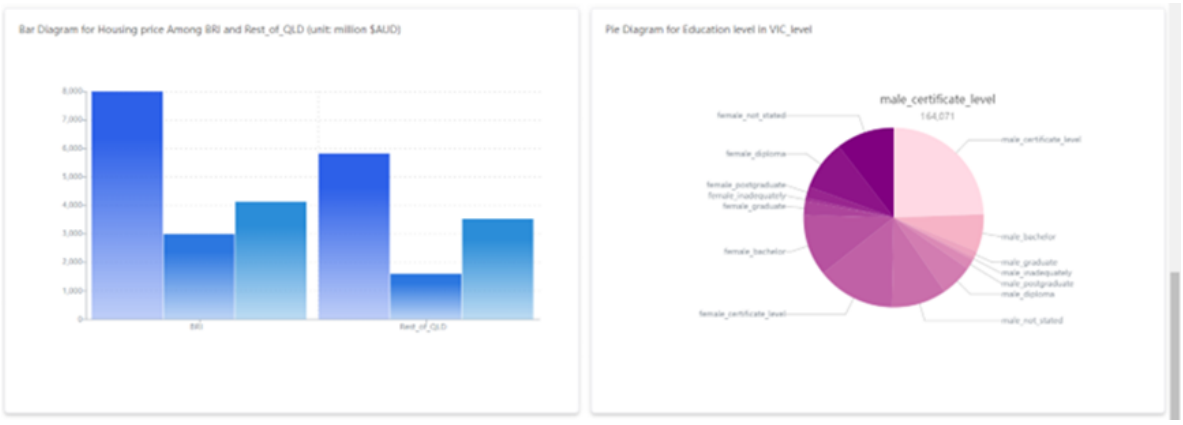


Figure 19: House price among QLD & Eduction level in VIC

7 Application scenario

7.1 Education level of Great Melbourne and the rest of VIC

This scenario describes the education level of Great Melbourne and the rest of VIC. The education level includes certificate level, bachelor degree, graduate degree, inadequately described, postgraduate degree, diploma, and not stated degree.

Why do we choose this?

Education level is an important factor which determines liveability. Relatively, a higher education level means higher liveability. The people's high level of education means the city's cultural atmosphere is relatively strong. As we can see from the diagram below, in Great Melbourne, the majority is male and female in bachelor level. In addition, there are also more people with a diploma, graduate and postgraduate degree. For people in the rest of VIC, many people are at certificate level and not stated. It is obvious that people in Melbourne have a higher education level than the rest of VIC.

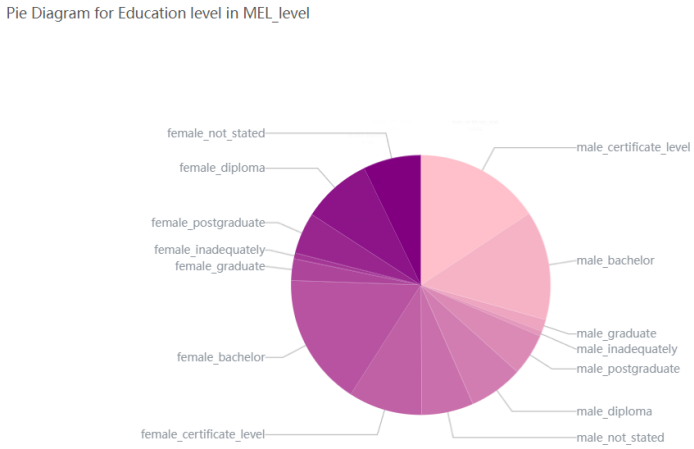


Figure 20: Education level in MEL.

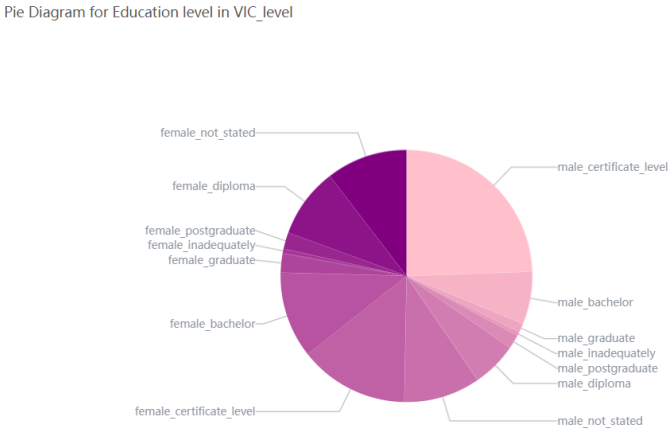


Figure 21: Education level in VIC.

7.2 Housing prices of Melbourne and the rest of VIC

This diagram shows the housing prices of Melbourne and the rest of VIC, Sydney and the rest of NSW, Brisbine and the rest of QLD, Adelaide and the rest of SA. Each part includes three types of price, total residence, new other residence and new house.

Why do we choose this?

As we can see from the diagram, the total price of residence in Melbourne is the highest which means people have relatively high living quality. Also, the price of houses in Melbourne is higher than other cities due to the fact that people can choose to live in better living conditions. This will also influence the liveability of a city.

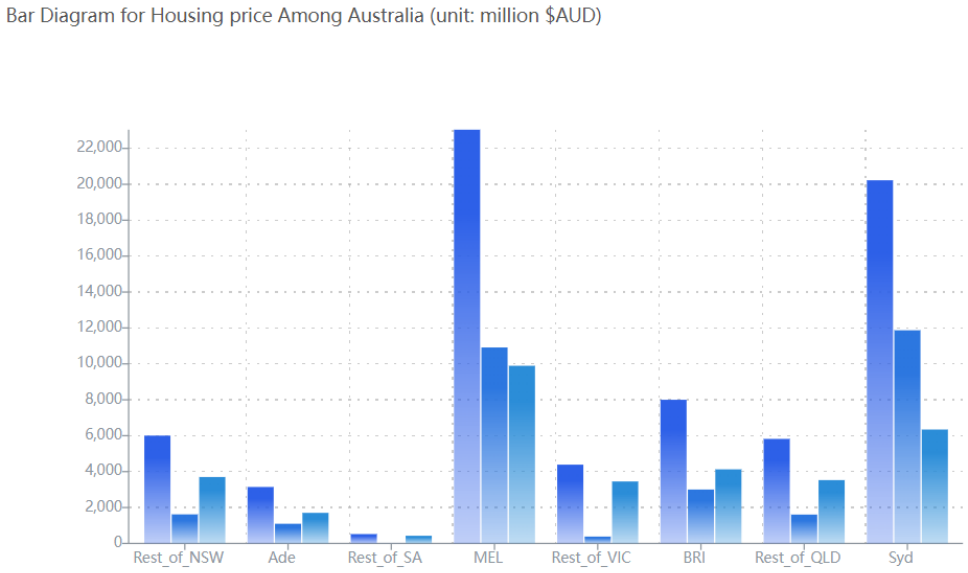


Figure 22: House price in Australia (millions $AUD).

7.3 Polarity and sentiment of the housing in the melbourne area

This scenario describes the polarity and sentiment of housing in the Melbourne area. The diagrams indicate the sum of the polarity and sentiment on housing..

Why do we choose it?

Polarity on the housing represents the level of subjectivity of the housing situation in Melbourne. The sum of the sum shows the level of polarity in that year regarding housing problems. The higher the polarity value is, it is likely that the people are more objective about the housing problems. The level of objectivity on housing problems is decreasing year by year, this shows that people are more reasonable about housing problems.

Sentiment on housing representing the level of the people's idea on the housing problem is positive or not in Melbourne. The higher the sum sentiment value is, it is likely that the people are more positive about the housing problems. The level of positively on housing problems is decreasing year by year, this shows that people are more negative about housing problems may be due to the bad economy caused by the COVID.

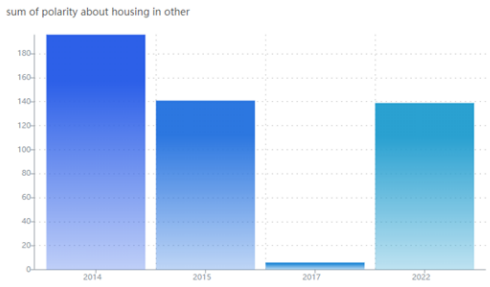


Figure 23: Sum of polarity about housing

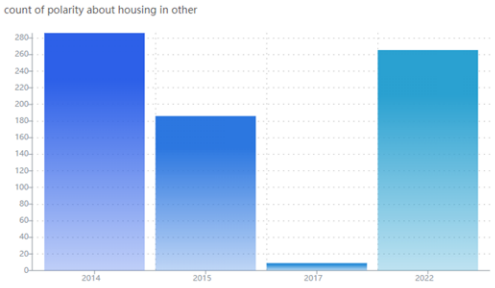


Figure 24: Count of polarity about housing

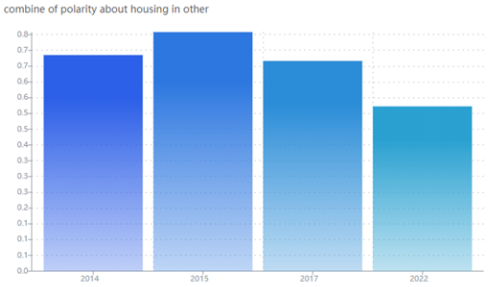


Figure 25: Combine of sentiment about housing

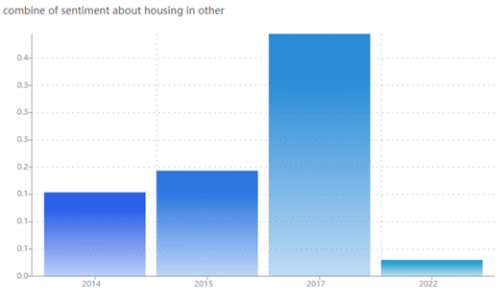


Figure 26: Combine of polarity about housing

7.4 Polarity and sentiment of education in the melbourne area

This scenario describes the polarity and sentiment of education in the Melbourne area. The diagrams indicate the sum of the polarity and sentiment on housing. The combination shows the sum over count.

Why do we choose it?

Polarity in education represents the level of subjectivity of the education problems in Melbourne. The sum of the polarity shows the level of polarity in that year regarding education problems. The higher the polarity value, it is likely that the people are more objective on the education problems. The level of objectivity on education problems is climbing after the COVID, this shows that people have more and more ideas and their own thoughts on the education problems. And this indicates, that in Melbourne, especially after the pandemic, people focus more on the education problems that we are facing.

Sentiment on education representing the level of the people's idea on the education problems is positive or not in Melbourne. The higher the sum sentiment value is, it is likely that the people are more positive about the education problems. The level of positively on education problems is decreasing after COVID time, this shows that people are more negative about education problems may be due to the online education brought by the COVID. Students and parents are feeling sad about studying from home. This may be the reason sentiment values are decreasing.

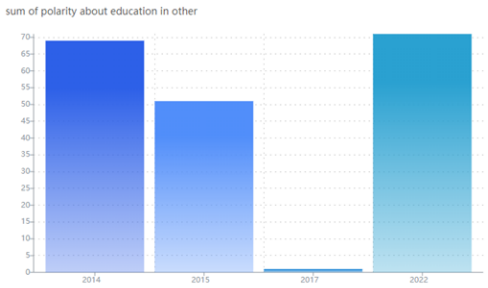


Figure 27: Sum of polarity about education

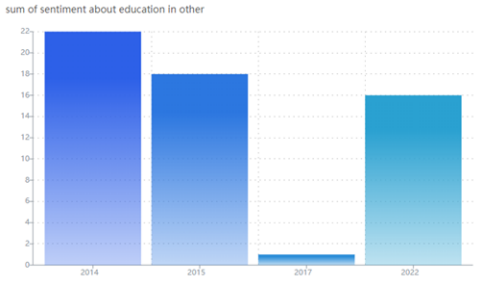


Figure 28: Count of sentiment about education

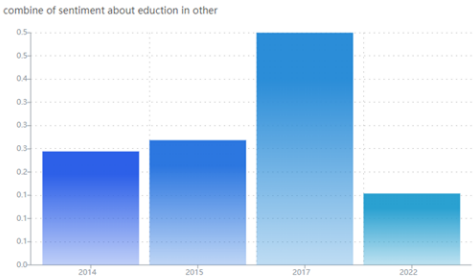


Figure 29: Combine of sentiment about education

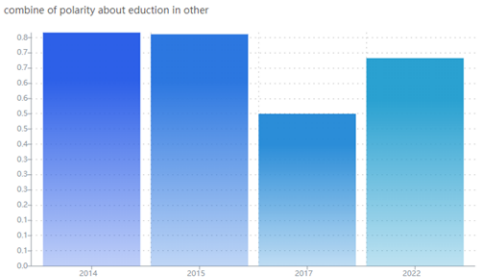


Figure 30: Combine of polarity about education

8 Error Handling

8.1 Crash error handle

In the cloud application, the unexpected node's crash cannot be avoided.

For the data backup, the CouchDB's cluster is used to store three copies of the database. For the user, a response time limit is set in the backend. If there is no response from one node in a limited time, the backend will try to visit another node from the cluster. If the backend fails three times (which means all three nodes crash), it will send the error status code 403 to the frontend with "no available database" content. Moreover, each time the backend tries to get data, it will randomly select one node to visit, which can balance the reduce the load of a single node.

Furthermore, the node's crash will suddenly stop the running scripts. In order to avoid the duplicate value generated by re-starting the script, there is two insurances used. Firstly, the CouchDB does not allow the duplicated value, which prevents the repetition from storing the same tweets twice. Secondly, the script will read a configuration file, which stores data in the database. The value in the configuration file marks that the script can start to process and store the data from which row. The programmer can modify the configuration file by checking the number of exited data in the database. This setup is for the data processor, which get data from existing database.

Moreover, to avoid the exception of the data processor or unusual data, which may influence the result be shown, there is a backup in the cache database. The cache database will contain results within 24 hours. If there is an exception in the newest result, the user can still see the typical result from the backup.

8.2 Crawler Error Handle

Error	Handle
API rate limit reach	Let the api to rest for 15 mins as long as it reach the limit.
Data type of saving	Check and the data crawled and loads them to dictionary, and drop the data that cannot change to dictionary type.
Other api error	Print the error code and print the possible error may facing.
The data cannot save to the database	If it is the key error passed that data, if it is other errors, print the error and the crawled data
Missing or wrong key and token	Print the error and let the user to check the key and token file and exit the function.
Back up issue	Print the error and let the user to check the back up key file
Check duplicates	Do not save the tweet if the id of the tweet is in the database.
Conflict on key and token using	Use schedule function to make sure search and stream function does not run together at the same time.
Cache memory overload	Add the saving function in the streaming build in function.

9 Creativity

9.1 Creativity

There are several creative parts to this project:

- All scripts are deployed by ansible and run in docker. The whole project is automated and does not affect the server's local environment.
- The data is dynamic. The Twitter crawler and data processor will not end their work. As the sample size increases, the results shown to the user will be more and more accurate.
- The backend will randomly visit nodes to reduce the single node's load, and almost all kinds of errors are handled.
- There is a cache for the backend to use to improves the user's experience.
- The Twitter crawler and old data file reader are worked under multi-thread for efficiency. The mmap method is used to read large files.
- The frontend is valid and abundant.

10 Team Collaboration

This assignemt was delivered by a gourp of hard working poeple, who working as a high degree of cooperation and helpful team. Each of the team members has their own jobs and they are all responsible for their tasks. The programming language for this assignment is diversity, such as node.js (javaScript), python, Java etc.

Throughout this assignment, multiple tools are used, such as Zoom, github, Tencent Meeting etc.

Team member	Tasks
Zening Zhang	Twitter api, crawler, map_reduce
Han Wang	ansible,aurin .
Yanbei Jiang	ansible,nlp
Wei Ge	backend, couchdb
Yiwen Zhang	Frontend Devlopment

11 Links

Github: https://github.com/Yanbei-Jiang/COMP90024_Ass2 :group03 (github.com)

Youtube: https://youtu.be/dki5nmgq5hg

Frontend: 172.26.133.159:3000

12 Conclusion

Overall, although the sentiment and polarity score about education and house after COVID does not display a good score.The Melbourne is undergoing a social turbulence especially in the house part. Despite the social turbulence does not indicate to the bad prospects. It also contains the hope and potential for growth in the future. Besides, mostly objective opinions from Melbourne citizens show that they have a intense wish to make progress to the society. In the summary, we are all bullish on Melbourne, and believe that Melbourne will finally over come this difficulty in the near future!