COMP90024 Cluster and Cloud Computing
Team 7
Assignment Report

Xi Qu
1099431
xiqu@student.unimelb.edu.au

Dongcheng Ding
952328
dongchengd1@student.unimelb.edu.au

Yifei Yu
945753
yifeiy9@student.unimelb.edu.au

Yanze Mao
988142
yanzem@student.unimelb.edu.au

Yifeng Peng
962654
yppe@student.unimelb.edu.au

Abstract

This report summarizes the work performed in Assignment 2 of COMP90043 subject. In this assignment we have implemented a cloud based system to retrieve and analyze the impact of Covid and attitude towards vaccination in melbourne cities, reflected by sentiment in tweets from these areas. Together with social-economic information of targeted areas, we could unveil the driver of those sentiments and hopefully help governments plan immunization campaigns accordingly.
In Section 2, we will walk through the system architecture design and implementation of key components. The deployment process and system functions are covered in details in section 3 and 4. In section 5, we will discuss the challenges and issues faced in the project and our solution to them. In section 6, we will review the achievements in our project and focus area for any future works.

1. Introduction

1.1 Background and Motivation

The COVID-19 pandemic has thrown the world into chaos. As highlighted by WHO: “The COVID-19 pandemic has led to a dramatic loss of human life worldwide and presents an unprecedented challenge to public health, food systems and the world of work. The economic and social disruption caused by the pandemic is devastating”[1]. What’s more concerning is the rollout of vaccination is slow in many countries due reasons like distrust or fear of side effects. In this project, we attempt to understand the covid impact to australia cities and people’s attitudes towards vaccination using tweet data, and also provide insight on the socio-economic drivers that may have shaped those sentiments.
We utilized resources in Melbourne research cloud and open-stack computing framework to build our system. The system is capable of harvesting and processing tweet data of interested areas and providing sentiment results and insights through an interactive web interface. We hope the insights obtained from this project will be useful in shaping government policies and guidelines to battle against Covid efficiently.

1.2 Related Work

There is much research and work focused on extracting insights from social media data especially twitter. Kharde (2016) provides a survey and comparative study of existing techniques for sentiment analysis of twitter and shows that machine learning methods have the highest accuracy and can be regarded as the baseline learning methods[2]. Jenifer (2020) also presented an implementation of twitter sentiment analysis framework using Convolutional Neural Network and has shown neural network approach has its superiority in this task[3].

On the other hand, the availability of geographic location based data has enabled much broader applications of sentiment analysis, allowing researchers to perform fine grain analysis of specific regions and identify correlation between observed sentiment from social media and characters of the region. The AURIN system provides thousands of multidisciplinary datasets and has made the geographical analysis much easier.

2. System Components

2.1 Overall Architecture

The logic architecture diagram of our system is shown in diagram 1 below. All the components are deployed into virtual instances in Melbourne research cloud through Ansible automatic deployment scripts. The technical design and setup of each component will be covered in subsequent sections in detail.

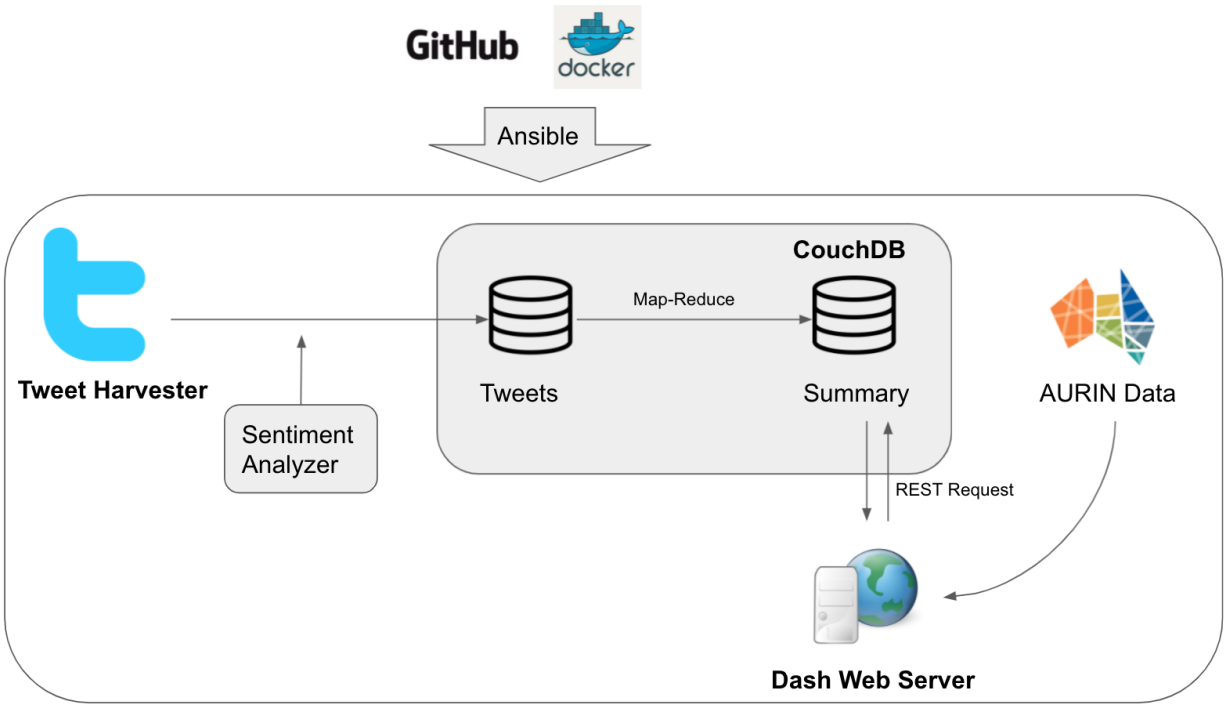


Diagram 1: System logical architecture

2.2 Melbourne Research Cloud

Melbourne Research Cloud (MRC) is a private cloud system offering Infrastructure-as-a-Service (IaaS) cloud computing capability to academic researchers. The service makes it easy for researchers to quickly access scalable computational power as their research grows, without the overhead of spending precious time and money setting up their own compute environment. There are many commercial cloud providers such as Amazon Web Services (AWS), Google Cloud Platform that offer similar services as MRC. We have compared advantages and disadvantages of MRC with those commercial clouds and also traditional on-premise systems.

- **Mission and purpose.** MRC is set up to help academic researchers with their studies and projects; the users are mainly students and researchers. On the other hand, AWS is

a general purpose cloud service that serves customers from different industries with different kinds of needs. So AWS offers a wider range of services compared to MRC.

- **Cost.** MRC is free for students/researchers at the University of Melbourne. For this project, we are allocated 4 VM instances with 8 CPU and 36GB RAM. AWS has free tier resources as well but the VM spec is restricted to 1 CPU with 2GB RAM, users have to pay to get more resources. AWS also has very comprehensive pricing options to cater for users with different needs (e.g. time critical jobs or time insensitive jobs). Moreover, compare to on-premise computing systems, cloud based systems like MRC and AWS offer:
    - Eliminate effort and upfront cost of setting up hardware and operating systems
    - Lower system maintenance costs through backup and image
    - Higher economic output through better utilization of of computing resources

- **System Stability & Support.** During the project, we have experienced intermittent errors on MRC. It appears that the system is overloaded and the network is not very stable. AWS, being the pioneer and leader of the distributed computing service provider, has its advantage of offering robust service thanks to its large infrastructure.

- **System Control.** Both MRC and AWS allow users to have full control over the instances created. In on-premise systems, usually multiple projects share the same resource, so a system admin role is required to manage the system and ensure key resources are protected. On the other hand, cloud computing services have the advantage of allowing each project to have their own virtualized resources. This eliminates the risk of interference with other projects and makes our project management much simpler.

- **Scalability.** Cloud computing has the ability to increase or decrease resources based on demand. Scalability is the hallmark of cloud computing and it's achieved through:
    - Virtualized and shared infrastructure. Unlike on-premise systems, virtualization makes it easy for users to request additional resources such as CPU/Disk/Memory without upgrading the actual hardware.
    - Distributed computing framework. The big data applications such as Spark, hive, HDFS are designed to process tasks in parallel in a cloud environment and they offer much better scalability compared to traditional applications such as Oracle Database.

2.3 Ansible

Ansible is an automation engine that is used to automate instance deployment, instance configuration, application deployment and orchestration. In this project, we used ansible to deploy and run our city analytics program on the Melbourne Research Cloud. This makes our program easily transferable and deployable among any openstack infrastructures. Our ansible playbook includes the following functionality:

- **Dynamic deployment of new instances.** New instances can be created in the MRC by specifying the number of instances to create, the size of volume to be attached to each instance and the security groups for each instance.
- **Configure instances.** Instances can be configured by specifying a list of required packages and a list of required tasks such as adding a proxy.
- **Deploy and run applications.** Applications such as couchDB and Twitter crawler can be deployed automatically by running the specific ansible playbook.
- **Delete the whole system.** The whole system can be deleted by just running a single command.

2.4 Docker

Docker is a platform that allows developers to create, deploy and run the application in an OS-level virtual environment called containers. The advantage of running applications inside the docker container is that we don't need to configure the instance anymore, the docker containerd application can be deployed in any environment. Docker container is also lightweight and fast; it allows fast deployment and uses less resources.
In this project, we have built our own dock images and published them into docker hub. The ansible script automatically retrieves required docker images from dockerhub and deploy into MRC instance during deployment.

2.4 CouchDB

CouchDB is a lightweight distributed database with an intuitive JSON API. Also it has native mapreduce capabilities to support analytical use cases. In this project, we have set up a couchDB cluster to store tweet data and perform data analysis based on the following considerations.

- It's a NoSQL database. CouchDB could handle store/retrieve of nested and unstructured data easily by storing data in JSON based document format. In particular, this suits tweet data collected in our project which is also in JSON.
- It has a fast indexing feature, Each record (document) that is stored in the database is given a document-level unique identifier (_id) as well as a revision. In our project, there are essential requirements of removing duplicate tweet records, so we utilized the document unique identifier to achieve this requirement efficiently.
- It's mapreduce capability. In our project, we need to process all the tweet data and extract insights. Instead of setting up big data analytical tools such as Hadoop or Spark, CouchDB has built-in mapreduce function through views. We have set up views to calculate the sentimental score of tweets for each zone and every month.

2.5 Twitter Harvester

Twitter Harvester is a python process built to pull tweet data via twitter API. There are two main APIs available: real time tweet streaming and historical tweet search. Based on our testing, they each have pros/cons so we have implemented both methods in our process in order to collect more data.

As shown in below diagram 2, our twitter harvester consists of:

- A streaming process that runs in the background all the time. Twitter's streaming API doesn't support filtering of location and keywords at same time, so we have implemented a keyword filter to only keep tweets of interest.
- A keyword search process that wakes up every 15 mins and performs tweet search using pre-configured keyword, location and since_date filter through Twitter's search API.
- A user timeline search process that looks up a user's historical tweets and retrieves ones that contain keywords. This is triggered when a tweet is harvested from the streaming or keyword search process. For this instance, the same user may have tweeted relevant tweets before as well.

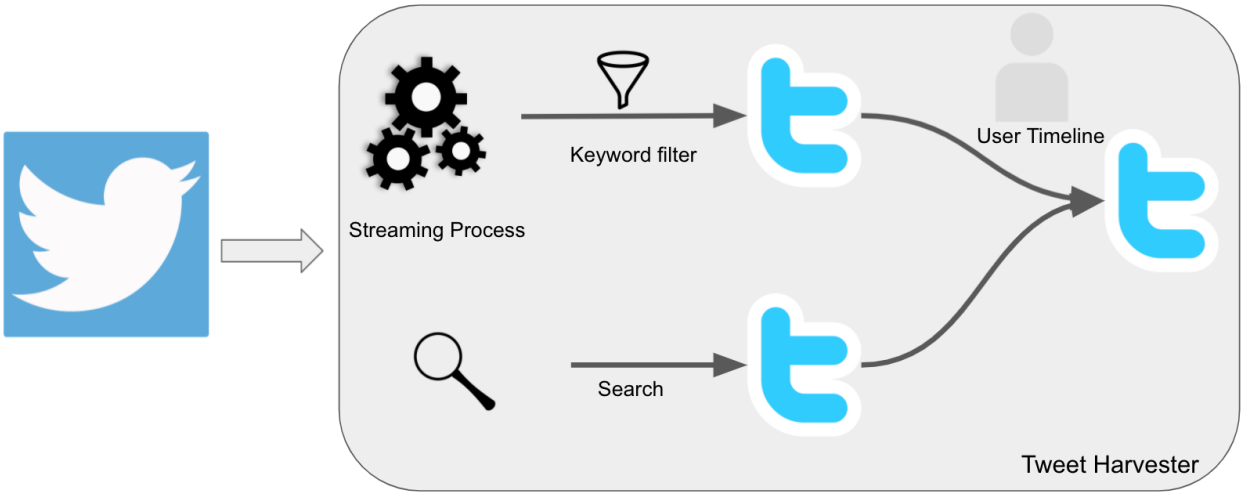


Diagram 2: Twitter Harvester Setup

Harvested tweets go through post-process steps before saved into the database:

- Duplicate checking, tweets are checked using hashed ID against existing tweets, so that duplicate tweets are ignored and not saved.
- Data extraction, the tweet json object is parsed and only key tweet information relevant to our project such as id, text, created_at is saved into the database to reduce storage load.
- Data enrichment, sentiment score and zone id are calculated for the tweet and saved together with tweet data. That could make subsequent analysis jobs much more efficient.
- Data analysis, top 10 popular words for each zone/period are found by checking newly collected records and original records every day and updating them to couchDB.

2.6 Dash/Flask

The website is powered by the Dash python framework using data extracted from couchdb. Dash is chosen as the web design platform since it is simple to use. Supported by a backend based on flask, dash can use large varieties of packages and make applications easily.[6]
It has higher compatibility with latest technologies, smaller codebase size, and higher scalability compared to django. Since dash plotly can be navigated by python 3, it is very easy to build a quick prototype and routing URL upon it.

3. User Guide

3.1 Deployment Guide

This is a step by step guide on how to use ansible to deploy the entire system onto the Melbourne Research Cloud.

3.1.1 Create Instances

Before creating the instances, we need to create an instance configuration file that is used to tell ansible how many instances we are creating, the security groups for the instances and the size of the volumes to be attached to each instance. There is a python script in the host_vars folder called ‘generate-instance-config.py’, it is used to create this instance configuration file based on how many instances to create. Once the instance configuration file ‘mrc.yaml’ is created inside the host_vars folder, we can run the ‘create-instance.sh’ which it calls the ‘create-instance.yaml’ ansible playbook to create the instances. The content of this ‘create-instance.yaml’ is shown in the table below:

Hosts	**Roles**	**Descriptions**
Localhost	openstack-common	To install python-pip and openstacksdk locally.
	openstack-images	To get and show all available Openstack images
	openstack-volume	To create volumes.
	openstack-security-group	To create security groups.
	openstack-instance	To create instances in the MRC and store instances ip addresses in the /inventory/hosts.ini file.

3.1.2 Assign Tasks to Instances

Once the instances have been created, we need to assign tasks to each instance. There is a python script in the inventory folder called ‘assign-tasks-to-instances.py’, it is used to create the ‘hosts.ini’ file that assigns tasks to each instance based on the number of instances for each task to run on.

3.1.3 Configure Instances

To configure all the instances, we need to run the ‘configure-instance.sh’, which calls the ‘configure-instance.yaml’ ansible playbook to configure the instances. The content of this ‘configure-instance.yaml’ is shown below:

Hosts	Tasks	Descriptions
instances	add-proxy	Add a proxy so that the instances can access the external internet.
	configure-dependencies	Install all the required packages.
	clone-github	Clone github repository onto every single instance.

3.1.4 Clone GitHub Repository

To clone the github repository, we need to run the ‘clone-github.sh’ which calls the ‘clone-github.yaml’ ansible playbook to clone the github repository. The content of this ‘clone-github.yaml’ is shown in the table below:

Hosts	Tasks	Descriptions
instances	clone-github	To clone the github repository onto the instances.

3.1.5 Deploy CouchDB

To install the couchDB, we need to run the ‘deploy-couchdb.sh’ which calls the ‘deploy-couchdb.yaml’ ansible playbook to install the couchDB. The content of this ‘deploy-couchdb.yaml’ is shown in the table below:

Hosts	Tasks	Descriptions
database	deploy-couchdb	To deploy couchdb inside a docker container.

3.1.6 Deploy Twitter Harvester

To deploy Twitter Harvester, we need to run the ‘run-crawler.sh’ which calls the ‘run-crawler.yaml’ ansible playbook to deploy the Twitter Harvester. The content of this ‘run-crawler.yaml’ is shown in the table below:

Hosts	Tasks	Descriptions
crawler	run-crawler	To deploy the Twitter Harvester.

3.1.7 Deploy Data Analysis

To deploy the Data Analysis, we need to run the ‘run-data-analysis.sh’ which calls the ‘run-data-analysis.yaml’ ansible playbook to deploy the Data Analysis. The content of this ‘run-data-analysis.yaml’ is shown in the table below:

Hosts	**Tasks**	**Descriptions**
dataAnalysis	run-data-analysis	To run the Data Analysis.

3.1.8 Deploy Website

To deploy the website, we need to run the ‘deploy-dash.sh’ which calls the ‘deploy-dash.yaml’ ansible playbook to deploy the dash website. The content of this ‘deploy-dash.yaml’ is shown in the table below:

Hosts	**Tasks**	**Descriptions**
frontend	deploy-dash	To pull the dash image from the docker hub, add an env file for the website and run the dash image.

3.1.9 Delete the Whole System

To delete the whole system, we need to run the ‘delete-instance.sh’ which calls the ‘delete-instance.yaml’ ansible playbook to delete the whole system. The content of this ‘delete-instance.yaml’ is shown in the table below:

Hosts	**Tasks**	**Descriptions**
instances	openstack-common	To install python-pip and openstacksdk locally.
	delete-instance	To delete the deployed instances
	delete-security-group	To delete the created security groups
	delete-volume	To delete the created volumes.

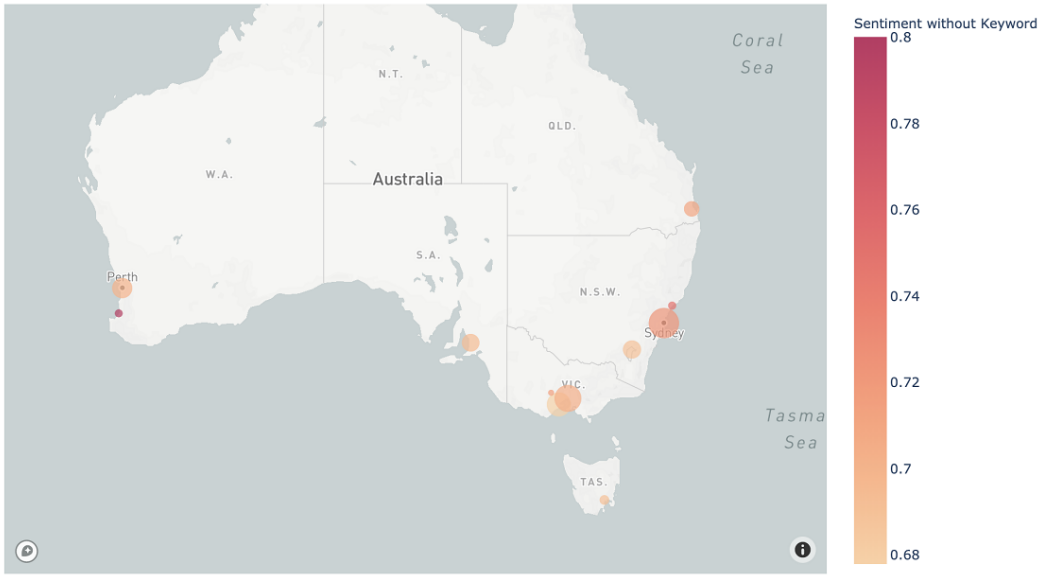
3.2 Web Application User Guide

The web application offers an interactive UI for users to view tweets and geographic insights. There are 4 components in the web page as illustrated in each subsection below. The user operations available in Web UI are:

- Change the “Map Bubble Size”
- Change the “Map Bubble Color”
- Hover mouse on bubbles in the “Background Map”
- Click the timeline of the “Monthly Top Word Bar Chart”

3.2.1 Background Map

The background information retrieved from AURIN dataset is geographically mapped with a bubble plot on map. The size of each bubble represents the magnitude of each information of the city.

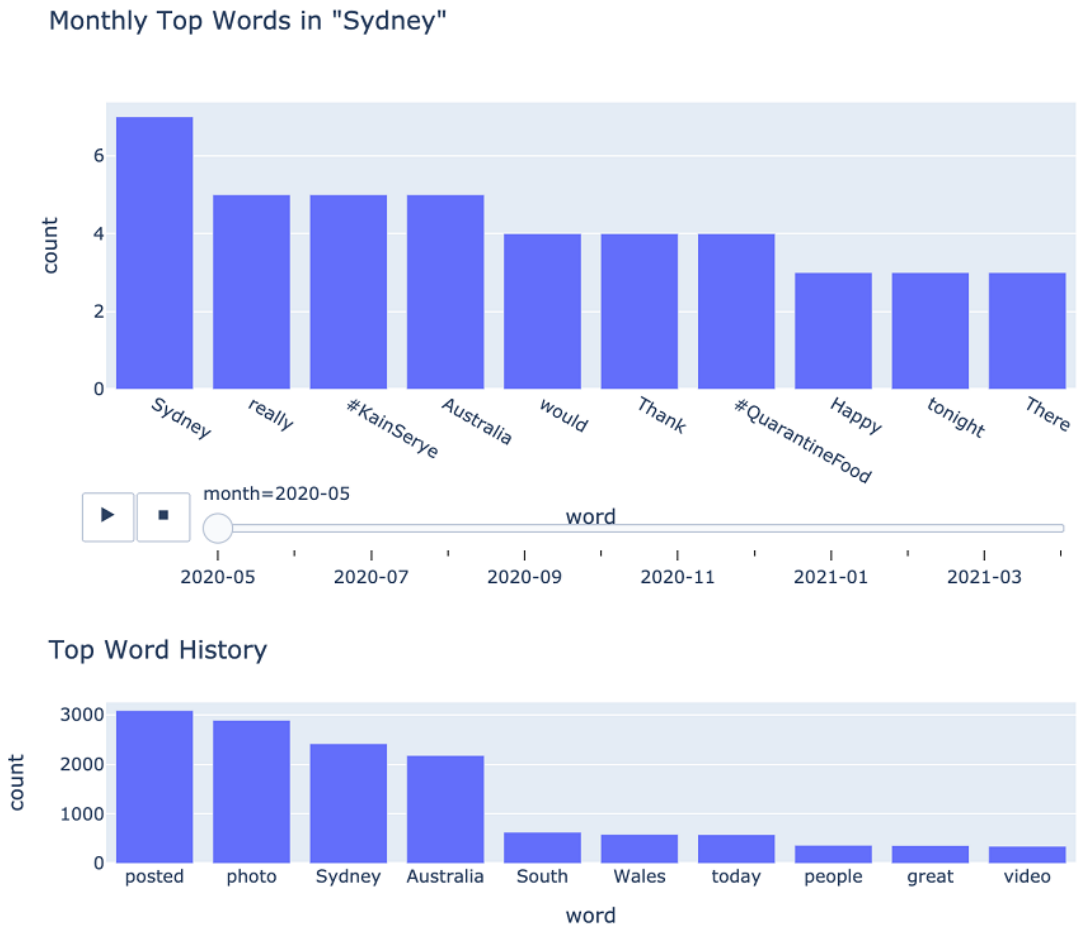


Usage:

- Select attribute of “MapBubbleSize’ to decide the size of bubbles.
- Select attribute of “Map Bubble Color” to decide the color of bubbles.
- Hover mouse on bubbles to provide geo information for other charts.

3.2.2 Top Word Charts

Based on most frequently mentioned topics in tweets from a region over a period of time.

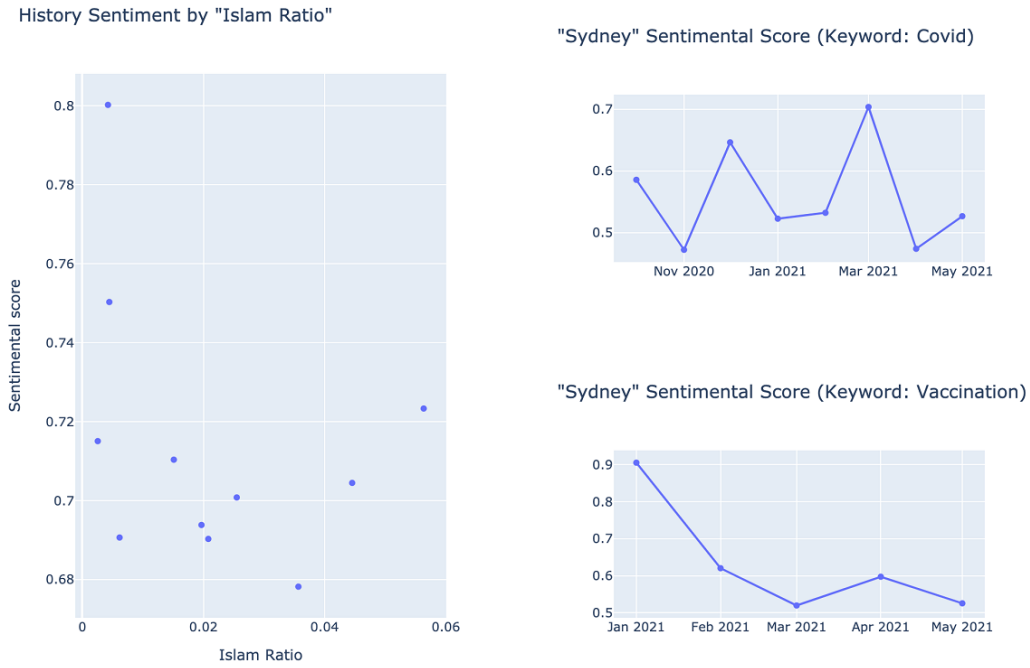


Usage:

- Viewers can use the timeline to select the period they are interested in, by month.
- The region of interest depends on the hover information in the background map above.

3.2.3 Sentimental

Based on average sentimental analysis score of tweets on topics of covid and vaccination in each city.

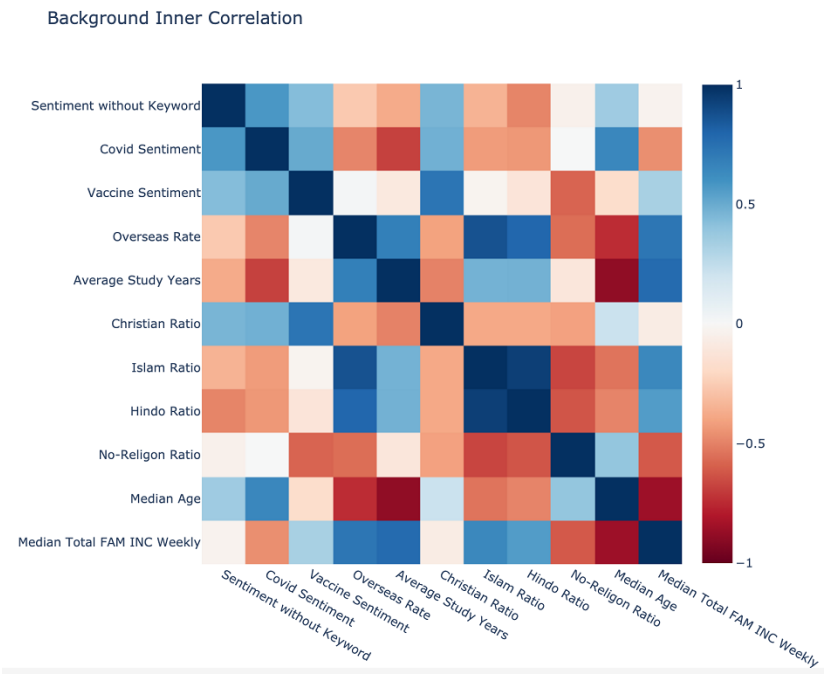


Usage:

- For the history sentiment by “*”chart, change the attribute of option “ Map Bubble Size” to decide it’s x axis.
- For the rest charts, hover mouse on different bubbles to control the Zone limitat.

3.2.4 Background Correlation Heat Map

Provide correlation between each variable examined as background information.



4. Scenario Analysis

For this project, we would like to study the impact of Covid 19 pandemic and acceptance of vaccination through analysing twitter sentiment and understand the correlation between that sentiment and other social-economic factors such as income, age, religions. This study could provide insights about the groups/areas that are mostly affected by covid and also give ideas on the focus area to improve vaccination acceptance.
To achieve the objectives, we first analyze relevant tweet data from several regions in Australia cities, which could provide us the public opinion about the topics (covid, vaccination etc) in those regions; then we obtain datasets from AURIN that provide information about social economic status of people in those regions. Overlaying these two set information, we could then generate insights.

4.1 Sentiment Analyzer

Machine learning models can be used to analyze sentiment in tweets and hence generate insights. Several NLP based algorithms and methods are compared and tested to find the most accurate method to analyze tweet sentiments:

- NLTK Vader sentimentAnalyzer, this is a pre-trained model used for text sentiment analysis that is sensitive to both polarity (positive/negative) and intensity (strength) of emotion. It's specifically attuned to sentiments expressed in social media.
- TextBlob is another popular library for processing text data. It also has a pre-trained model function that predicts sentiment of text. And similar to the NLTK Vader model, it's capable of handling negation such as 'I don't like...'
- Neural Network model, we have trained a LSTM neural network based on word Embedding. Word Embedding is a numeric representation of text with words having similar meanings encoded together, and it produces a matrix that could be input for neural networks.LSTM is well suited to process sequential data such as text as it has a feedback connection that allows contextual information to be captured and understood. Below diagram 3 shows the data pipeline for our model.

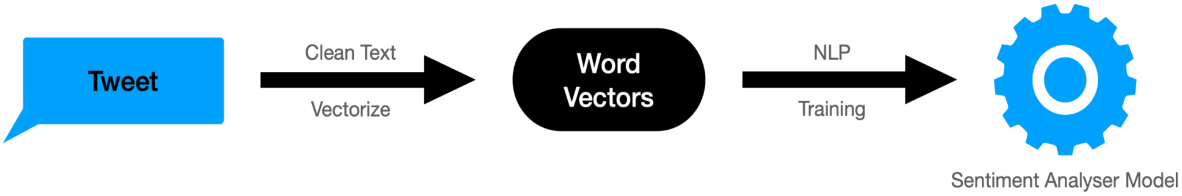


Diagram 3: Sentiment Analyzer Setup

Different activation functions and overfitting prevention methods (dropouts, max pooling) were tested and below is the final model that we adopt.

Model: "sequential"

Layer (type)	Output Shape	Param #

```
=================================================================
embedding (Embedding)        (None, 140, 300)          72980100
_________________________________________________________________
dropout (Dropout)            (None, 140, 300)          0
_________________________________________________________________
lstm (LSTM)                  (None, 100)               160400
_________________________________________________________________
dense (Dense)                (None, 1)                 101
=================================================================
Total params: 73,140,601
Trainable params: 160,501
Non-trainable params: 72,980,100
```

Sentiment140 dataset[4], a publicly available dataset that contains labeled positive and negative sentiment of tweet text, is used for the testing. It's obvious the neural network model is significantly better than the other two.

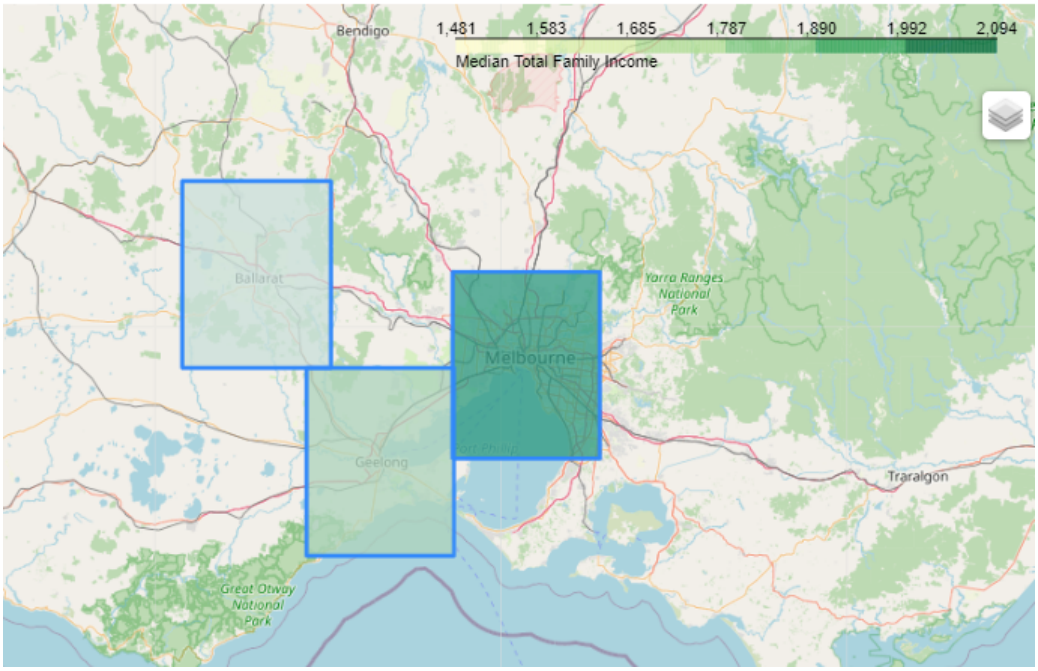
Method	**Accuracy**
NLTK Vader SentimentAnalyser	61.5%
TextBlob	60.4%
Neural Network model	81.8%

4.2 Aurin Datasets and Zones

Our hypothesis is the impact of covid and acceptance of vaccination will be affected by demographic and socioeconomic measures such as age, income, education, religion. As such, we obtained datasets from AURIN portal and extracted measures as described in below table:

Dataset Name	**Measures**
SA2-P02_Selected_Medians_and_Averages-Census_2016	Household Income
	Average age
ABS_-_Data_by_Region_-_Persons_Born_Overseas__SA3__2011-2016	Ratio of overseas residents
SA3-G14_Religious_Affiliation_by_Sex-Census_2016	Ratio of Christian/Hindu/Muslim/Free thinker
SA3-based_B16A_Highest_Year_of_School_completed_by_Age_by_Sex_as_at_2011	Average years of education

Most datasets in AURIN are at SA2/3 (statistical area 2/3) or LGA (local government area) level. Due to restrictions on twitter API and availability of geo enabled tweet data, we can't capture our tweets and perform analysis in such geographic granularity, we have to aggregate.
After reviewing the geographic measures, we have created 10 zones for our data collection and analysis. Those zones are built around different australia cities and covering both regional and metropolitan areas, the idea is those zones are large enough for us to collect relevant tweet data and in the meanwhile preserve distinct demographic and socioeconomic characteristics for further analysis. Below graph 1 is a geographic map representation of median family income for 3 zones: Melbourne, Geelong and Ballarat. As we can clearly see, Melbourne has the highest income among the 3 cities and Geelong is slightly higher than Ballarat.

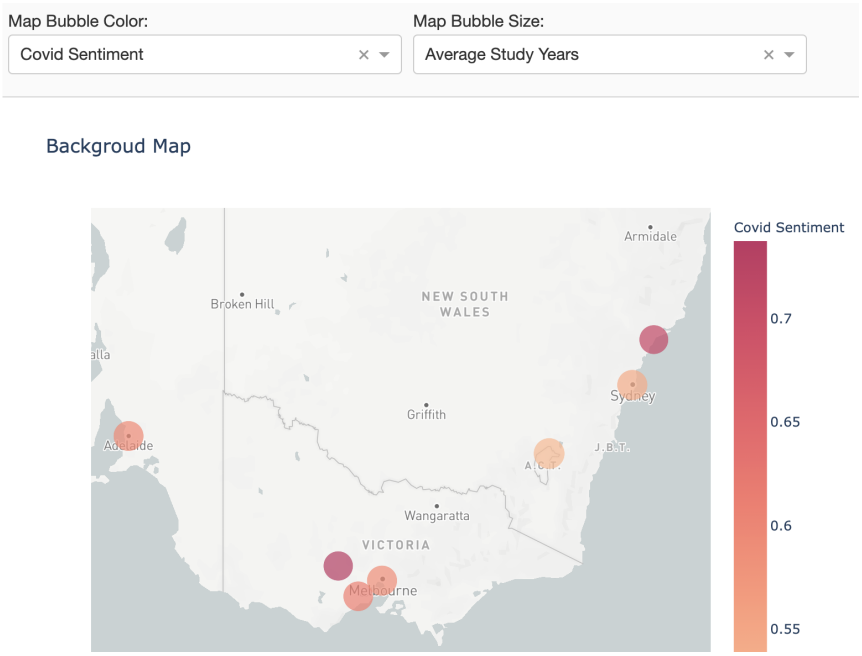


Graph 1: Geographic view of median family income

4.3 Scenario Analysis - Interactive Map View

The interactive map view provides an intuitive way to understand our measures as well as tweet sentiments for each city. By using this map view, users could not only see which city has the most significant measure (e.g. highest income, youngest population) but also be able to view their correlations easily.
As shown in the graph below, the demographic measure selected is 'Average Study Years' which is visualized through bubble size and the 'Covid Sentiment' is visualized through bubble color. The metropolitan cities such as Melbourne and Sydney are having higher 'Average Study Years' but relatively low 'Covid Sentiment' and the regional cities such as Geelong and Newcastle are having higher 'Covid Sentiment' with lower 'Average Study Years'.

This clearly demonstrates that there is a negative relationship between ‘Average Study Years’ and ‘Covid Sentiment’, which could be a good starting point for further research such as whether mental counseling is necessary to help to alleviate covid related stress for university students/researchers.

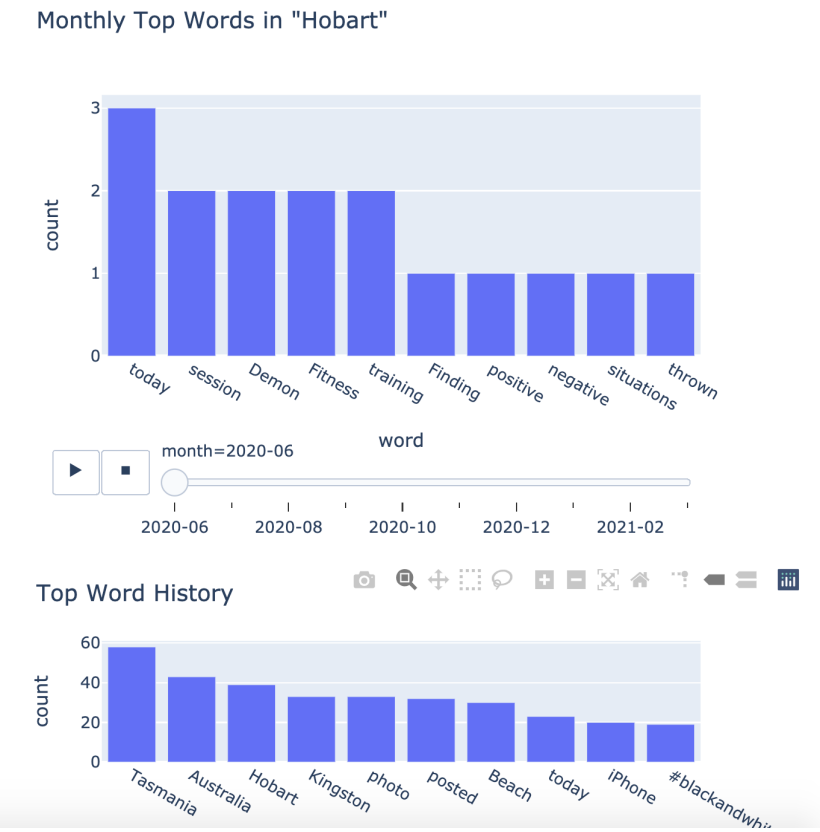


Graph 2: Web application interactive map view

4.4 Scenario Analysis - Top tweet word view

Our web application measures the top tweeted words for each city. Top frequency words are found by checking the vocabularies used in all tweets in a city over a month. These are displayed in the leaderboard above. Each top 10 words leaderboard can demonstrate the popular words in tweets over a period of time, representing the trending topic of residents in the city. Those high frequency words leaderboard is automatically updated every day, as new data is added.

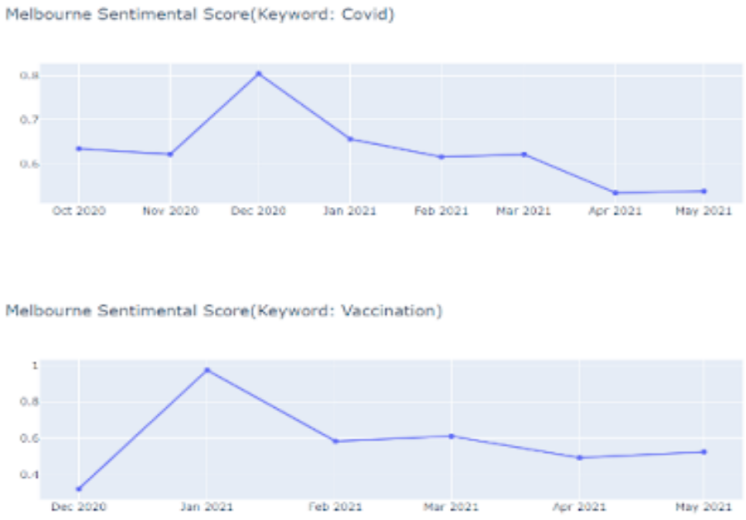
Hopefully, through this top word view, we could understand what’s the most discussed topic in tweets. As shown in the graph below, ‘beaches’ and ‘photo’ are popular topics for people in Hobart of all time and they are enjoying a bit of training and fitness in June 2020.



Graph 3: Web application top tweet words view

4.5 Scenario Analysis - Sentiment Trend

Our web application could measure the tweet sentiment trend of selected city overtime as shown in the graph below. Tweets with keywords: “covid” and “vaccination” are aggregated by month to produce average sentiment score in each group.



Graph 4: Web application sentiment trends view

It is very important to understand how people's sentiment around Covid and vaccination changes over time. Through the time series view of sentiment analysis, we could understand the potential driver of the sentiment shift. For example, we could clearly see in above diagram:

- A significant positive shift in covid sentiment in December, correlated with tweet top word view, we can see that it's because people are tweeting wishes that Covid will be gone during the Christmas and New Year period.
- The sentiment around vaccination has dropped over time since January. That matches the negative news around increasing risk of blood clot with AstraZeneca vaccination and the stop rollout in several European countries[5].

By combining the top words and sentiment trends, we could clearly derive quite useful insights.

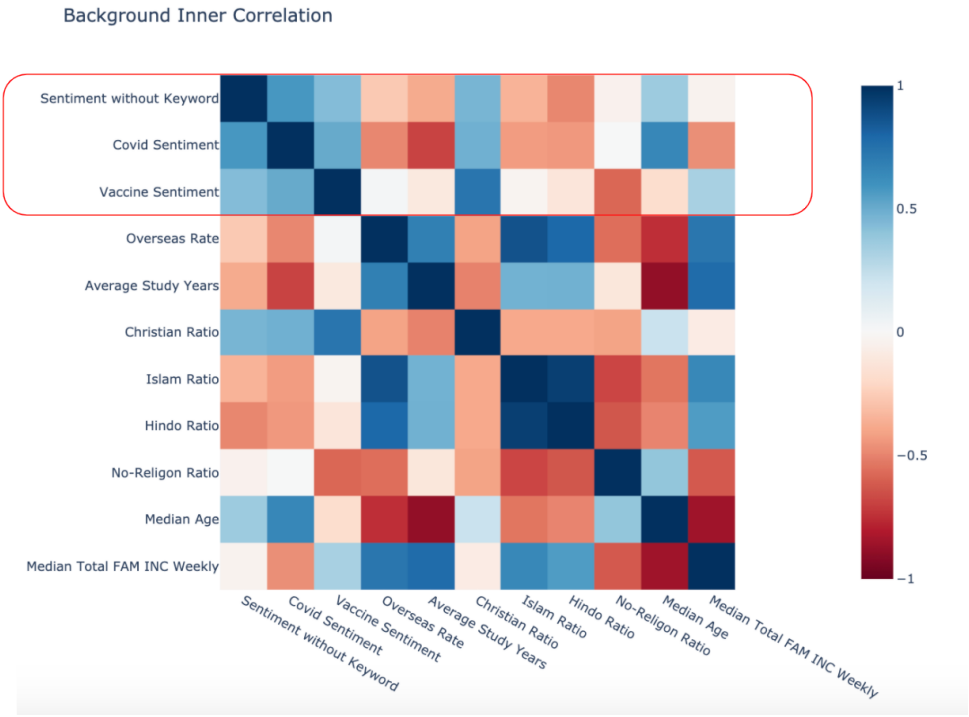
4.6 Scenario Analysis - Correlation view

The correlation view aims to provide some insights to help answer our main research question: which group is mostly affected by Covid and which group is more likely to to accept vaccination. We hope to answer this question by analyzing the correlations between tweet sentiment and collected demographic and socioeconomic statistics from 11 sampled cities.

The correlation view is shown in below graph as a heat map, and there are some interesting observations we can draw from that:

- The ratio of Christian is positively related to tweet sentiments. It seems the consolation of religion may have helped alleviate the stress from covid. The correlation for other religions is not significant due to much smaller population size.
- Age also is positively correlated to Covid sentiment as well. In this pandemic, elderly are exposed to much higher risk compared to younger people, this could potentially explain this correlation.

- Interestingly, the area with higher ratio of no-religion population presents lower acceptance to covid vaccination. Furthermore, it is a useful insight that may indicate the government should spend more effort to promote vaccination to no-religion groups.



Graph 5: Web application correlation view

5. Issues and Challenges

5.1 MRC Stability

During our project, we have experienced sporadic connectivity or deployment issues on MRC. The issues presents no clear pattern and is not related to code/configuration:

- When deploying instances using ansible, we encountered host key verification failure intermittently and the error message indicates that the remote host identification has changed and it might be man in the middle attack. This issue resolves itself after some time or can be solved by creating a new private key and redeploying instances using the new private key. But we are unsure of the root cause of this issue.
- When running the same ansible playbook on all instances, it sometimes won't work on some of the instances, this is very bizarre as every single instance is configured the same and it doesn't make sense to only work on some of the instances. This problem often gets resolved when we run the same ansible playbook again.

Hypothetically, these issues are environment related as MRC servers may be overloaded or undergo patches/changes. We have tried to workaround this challenge by reducing deployment activities and utilizing incremental deployment: deploying only the required components into MRC.

5.2 Docker Image

In this project, we use docker hub to manage our docker images. However, the docker hub only allows us to create one private repository, and only one image can be stored in one repository. This means that if we want to use more than one docker image, we need to make the other repositories public.

5.3 Twitter Harvester

Other than the common challenges such as request limit of twitter API, we have faced/overcome more issues with twitter:

- Initially, we attempted to only take tweets that have exact coordinates so that we could map tweets to geographic areas such as SA2/LGA. The problem is few tweets have geolocation enabled as people are more aware of the privacy risk. This means we have to use bounding boxes to search for tweets and allocate them into different zones. Then we also need to aggregate demographic and socioeconomic data at SA/LGA level into zone level so it could match the tweet data.
- Twitter's streaming API doesn't support filters on both location and keyword. Without awaring of this limitation, we saved a lot of tweets that are not relevant to our analysis topic and may skew the results. So we had to add a custom filter on keywords for our streaming process and cleanup irrelevant tweets in the database.

5.4 CouchDB Cluster Deployment

Although we tried to set up couchDB clusters to prevent losing data from database crashes, the partitional extraction from CouchDB cluster is unsuccessful. It suggests there are issues with our cluster set up. So we didn't manage to finish data duplication in the end.

6. Summary

6.1 Achievements

In this project, we have successfully implemented a cloud-based analytical system that collects tweet data and provides insights on covid and vaccination. There are several key features of this implementation:

- High system availability. As illustrated in section 2, replications are built for the database and web server to ensure the system availability in case of node failure. The cluster setup could also improve overall system performance by sharing the load.
- Automatic/intelligent system deployment. The whole system could be deployed and configured through Ansible scripts. The deployment component also supports incremental setup by skipping the components that are already up running.
- Extensive error handling. Error handling the key to build a robust application, especially in a distributed environment. We have considered and built exception handlers for most of the common errors. For example, the tweet harvester program could handle issues like handle authentication timeout, service request limit reached without throwing out exceptions.
- In-depth analysis. A sentiment analysis based on neural networks is implemented which could provide more accurate results. Also we have made use of geographic dataset in the AURIN system to provide further analysis on the potential cause of the sentiments.

6.2 Future focus

Due to the time constraint, we could not implement all our ideas and features. Here are some suggestions for future work on this system:

- Optimize tweet harvester logic to collect tweets on more granular areas. In current implementation, we created large zones at city level so that we could collect enough data; the downside of this approach is we have to aggregate AURIN data and lose insight at suburb level. Suburbs could contain more distinct socioeconomic features and thus will be more valuable for our analysis. Future work could explore parallel tweet harvester processes or improve efficiency/speed of tweet harvest to collect data at suburb level.
- Generate automatic insight of sentiment shift from top words. We have successfully built a time serial view of covid and vaccination sentiment and also a tweet top words list for each month. It will be quite useful to build another module that automatically performs

sentiment analysis on the top words and hence provide insight and possible driver of the sentiment shift.
- Setting up CouchDB Cluster properly to prevent data losing from database crashes.

Appendix

Youtube video link

https://www.youtube.com/watch?v=qXI08fkBNeY

Github repository

https://github.com/rexding97/COMP90024CCC_ASS2

References

[1] Impact of COVID-19 on people's livelihoods, their health and our food systems: https://www.who.int/news/item/13-10-2020-impact-of-covid-19-on-people's-livelihoods-their-health-and-our-food-systems
[2] Sentiment Analysis of Twitter Data: A Survey of Techniques: https://arxiv.org/pdf/1601.06971.pdf
[3] Twitter Sentimental Analysis Using Neural Network: http://www.ijstr.org/final-print/feb2020/Twitter-Sentimental-Analysis-Using-Neural-Network.pdf
[4] Sentiment140 https://www.tensorflow.org/datasets/catalog/sentiment140
[5] AstraZeneca's COVID-19 vaccine: benefits and risks in context: https://www.ema.europa.eu/en/news/astrazenecas-covid-19-vaccine-benefits-risks-context
[6] "Flask vs Django: What's the Difference Between Flask & Django?" Guru99, www.guru99.com/flask-vs-django.html