COMP90024 Cluster and Cloud Computing Project2

Team 83

Chayanit Jaroonsophonsak - 1025399

Jason Mack - 993060

Li Sean Wong - 1074679

Lu Chen - 1238973

Seth Ng Jun Jie - 1067992

YouTube Video Links:

Web App Demo: https://youtu.be/-s0TVhdYj1M

Ansible Demo: https://youtu.be/CMTDo5zTWAQ

Github Link: https://github.com/seth300301/cluster_computing_a2

Contents

1 Introduction

In this project, we aim to create a cloud solution to analyze tweets from a provided Twitter dataset and toots from Mastodon servers. We leverage the deployment of multiple Virtual Machines (VMs) to pre-process Twitter data as well as harvest toots through the available Mastodon Application Programming Interfaces (APIs). Furthermore, we employ CouchDB as our database and use certain data sets from the Spatial Urban Data Observatory (SUDO) for further exploration.

The focal point of this project lies in harvesting toots within Australia on the University of Melbourne's cloud computing infrastructure, the Melbourne Research Cloud (MRC), and conducting a range of social media data analytic scenarios. These scenarios aim to reveal captivating narratives about life in Australian cities and more importantly, showcase how Mastodon toots can be juxtaposed, compared, or complemented with the data available within the SUDO platform as well as with the tweets on the Twitter dataset. To achieve this, we have explored five intriguing scenarios. Additionally, we discuss the technology stack employed in our project and address the main challenges encountered throughout.

2 System Architecture, Design, and Frameworks

2.1 System Architecture Design Diagram

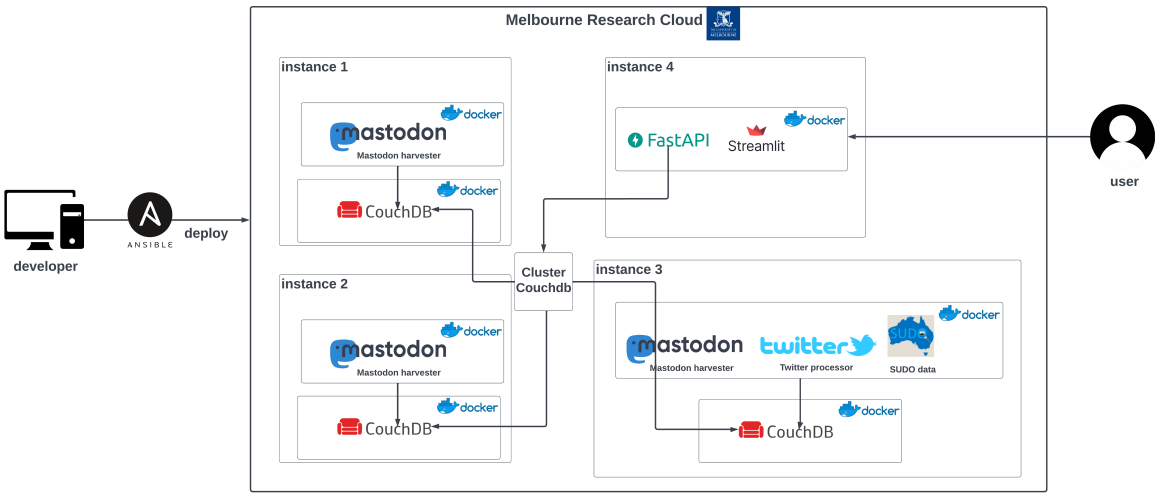

Figure 1: System Architecture Design Diagram

Figure 1 shows the overall architecture of our system. We deploy the system on the MRC with four instances and use Docker to deploy a CouchDB cluster, three Mastodon harvesters, a Twitter processor, and a user interface (UI) application. After the Mastodon harvesters and the Twitter processor finish analyzing data, the processed data will be uploaded to the CouchDB database, regardless of which instance it is processed from. And with FastAPI querying MapReduce views from CouchDB, the relevant data can be displayed on the website.

2.2 Ansible

As a configuration management and application deployment tool, Ansible [1] can standardize and automate the deployment process. With the properly configured playbooks, we can deploy the whole system over a few commands instead of rerunning all the scripts manually. And the result is predictable because Ansible ensures that the same playbook can be run multiple times without causing unintended changes. Also, it's easier to learn and write automation tasks with Ansible, since it uses human-readable YAML syntax to express playbooks.

2.3 Docker

In our system, we deploy the aforementioned CouchDB cluster, Mastodon harvesters, Twitter processor, and UI application using Docker [2] containers. With services, networks, and volumes defined in Docker Compose YAML files and container configurations and dependencies specified in the Dockerfiles, we build their respective containers based on them.

By leveraging Docker containers, applications can be encapsulated with their dependencies into standardized units, making it easier to build and deploy them. One notable advantage is that applications created from the same Dockerfile will return the same output, irrespective of the system on which they run on. This ensures reproducibility and eliminates environment-related issues, making Docker an ideal choice for streamlined development and deployment workflows such as this project.

2.4 Melbourne Research Cloud (MRC)

2.4.1 Description

Built upon an OpenStack framework, the MRC provides Infrastructure-as-a-Service (IaaS) cloud computing, allowing users to perform on-demand computations and utilize storage resources without having to handle complex cloud architecture or maintain the hardware. Using the MRC has no hardware costs and the cloud service is free for us to use. The MRC also has other advantages such as availability, scalability, and high-level security. Moreover, with proper network settings, users can access the MRC from anywhere in the world and adjust resource allocation accordingly to scale up their applications.

However, it's important to note that the MRC can experience occasional instability, which can pose challenges for Ansible deployment. Furthermore, the limited allocated resources may have a negative impact on system performance.

Figure 2: MRC Resource Allocation

2.4.2 Resource Allocation

On the MRC, we have been allocated eight servers (instances) with a total of eight virtual CPUs and no constraints on memory. We are allowed to create up to 500 volumes with a combined storage of 500GB. We have set up four instances each with one CPU.

Our plan is to mount 3 volumes, each with the size of 50GB, to the 3 instances (instances 1, 2, and 3 from Figure 1) that store the CouchDB data. This is to accommodate the continuous retrieval of Mastodon data to our CouchDB database, which might surpass the storage limit of 30GB per instance in the future. Unfortunately, in the end we inadvertently overlooked the task that was initially planned. Even though we fully recognize its significance, our current time constraints make it difficult for us to promptly address this oversight.

2.5 Spatial Urban Data Observatory (SUDO)

SUDO [3] is a web application by the University of Melbourne that contains many datasets ranging from countless of categories and features across Australia. These datasets include the locations of their instances, which can be viewed on SUDO's map feature. Multiple datasets can be selected and integrated to achieve more refined or specified results, giving users a lot of functionality and customisability to its visualisation tool.

2.6 Streamlit

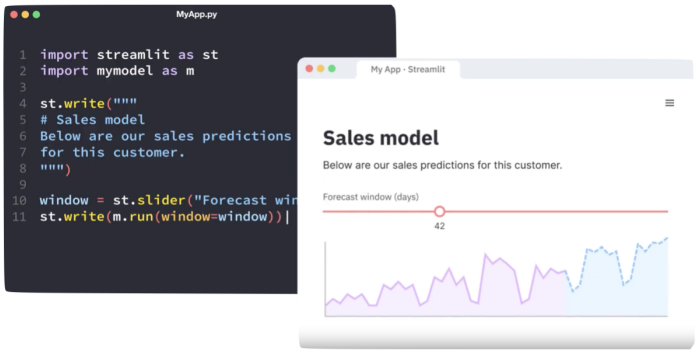

Figure 3: Streamlit Snippet Code Example

In our system, we chose to develop and deploy our UI application with Streamlit [4]. Streamlit is an open source Python based library that is used to build and share interactive web applications and has been growing in popularity due to its intuitive and easy-to-manage syntax. Because of that even inexperienced developers will be able to create and deploy data-driven applications, visualisations, and dashboards with ease, which allows developers to put more focus on building impactful applications without having to worry about the complexities of web development. Furthermore, other external visualisation package, such as Matplotlib and Plotly Dash, can integrate their figures and plots with Streamlit and still offer a high level of customisability for them.

With Streamlit, we have a multipage web application with a simplistic setup. Our dashboard directory consists of a data folder, the main page file (the starting page when the web application is launched), and a pages folder for all the other page files for simplicity. Furthermore, Streamlit's live reloading feature enables us to see the changes in real-time as we modify the code, providing a smooth development experience.

2.7 FastAPI

We chose FastAPI [5] for the creation of RESTful APIs to manage user's query for the Streamlit dashboard visualisations. FastAPI is an open source, lightweight, high-performance Python web framework for building APIs and offers high performance because of its asynchronous programming and Starlette framework. As its name suggests, FastAPI is one of the fastest Python web frameworks, with its speed being up to par with Node.js and Go. Like Streamlit, FastAPI is a simplistic, speedy, and comprehensive tool, making it an excellent choice for developing fast and reliable web APIs.

2.8 CouchDB

Apache CouchDB [6] is a NoSQL document-oriented database that offers a flexible and scalable solution for managing data. It stores data in a schema-less JSON format, allowing for easy and dynamic data modeling. It supports distributed and replicated architecture, providing high availability and fault tolerance. CouchDB also provides a powerful query language called MapReduce, which enables efficient data retrieval and aggregation. Additionally, its built-in conflict resolution mechanism ensures data consistency in distributed environments. Overall, CouchDB is a reliable and versatile database solution for modern applications.

A 3-node CouchDB cluster is used for storing the data and handling MapReduce view requests from the backend server in our system. They are deployed on instances 1, 2, and 3 from Figure 1. In order to set up the CouchDB cluster, we modify the instances' configuration files to change their node names to their IP address accordingly and bind the node addresses to 0.0.0.0 when installing CouchDB, so that they can be accessed by each other node. Once properly configured on each instance, we set up the CouchDB cluster by registering and adding all child nodes (1 and 2 from Figure 1) into the cluster.

3 Data Collection and Delivery

3.1 Twitter Data

We were provided with roughly 63GB worth of Tweets (approximately 52 million Tweets) in a JSON file, with all of them ranging between February 2022 and August 2022.

3.1.1 Entity Pre-Processing

The 52 million tweets were pre-processed to obtain the entities identified in them by first filtering out URLs, usernames, hashtags, newline characters, and emojis (which proved the most cumbersome to abolish) so that we end up with the remaining English string parts. Using a natural language processing (NLP) model called spaCy and its named entity recognition (NER) model, 'en_core_web_sm' [7], the remaining string is read in and a list of recognised entities in it are returned along with the respective entity category label. Only the following three category labels were considered: i) 'GPE': geo-political entity, ii) 'NORP': nationality, ethnicity, religion, or political affiliation, and iii) 'EVENT': events ranging from sports events, concerts, conferences, ceremonies, festivals, natural disasters, accidents, and more. These entities and category label pairs act as entity "UIDs" and the number of times they have been observed in tweets are recorded.

3.1.2 Hashtag Pre-Processing

In a similar way to the pre-processing for entities above, the 52 million tweets had their URLs, usernames, newline characters, and emojis removed, however, their hashtags were retained and recorded how many times they have been tweeted.

3.1.3 Location Pre-Processing

Since we plan to carry out some analyses based on location, we pre-process the data by only extracting tweets that have a valid location. This is done by iterating over all 52 million tweets, checking if they have an associated location, and writing them to a separate JSON file. This resulted in roughly three million tweets extracted (approximately 6% of the tweets provided). These tweets are then classified to one of the six states: New South Wales, Victoria, South Australia, Western Australia, Tasmania, and Queensland.

3.2 Mastodon Data

Mastodon toots are retrieved in real time from three selected Mastodon servers in the Oceania region through the Mastodon APIs. The selected servers are 'aus.social' [8], 'mastodon.au' [9], and 'tictoc.social' [10].

3.2.1 Connecting to Mastodon API

In order to connect to the Mastodon APIs for each server, we need to create accounts and add an 'Application' for each server. This gives us an access token, which will be used to access the server's incoming toots.

To retrieve data using Python, we use the Mastodon.py package. First we connect to the sever by the following function: Mastodon(api_base_url=*server_url*, access_token=*access_token*). Then, we define our own Listener class based on the existing StreamListener class that comes with the Mastodon.py package. Specifically, we define the on_update function such that when a new toot is posted on the server it is pre-processed, filtered, and the output from it is sent to the CouchDB database. The Listener class object will be used as a variable in stream_local to stream the toots. Here, we increase the timeout from the default 300 seconds to 3600 seconds (1 hour). This is because Mastodon doesn't have that much users. To handle potential connection errors while streaming, we use a try block and an except block. 'stream_local' is put in the try block, with the variable reconnect_async sets to "True", which enables automatic re-connection whenever there's a connection error. The except block will catch the error and print out an error statement to let users know that the system is reconnecting to the server.

3.2.2 Mastodon Data Pre-Processing

When a new toot has been posted on the server's local timeline, our system filters and extracts the entities and hashtags from the toot, just like how it is done for the tweets.

At the same time, it also obtains the English string and removes any stop words and irrelevant links from it, where it then searches for keywords based on the scenarios. If a toot contains at least one keyword for a scenario, we store it in the CouchDB database according to the scenario it matches. To ensure accurate matching of short keywords like "need" or "lack", we adopt a strategy that includes an empty space before and after the keywords. This eliminates any partial matching cases, ensuring that only complete and distinct occurrences of the keywords are considered. Consequently, toots containing words such as "needle" or "black" will not be included in our database, maintaining the integrity of the data.

3.3 SUDO Data

The data obtained from SUDO is used to compliment and compare data obtained from the Twitter dataset and Mastodon. The SUDO data obtained is downloaded as a JSON file, as this it is the format required for both the front and back-end of the application.

3.3.1 SUDO Data Pre-Processing

The SUDO data obtained is separated into locations based on Greater Cities. Since this project analyzes location based on states, the pre-processing stage for SUDO data consists of re-classifying the data from Greater Cities to their respective state.

4 System Functionality

In this section we talk about our functionality achievements from our system, our five scenario descriptions, and their justifications, and result analyses.

4.1 Functionalities achieved

The following are the functionalities achieved in our system:

1. Auto harvest Mastodon toots for social media analysis
2. Auto deploy our CouchDB cluster, Mastodon harvesters, Twitter processor, and UI web application

3. Clean and analyze the collected data
4. Visualize the analyzed data in the UI web application with a RESTful API framework querying CouchDB MapReduce views

4.2 States to be Studied

For scenarios regarding locations, we chose six states to be included in our study: New South Wales, Victoria, South Australia, Western Australia, Tasmania and Queensland. This is because majority of tweets were posted in these states and the Northern Territory and Canberra states practically had none compared to them.

4.3 Scenarios

The scenarios we have studied are:

1. Analysis of Trending Entities and Hashtags
2. Twitter Sentiment Analysis Between States
3. Twitter Language Analysis Between States
4. Comparative Analysis of Shortage Complaints
5. Analysis of High Rent Related Tweets

4.3.1 Analysis of Trending Entities and Hashtags

4.3.1.1 Description

Using the pre-processed data from the tweets from the Twitter dataset and the toots harvested from the three chosen Australian Mastodon servers mentioned above we analyze the top 100 entities and hashtags on both platforms and make data visualisations with them using bar charts and word clouds for easier comparisons between them. It is important to note here that while the Twitter dataset contains more than 52 million tweets the amount of harvested Mastodon toots is no where near as much as the entity and hashtag harvesters have only been active for three days.

4.3.1.2 Justification

In today's day and age, almost every business and organization has the need to recognize trends that form within the general public or their own respective audiences for a multitude of different reasons, such as catering to their audiences better to improve customer satisfaction or making better informed decisions based on these opinions of large groups of stakeholders. By staying attuned to these trends, organizations can adapt their strategies, products, and services to effectively meet the demands of their audiences and stay ahead in the competitive market.

4.3.1.3 Results and Analysis

Category	Top 1 Entity	Top 1 Hashtag
Tweets	australia GPE - 729561	auspol - 432608
Toots	australia GPE - 43	auspol - 30

Table 1: Top 1 Entities and Hashtags from Tweets and Toots Obtained

As seen from Table 1, even though there are no where near as many toots compared to tweets the same entity and hashtag are continuing to be trending in Australia even on different platforms as of May 2023 when compared to date range of the Twitter dataset (February - August 2022). While the 'australia GPE' entity may be expected to take first place as an entity, '#auspol' seems less obvious. Auspol refers to Australian politics and upon further reflection it does sound convincing that Twitter and Mastodon users are likely to be people who want to share their opinions on these kind of matters.

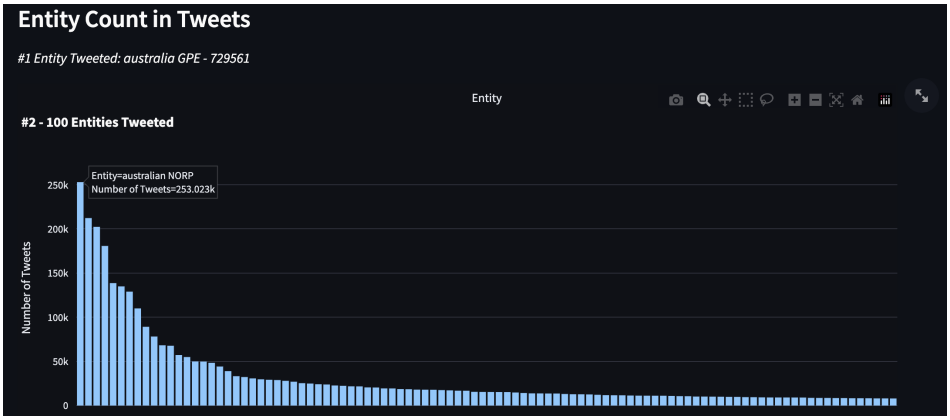

Figure 4: Top 2 - 100 Bar Chart Example

Figure 4 showcases an example of the interactive bar charts on our Streamlit web application. As shown, hovering over the bars shows the entity/hashtag of that instance as well as the number of times it has been tweeted / tooted. The top 1 entity and hashtag have been excluded and reported above them. This is due to the large difference between the first and second placed instances (in this example 720,000+ vs 250,000+) which causes the bar charts to looks very distorted, making it difficult for viewers to understand the data.

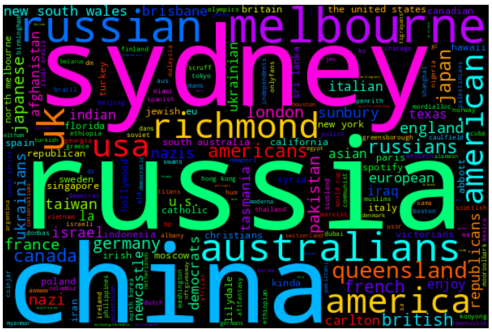

Figure 5: Entity Word Cloud from Twitter Dataset

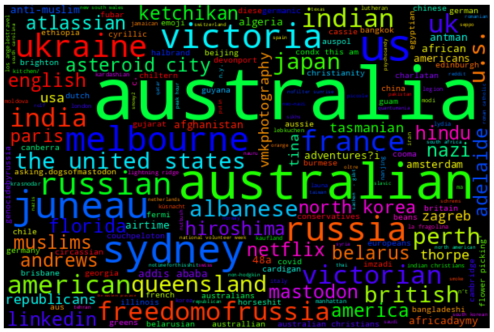

Figure 6: Entity Word Cloud from Harvested Toots

From Figures 5 and 6 we can observe that most entities the two datasets have in common are GPE entities relating to countries, cities, and even Australian suburbs. Note here that the 'australia GPE' entity was purposely removed in the word cloud only for the Twitter dataset due to the same reasons mentioned above but to a much more extreme degree than that of the harvested toots - it would take way too much space and again viewers will find it difficult to see the other entities.

We can also observe here that Sydney was the most trending Australian state during the Twitter dataset's time period, followed by Melbourne then Queensland. However, Sydney's popularity seems to have declined on the Mastodon servers this year with Melbourne being almost as trendy. Perth also seems to be more popular than Queensland amongst tooters.

As expected, the Russia and Ukraine war has made an impact on these datasets with the two countries being large contributors in both word clouds. This is line with the previous statement of how Twitter and Mastodon users are likely the kind of people who want to share their political views, not just on Australian events but worldwide ones too. To support this even further, many political phrases, such as 'republicans', 'democrats', 'nazis', and 'communist', can be observed in the word clouds.

A silent theme that can be identified from observing these figures is religion, with the entities 'catholics' and 'muslims' being the most prominent of the group. If the above observations are not

enough to depict the kind of people Twitter and Mastodon users are, even regarding religion these users wish to express their opinions on them with the discriminatory entity 'anti-muslim' being visible.

Figure 7: Hashtag Word Cloud from Twitter Dataset

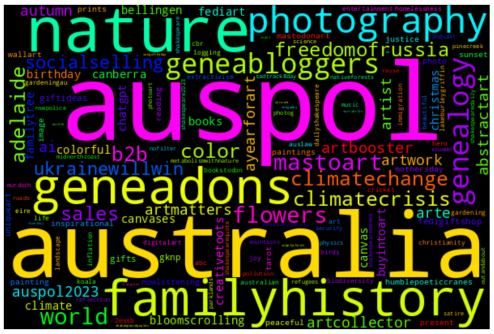

Figure 8: Hashtag Word Cloud from Harvested Toots

Aside from the aforementioned '#auspol' and fairly obvious '#australia' hashtags, this is where Twitter and Mastodon users seem to differ. One trending theme relating to hashtags from the Twitter dataset is cryptocurrency, with hashtags such as '#bitcoin', '#nfts', and '#crypto' being large contributors in the Twitter word cloud. This is likely due to cryptocurrency and NFTs short popularity period that occurred in 2022.

One trending theme relating to the hashtags from the harvested toots is '#genealogy', which is the line of descent traced towards ones ancestors. It has been incredibly for those three days to the point where it has its own specialised word and phrases like '#geneabloggers', #geneadons', and '#family-history'. This is one way of how we can tell that Mastodons is less popular than Twitter and, therefore, is less general and has a more unique community, and in this case it seems that Mastodon users are fond of genealogy.

Both platform users, however, still share their opinions and views on the Russia and Ukraine war through hashtags, such as '#ukrainewillwin' and '#freedomofrussia'. If it makes readers feel any better, there are some hashtags from both platform's word clouds that do give off a sense of normalcy and does not give off political impressions of these users (e.g. '#photography', '#music', and '#nature'.

However, it is important to remember here that our Mastodon harvesters for hashtags have only being active for three days. While some of these hashtags from the harvested toots are in fact clear trends when compared to those of the Twitter dataset, others may just be phases that coincidentally happened during those three days of harvesting.

4.3.2 Twitter Sentiment Analysis Between States

4.3.2.1 Description

We analyze the sentiment of tweets within each state as well as try to find a connection between the sentiment and unemployment rate. Since the Twitter data was obtained during the COVID-19 period (February - August 2022), most people were unable to travel inter-state due to travel restrictions, which would make the tweets from these state clusters more accurate.

4.3.2.2 Justification

Sentiment analysis is extremely valuable, especially for social media monitoring. A platform like Twitter that has tweets, which mostly comprises of relatively short texts, makes it easier to obtain the general public's opinion on certain topics. Analyzing these tweets, in particular the general sentiment, could provide companies and / or organizations with valuable information regarding the public's opinions in which they may act accordingly to to their benefit.

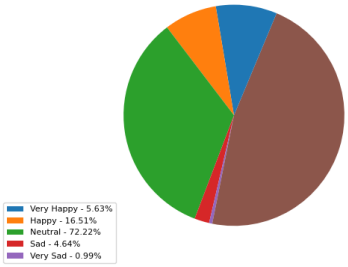

Figure 9: Sentiment Pie Chart for New South Wales

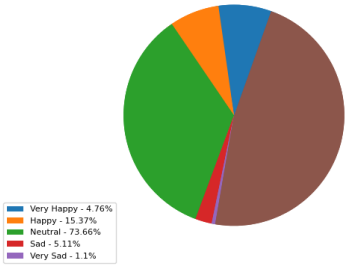

Figure 10: Sentiment Pie Chart for Victoria

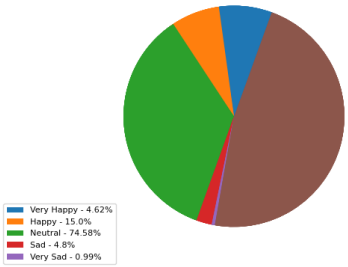

Figure 11: Sentiment Pie Chart for South Australia

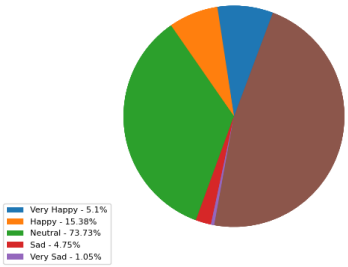

Figure 12: Sentiment Pie Chart for Western Australia

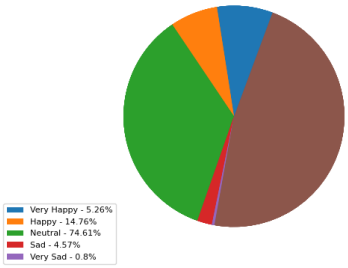

Figure 13: Sentiment Pie Chart for Tasmania

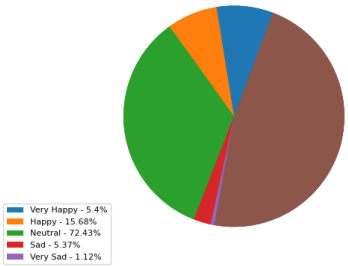

Figure 14: Sentiment Pie Chart for Queensland

4.3.2.3 Results and Analysis

The six pie charts illustrate the sentiment analysis done for the six considered states. Since the aim is to visualize the proportion of tweets (sad, neutral and positive), a pie chart is used as it excels at showing the proportion distribution of different categories within a whole.

The sentiment value, which was included with the tweets from the large Twitter dataset, ranges on a scale of negative one to one; negative one represents very sad and one represents very happy. We categorized the sentiment values into five bins:

From the illustrations, we observe that each of the states have very similar percentages between each other in every category. We observe similar distributions in terms of the categories for each state. After running an ANOVA, we observe no significant difference between the states at the 0.05 signifi-

Category	Bins
Very Happy	x > 0.7
Happy	$0.3 < x \leq 0.7$
Neutral	$-0.3 \leq x \leq 0.3$
Sad	$-0.7 \leq x \leq -0.3$
Very Sad	x < -0.7

Table 2: Sentiment Categories and Bins

cance level. However, we still observe that New South Wales has a higher "Very Happy" percentage than the other states (about 0.6% to 1% higher).

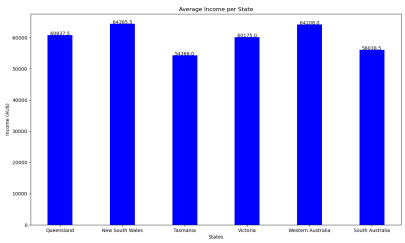

Figure 15: Average Income per State

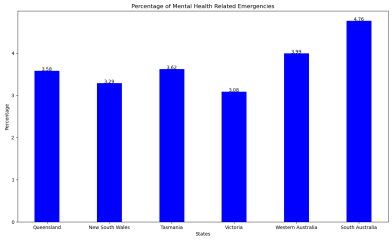

Figure 16: Percentage of Mental Health Emergencies

In regards to income data for each state, we observe that New South Wales has the highest average income ($64385.50) among all the other states. It is plausible that the higher income levels in New South Wales contributes to the generally positive sentiment amongst Twitter users. Furthermore, New South Wales also has the 2nd lowest percentage of mental health related emergencies at 3.29% (just behind Victoria's 3.08%). These factors show that the general quality of life in New South Wales is higher than the other states, which could be the driving force behind the slightly more happier Tweets.

4.3.3 Twitter Language Analysis Between States

4.3.3.1 Description

We analyze the languages used for tweets and compare them to the percentage of foreigners within each state. For this analysis, foreigners are classified as individuals that are not born in Australia.

4.3.3.2 Justification

Australia's popularity amongst international students has transformed it into a culturally diverse nation that attracts individuals from various backgrounds. Moreover, Australia is also a favored destination for migrants seeking new opportunities. This unique setting presents an intriguing research opportunity: exploring the potential influence of diverse cultural backgrounds on the languages used within short-text platforms such as Twitter.

4.3.3.3 Results and Analysis

Firstly, it is worth noting that tweets that were classified with an undefined language (labelled "und") were removed prior to analysis.

From the figures above, we firstly observe that across all states, the amount of English tweets, relative to each state's total amount of tweets, are extremely similar, differing by less than 5%. We also observe that Victoria has the highest percentage of foreigners at 34.24%. Logically, this makes sense due to Victoria having more universities and, therefore, international students compared to the other states. However, this statistic is not reflective on the diversity of languages used on Twitter. There could be many factors for this such as Twitter being blocked in China, a country where a majority of the foreigners are originally from.

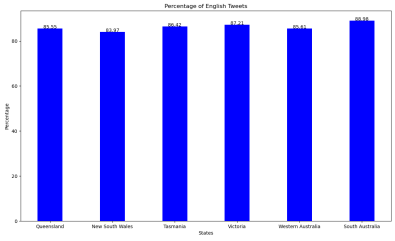

Figure 17: Percentage of English Tweets per State

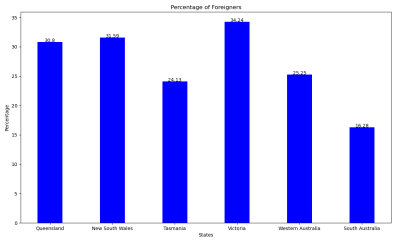

Figure 18: Percentage of Foreigners per State

4.3.4 Comparative Analysis of Shortage Complaints

4.3.4.1 Description

We extract the tweets and toots than contain the keywords relating to 'shortage; such as 'low supply', 'out of stock', or 'lack'. These keywords serve as indicators of individuals expressing their concerns about shortages or the need for additional resources. For the tweets and toots extracted, a bag of words is generated to get the number of different words appear. Then, the same model we used earlier, the "en_core_web_sm" model from the spaCy library, is used to extract only the NOUN words from the bag of words. We assume that these nouns represent the physical objects or the abstract concepts that people are complaining about having a shortage.

After obtaining all the nouns, we promptly examine the top 100-200 words and eliminate those that are irrelevant to our scenario. Specifically, nouns like 'people,' 'day,' or 'way' may not contribute meaningful insights. For Mastodon toots, we can't manually do this since it's a real time analysis so we use the list of non-meaningful words from Twitter to filter the nouns obtained from toots as well. Finally, bar charts and word clouds are generated for these tweets and toots, to visually illustrate what people are complaining about in terms of scarcity. For the Mastodon toots, users can select whether they want to see the overall trends, which are generated from all toots collected so far, or they want to see the current trends, which are generated from the recent 300 toots.

Since the number of tweets provided is massive, in this scenario, we've selected a large sample of three million tweets for our analysis. We believe this sample size is sufficient to accurately represent the overall tweets and provide valuable insights.

4.3.4.2 Justification

Identifying things that are in shortage can be helpful to many sectors. The government can develop targeted policies and interventions to mitigate shortages and ensure the availability of essential goods, services, or resources. Retails or supermarkets can adjust their inventory management strategies and optimize pricing to increase their profits. The general public can understand the current shortage trends in the region, enabling them to make necessary preparations.

4.3.4.3 Results and Analysis

On twitter, the top 10 meaningful words that co-occur with the tweets that mentions shortages include: 'work', 'love', 'support', 'government', 'money', 'covid', 'game', 'health', 'home', and 'auspol', as shown in Figure 19. The same word used in different tweets can carry different meanings depending on the context. In our analysis, we will focus on a few potential interpretations of each word.

Considering the tweets were collected during the COVID-19 pandemic, it is possible that the high occurrences of 'work' and 'home' are related to working from home. When working remotely, people may experience a lack of various elements, such as office equipment, social interaction, or work-life balance. Since people were unable to go out, they might have felt lonely and sad, leading to complaints about needing more 'love' and 'support' from their friends and family. Mentions of 'government' and 'auspol' could suggest people blaming the government and its policies for failing to address the shortages they were facing. The word 'money' likely relates to the financial struggles experienced by many who lost their jobs during the pandemic, and were also affected by the high inflation rate. Regarding

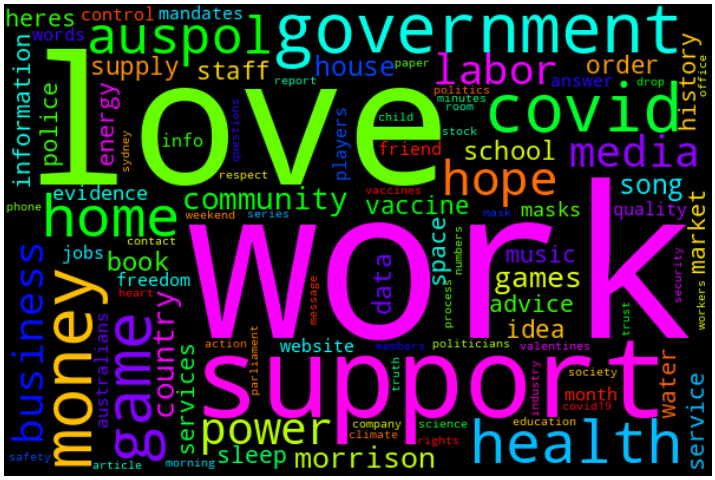

Figure 19: Top words co-occurring with shortage keywords on Twitter

'covid,' it can have various implications, such as vaccine, mask, or rapid antigen test shortages. Due to COVID-19, clinics and hospitals were all busy. That might lead to frustrations about the scarcity of healthcare workers or hospital beds, which explains the frequent appearance of 'health'. In addition, people may have discussed how the shortage situation impacted their mental 'health'. Lastly, the word 'game' could indicate the shortage of PS5 consoles due to the global semiconductor chip shortage during that time, or it could simply imply that people want to buy more games to entertain themselves while staying at home.

In addition to the top 10 words, there are other words such as 'labor' or 'information' that are also straightforward to interpret. Manufacturers could encounter labor shortages as a result of the widespread transmission of COVID-19. Moreover, during that period, people might complain about the insufficient information available about the pandemic.

On Mastodon, the top 10 meaningful words that co-occur with the toots that mention shortages include: 'work', 'placebo', 'game', 'love', 'media', 'home', 'support', 'business', 'climate' and 'apple', as shown in Figure 20. Note that this analysis is based on 628 relevant toots obtained so far. The result will constantly change since we constantly collect new toots to our CouchDB database in real time.

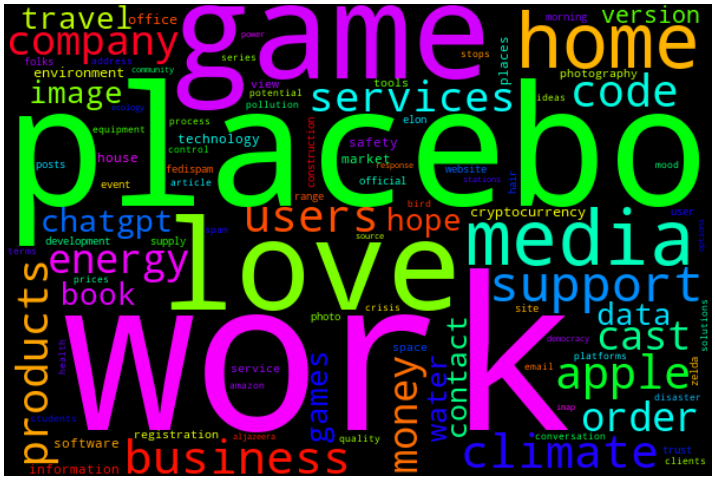

Figure 20: Top words co-occurring with shortage keywords on Mastodon

Tweets and toots share several popular words that co-occur with shortages keywords. These include 'work', 'game', 'love', 'support' and 'home'. 'Home' can be attributed to the current Australia's rent

crisis, caused by the shortage of new houses supply or the lack of social housing. Similarly, 'Apple' is likely coming from the shortage of Apple products. 'Climate' is undoubtedly related to climate change, which can alter rainfall patterns and lead to water shortage. Furthermore, extreme weather can impact agricultural productivity and lead to food scarcity. When it comes to the word 'business', we think its context differs from that on Twitter. Instead of businesses complaining about labour shortages, it is more likely referring to the recovering of Australia's labour market and the reduction in labour shortages. 'placebo' is a really interesting word to appear. After a quick search on Google, we discovered that there has been a shortage in Adderall in the past week. Adderall is a commonly prescribed stimulant for individuals with ADHD, and the shortage is primarily due to manufacturing issues and increased demand. Apart from these, other words can refer to different situations depending on the context. In this case, it is more difficult to make an assumed interpretation compared to when analysing the tweets, because now the impact of COVID-19 on Australia has significantly reduced compared to 2022.

To summarize, analysing shortage trends can provide valuable insights. For example, the word 'placebo' has helped us discover the ongoing shortage in a specific medicine. Nevertheless, obtaining accurate results can be challenging in some cases. One primary challenge is that a single word can have different meanings. The other big challenge is that the motivation of the toot or tweet is unknown. Co-occurring words may either support or contradict each other, or they may not have any direct relationship at all. To improve the interpretation quality, historical events during the relevant period have been taken into account. To further improve the analysis, advanced NLP techniques, such as context analysis and semantic matching, can be integrated into the existing approach.

4.3.5 Analysis of High Rent Related Tweets

4.3.5.1 Description

Our study focuses on identifying tweets discussing the issue of high rent across different states and examining the relationship between the popularity of this topic and weekly rental rates. To achieve this, we employ a filtering process that involves analyzing tweet tokens for the presence of keywords such as 'high rent' and 'expensive studio'.

4.3.5.2 Justification

Understanding the general public's opinions towards high rent can provide valuable insights on the supply and demand in the rental market. Thus, governments and organizations can monitor the overall state of the rental market and design effective strategies to address housing affordability challenges and ensure fair and transparent rental practices.

4.3.5.3 Results and Analysis

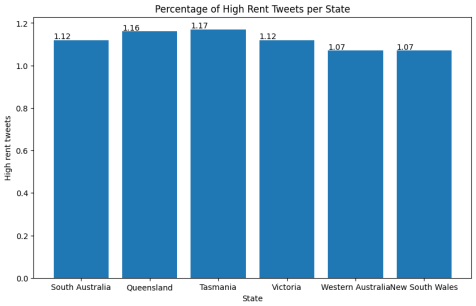

Figure 21: Percentage of High Rent Tweets per State

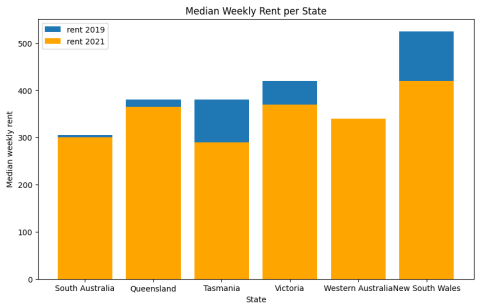

Figure 22: Median Weekly Rent per State

The percentages of high rent related tweets across all considered states are almost the same at around 1.1%. This shows that high rent was not a popular topic in twitter during the COVID-19 period and that there is no correlation between the amount of high rent related tweets and states. However, during 2022 the consumer price index (CPI) rose by 7.8% in Australia. When a high CPI increase occurs it usually indicates high inflation and increased cost of living, there should be more

discussions about high rent in Twitter, which is exactly the opposite of our analyzed results.

To look into the reasons behind this abnormal phenomena, we collect some rent data to learn more about the rental market. With the 2021 median weekly rent dataset from SUDO we make comparisons with the 2019 median weekly rent dataset from Domain's rental report to observe the rental market trends in 2022. As shown in Figure 22, aside from Western Australia the rent of other states in 2021 is lower than that in 2019. Considering how 2021-2022 were during the COVID-19 period overseas migrants were unable to visit Australia due to travel restrictions, so that would explain why the rental demands dropped sharply and the weekly rent plummeted, and hence why very few people talked about high rent on Twitter in 2022. Despite the sky-high inflation, the rent was relatively low in 2022. The rise of weekly rent was delayed compared to other living expenses, and could potentially have been because most people who were renting signed their leases during the COVID-19 period have not moved out or had a rental price increase during that time.

5 Issues and Challenges

5.1 Ansible

Ansible was one of the biggest challenges in this project. Apart from the technical issues we faced, a big issue was the separation between those working on Ansible and CouchDB with those working on the frontend and the Mastodon harvesters. This resulted in code that was not designed with the overall system architecture in mind and many parts had to be modified to be appropriately containerised, run in parallel, and be automatically deployed by Ansible. This could have been solved by clearly outlining the system architecture and program requirements to all team members before starting to code our individual parts.

5.1.1 Different Operating Systems

Members of the team used a variety of operating systems to run Ansible, specifically Windows, MacOS, and Ubuntu. Each had a different installation process and resulted in different configurations. This was especially true on Windows where its subsystem for Linux (WSL) had to be used. An older version of OpenStackSDK had to be used as well and there were several issues with the VPN interfering with WSL. This resulted in many issues when the team shared code and most Ansible development had to be done on-campus.

5.1.2 Connectivity Issues

We had several connectivity issues with Ansible. Some were related to SSH and the strict hostkey checking flag not being disabled correctly. Others were due to issues with Ansible being unable to connect to the MRC. One notable issue was caused by errors when pulling Python Docker images. These errors would occur seemingly randomly on some of the hosts and would result in the whole playbook failing. We are still unsure of the cause of these issues but they have mostly stopped as of now.

5.2 Cluster CouchDB

5.2.1 CouchDB Server Set Up

When we first set up the CouchDB server with Docker, the container kept returning error messages indicating that the server couldn't find the _users and _replicator databases. After researching, we found that in CouchDB the _users and _replicator databases serve specific purposes like user management and data replication management. Thus, we had to manually add _users and _replicator databases after creating the CouchDB server.

5.2.2 CouchDB Cluster Set Up

We faced many issues trying to set up the CouchDB cluster. While, we were able to install CouchDB on each node individually, we struggled to connect the nodes to each other and form a cluster between

them. Performing the cluster setup based on the demonstration from the workshop did not work for us and resulted in a variety of errors.

The team took several steps to resolve the long lasting issue. We decided to use the Apache CouchDB Docker image instead of the IBM CouchDB Docker image since the latter had issues with connecting to Fauxton. We also had to manually go into the Docker container and set the binding address to 0.0.0.0 in the local.ini file as well as the Erlang cookie in the vm.args file. These two changes ultimately resulted in us being able to form a cluster. However, this was an issue for automating the cluster setup process via Ansible as these steps were a highly manual process. The solution was to mount these files as volumes where they could then be dynamically generated during runtime using Jinja2 templates.

We also faced many other smaller issues. At first, when we sent the enable_cluster command, CouchDB returned the message saying that the cluster set_up state is finished. It turned out that adding _users and _replicator databases can change the cluster set_up state to finished. So the databases had to be added after cluster is set up. Another issue was with individual nodes having a node name of "nonode@nohost", which prevented us from creating a cluster.

Many of the issues we faced regarding CouchDB were due to unfamiliarity with Docker and Docker Compose. The extra layer of abstraction provided by Docker Compose made diagnosing and resolving errors much more difficult due to most resources regarding CouchDB not being explicitly tailored towards running it through Docker. However, this provided a lot of valuable knowledge and learning experiences for Docker and networking in general.

5.3 Data Processing

5.3.1 Twitter Data Location Extract

When we tried to extract location data from the includes field of the Twitter dataset instances, we found that there are two formats of includes field. In one format, the location is wrapped in a dictionary named places, while in the other format the location is placed in a list. Therefore, we wrote a try / except function to handle these two situations, which first tries to extract location with the first format and switches to other format if the first one fails.

5.3.2 Shortage Item Identification

Initially, after the bag of words were created, we wanted to use NER techniques to extract words with the tag 'PRODUCT'. However, this approach is overly specific, since it tends to only capture brand names instead of general products. So, we decided to use the part of speech (POS) tagging technique to capture noun words instead.

Still, it is challenging to extract the meaningful nouns from the list of all nouns that can co-occur with the shortage keywords. Having observed the results from the Twitter dataset, we have manually created a long list of noun words to be removed in the final stage. For example, words like 'way' or 'people' which are not really meaningful for our analyses. However, that list is not exhaustive, so there's still a possibility that our word cloud will display some words that do not convey insightful interpretation regarding shortages.

6 User Guide

All shell commands explained below can be found and executed under their respective sub-directories.

6.1 Deploy Whole System

To deploy our system, it is first worth noting that to ensure a smooth run the user might have to edit all the .sh files to include the user's private key. This is an essential step as the system would not be

able to run otherwise.

Firstly, execute the *run-all.sh* file. This can be done by executing the following command: *sudo ./run-all.sh*. This file executes the entire deployment, from creating instances to the final product.

Alternatively, the user can run the *run-first.sh* using *sudo ./run-first.sh* to firstly create the instances. This does not execute the entire deployment. However, this does split the deployment into smaller, more manageable parts. The user will have to manually copy and paste the IPs of all created instances into *inventory.ini* (our team took them straight from our MRC directory).

After the first steps are completed, the user can run *run-second.sh* using *sudo ./run-second.sh*, which executes the remainder of the deployment.

6.2 Add Extra Node to System (Scalability)

In the case where there is a need for a change in resource allocation, our system is able to dynamically scale up. Currently, our system deploys three nodes of CouchDB and the Mastodon harvesters. Executing the instructions below would add a fourth node to the mix.

To create another instance, run *scale-up-all.sh* using *sudo ./scale-up-all.sh*. This runs the entire deployment for creating a new node (from creating the instance to the final deployment).

Alternatively, the user can run the above execution in two parts: creating the instances first, then deploying them. Firstly, create the new instance by executing *sudo ./scale-up-first.sh*. After that is done, the user has to manually copy and paste the IPs of all newly created instances into *inventory.ini* (also, straight from our MRC directory). Then, run *scale-up-second.sh* by executing *sudo ./scale-up-second.sh*. This executes the remainder of the deployment.

6.3 User Interface Guide (Streamlit)

```
ubuntu@web:~$ sudo docker images -a
REPOSITORY   TAG       IMAGE ID       CREATED         SIZE
frontend     latest    98d6d950bee6   16 hours ago   1.93GB
backend      latest    d9fdd224a327   17 hours ago   1.24GB
ubuntu@web:~$ sudo docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS                                                NAMES
9bcaf6be93d3   frontend   "python3 -m streamli…"   16 hours ago   Up 16 hours   0.0.0.0:8501->8501/tcp, :::8501->8501/tcp   frontend
25f9df31e411   backend    "python3 main.py"        16 hours ago   Up 16 hours                                                backend
```

Figure 23: Web Application SSH Server's Docker Images and Running Processes Example

Once the system and its node / extensions have been deployed we can access the web application from the web applcaition instance's SSH server, which we named 'web'. This can simply be done by first logging into the 'web' SSH server and going to the website *http://localhost:8083/*. Since all the images and containers have been built and run on Ansible there are no further actions needed to run the website. However, users can check that the images and containers exist and are active running with the commands *sudo docker images -a* and *sudo docker ps -a* respectively. This is shown in Figure 23.

References

[1] Ansible is Simple IT Automation. Retrieved from https://www.ansible.com/

[2] Docker: Develop faster. Run anywhere. Retrieved from https://www.docker.com/

[3] Spatial Urban Data Observatory, The University of Melbourne. Retrieved from https://sudo.eresearch.unimelb.edu.au/

[4] Streamlit: The fastest way to build custom ML tools. Retrieved from https://streamlit.io/

[5] FastAPI framework, high performance, easy to learn, fast to code, ready for production. Retrieved from https://fastapi.tiangolo.com/lo/

[6] Apache CouchDB: Data Where You Need It. Retrieved from https://couchdb.apache.org/

[7] spaCy, Models and Languages. Retrieved from https://spacy.io/usage/models

[8] Explore - Aus Social. Retrieved from https://aus.social/explore

[9] Explore - Mastodon Australia. Retrieved from https://mastodon.au/explore

[10] Explore - TicToc Social Retrieved from https://tictoc.social/explore