Cluster and Cloud Computing Assignment 2
Australian City (Victoria) Analytics
Team 5 AMOST

Abhineet Gupta	Martin Valentino	Ao Li	Sheng Wu	Teeraroj Chanchokpong
719080	825178	798657	814626	820186

1.Introduction

The amount of data generated from social media in 2017 are enormous and are filled with a variety of content. Approximately 2.8 millions users are tweeting per month just in Australia alone (Cowling, 2017). The variety of content provided by twitter users allows for numerous research opportunities as each tweet potentially contain the coordinates of users. The content of the tweet itself can also be used for the sentiment analysis which provides further insight for users and society to a certain degree. With large amount of data available, it is essential to develop the solution for data analysis with architecture which can support big data. The key technology available to researchers today is the cloud computing architecture. Multiple Virtual Machines (VMs) can be utilized to harvest, store, and analyze twitter data. The cloud services utilized in this paper is the The National eResearch Collaboration Tools and Resources research cloud (NeCTAR) which operates based on the openstack protocol. Data for the analysis topics are also provided as part of Australian Urban Intelligence Network (AURIN)[1].

This paper develops and demonstrates the cloud-based system to harvest, store, and analyze the twitter data with sentiment analysis and demonstrate the usages of twitter data in other research fields. The example of research provided in this paper is to provide insight into the usage of twitter data and its potential in predicting premature mortality by ischemic heart disease and total average number of death in general.

The next two section introduces capabilities and architecture of our system. Section 4 provides user guide, while the fifth section discusses pros & cons of NeCTAR research cloud. The sentiment analysis and extra analysis topic (heart disease) are provided in the sixth and seventh section, respectively. The last section concludes our report. The script required for the project is stored on the GitHub repository located on https://github.com/martindavid/amost-twitter-analytics.

[1] https://aurin.org.au/about/

2. System capabilities

2.1 Overviews

The analysis requires the system to support large scale data storing and retrieval which is why NoSQL database CouchDB is utilized. CouchDB is a document-based database which supports MapReduce as a way to create a view of the data or retrieving a specific section of the data. The database is capable of storing large amount of data and allows for the grouping and retrieving large number of data, allowing further analysis to be done on the retrieved data. All the data including tweets data, AURIN data, and other relevant data are stored in the CouchDB.

The overall system mainly support two functions: the harvesting of the twitter data and the storage of data for the analysis. The analysis utilizes the data from couchDB and is done in Python, the steps is shown in jupyter notebook format. As NeCTAR provides infrastructure as a services, several virtual machines have been deployed into different function. The deployment of virtual machine makes use of Boto python interface with Amazon Web Service which is supported by NeCTAR. The installation and configuration of the virtual machine is done using ansible playbook provided in the repository. One virtual machine is setup as a harvester which stores the data in its local couchDB database. The twitter harvester is setup so that the data harvested is transferred to the CouchDB automatically. AURIN data is retrieved manually from the portal as JSON files. These data which contains both information and geolocation can then be stored inside CouchDB for analysis of extra topics.

The system is scalable due to the nature of the automated deployment using Boto and Ansible. If the key and tokens for Twitter APIs are not limited, data collection on a bigger scale is supported. Moreover, the system also supports the uses of sentiment analysis.

2.2 Twitter harvester

Twitter mainly provides two types of APIs for gathering tweets, REST and Streaming APIs. The REST APIs can both read and writes the Twitter, and within the REST APIs, the search API allows the user to read twitter data. To access the APIs, we create a python program to harvest the tweets using the tweepy library[2]. The harvester makes use of both APIs to maximize the amount of tweets gathered. PostgreSQL is also established on the VM to work with the python harvester. PostgreSQL database

[2] http://docs.tweepy.org/en/v3.5.0/

maintains the topics of interest as well as the keys and tokens required for twitter APIs. The harvester is setup on the same VM as the CouchDB to persistently get tweets from twitter and store these tweets into CouchDB. With the help of twitter id, duplicates are filtered out and add only unique tweets into CouchDB application. The architecture for data harvester is as follows:

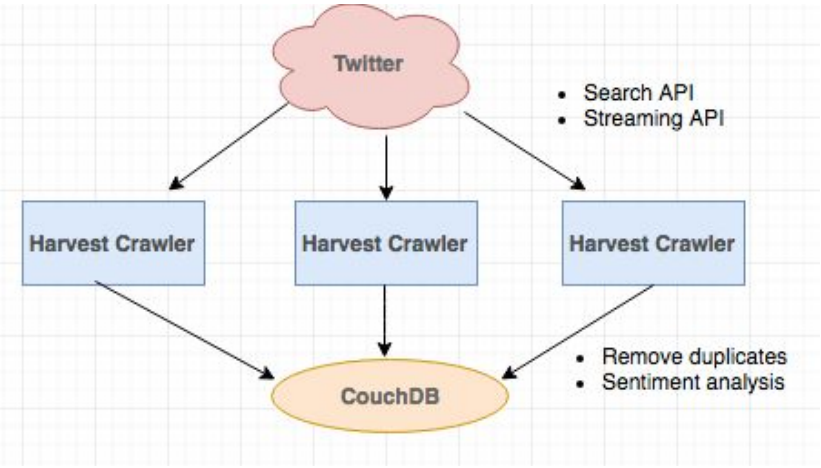

Figure.1 Architecture for Harvest Tweets

The harvester is a python program called cli.py (can be found on GitHub). Multiple harvesters can be running simultaneously while adding tweets to the same CouchDB. There are slight differences between two types of twitter APIs. Current version of harvester (updated throughout the development process) supports automated operation of tagging sentiment analysis along with twitter data prior to storing them on the CouchDB.[3]

2.2.1 SEARCH API

Search API is a part of REST APIs. It is retrospective and will give the tweets from past, up to as far back as the search index goes (usually the last 7 days). There are limitations of how many tweets are allowed to be harvested when using the Search APIs.

The first limitation for Search API is 180 requests per 15 minutes window per-user authentication while each request can ask for maximum 100 tweets, resulting in a total limit of 18,000 tweets per 15 minutes. In order to maximize the tweets Search API, harvesters need to maintain the upper limit allowed by the APIs. The second limit for Search API is that it will not be able to get all the tweets that match search criteria, even if they are present in the desired timeslot because not all of the tweets will be

[3] Some of the older tweets were not tagged with sentiment analysis score

indexable or made available. The third limit for Search API is that this interface is focus on relevance and not completeness[4]. But Streaming API is match for completeness, which is introduced in the following subsection.

2.2.2 STREAMING API

Compared to the search API, Streaming API can give developers a lower-latency access to twitter's global stream of tweet data. Also, Streaming API has a persistent HTTP connection from client side to server side so that it's a real-time delivery and can ensure the completeness of tweets as it can gain more tweets than the search API. For example, in respect with Search API, it works in a request-reply (RR) way. Streaming API is different regarding to the way server responds. Since server would send response to client whenever a tweet is available, so this is a type of socket program where continue stream of responses will come to client from server and the message will appear into the stream immediately with minimal overhead.

Unlike Search API, Streaming API does not limit the amount of tweets harvested per time slot, but only a small fraction of the total volume of tweets would be returned due to its nature of prioritizing relevance.

2.3 CouchDB

The document-based database CouchDB allows for the collection of large database. While the CouchDB only way to view the data is to utilize the MapReduce function, it becomes a right tool for research in social media due to the nature of the research questions. This is because of social media data commonly become more useful when aggregated due to its unreliability individually. While there are considerable improvements on the CouchDB 2.0, we have set up the harvesters quite early on and store them on version 1.6. The difference between 1.6 and 2.0 mainly exists in the efficiency gains. While CouchDB 2.0 is faster with sharding which parallelize the MapReduce, the stable 1.6 version also satisfy the requirements for collecting large amount of tweets for research. The Ubuntu package manager still only support 1.6.1 version for installation using apt-get package manager, otherwise, compilation from source codes is required. Several MapReduce function has been created to support our research topics. These include:

- Sentiment Analysis in Victoria
- Fast Food related tweets in Victoria
- Other general data gathered and stored in CouchDB for general analysis

[4] See https://dev.twitter.com/rest/public/search

The automation of sentiment analysis via harvester implies that various type of analysis related to twitter sentiments can also be automated using pre-defined MapReduce function.

2.4 Supported Scenarios

With the aforementioned capabilities of harvester and CouchDB, general analysis on various topics relevant to the data already contained within the tweets are supported. This include the analysis of selected keywords, locations. This, along with data which can be gathered from AURIN allows for a multitude of analysis with the twitter data if done manually. If the CouchDB MapReduce function is not complex enough for the research question, then additional tools are required. The analysis which requires the creation of polygon to separate the area, for example, may be too complicated for the MapReduce function alone.

Several views have been created to update the section of the data. These can be retrieved by the use of Python script or similar tools for further researches, and the data retrieval can be automated using any of the scripting tools or Ansible.

3. Architecture

The designed architecture utilizes three virtual machines (VMs) (though we are allocated four medium sized virtual machine on NeCTAR). One VM operates as a Web server for the front end and as place the place to do analysis using any analysis tools. It is designed so that the result of the analysis is stored in the same place as the web server for the front end. This virtual machine will have CouchDB installed for storing the result and is deployed manually only once. The other two virtual machines operates in pair. One does the duty of the harvester while another acts as a replica. Harvester runs the harvesting program alongside PostgreSQL to harvest and store the tweets document into its CouchDB. Another VM acts as a backup machine with replicated CouchDB. The following chart visualize the usage of the VMs. When there is a need for scalability, we deploy the harvesters and replicas virtual machines in pairs, if possible. Otherwise, we can choose to deploy only the harvesters as well. Once deployed, the script for analysis can include the IP addresses of new virtual machines to extracts data from the new harvesters as well as setting up the required views on new machines using any forms of scripting tools (bash, ansible).

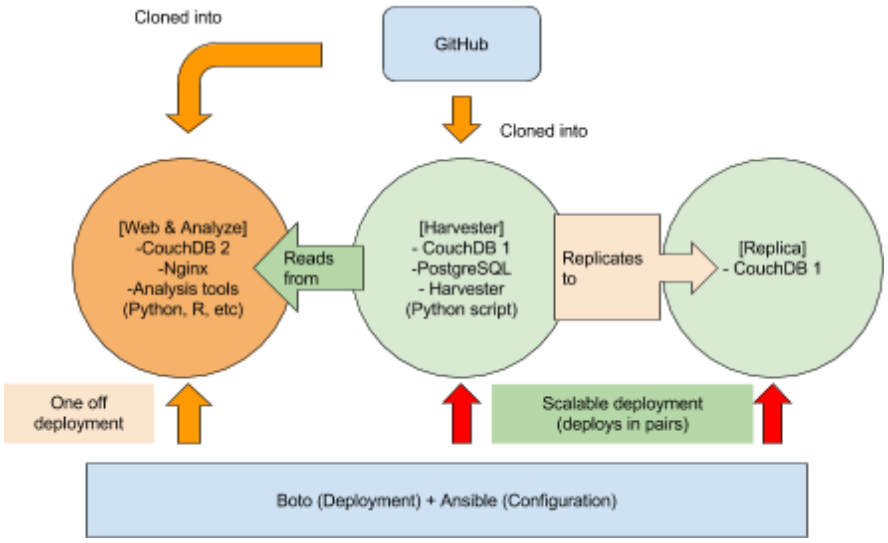

Figure.2 Virtual Machines allocation and responsibilities

Within the CouchDB, the views are created for the analysis and stored as separate document while the harvester stores the main raw data into the document called “tweets”. With the use of CouchDB library made available for Python, analysis can be done for selected topics. The data imported from AURIN can also be stored as separate for the analysis or processed directly by the Python script. In our case, AURIN data as JSON are stored as separate document on the CouchDB, however, the shapefile are processed directly by the Python script to analyze the data. Figure 3 illustrates this overall process.

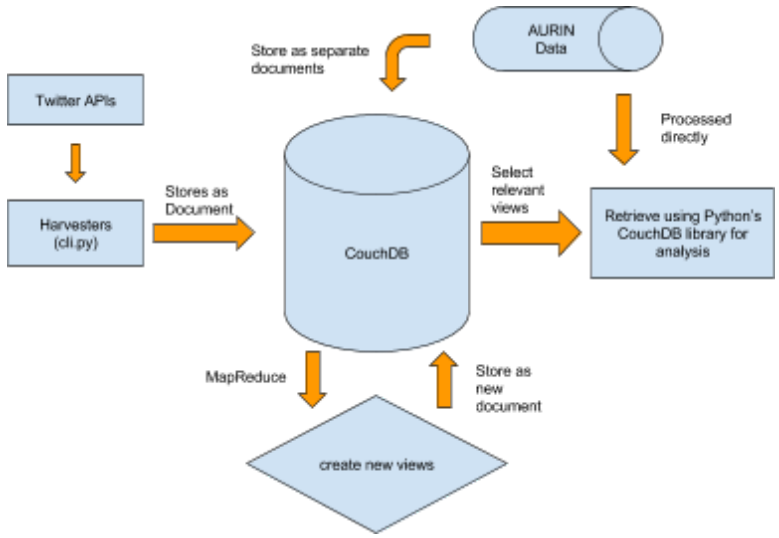

Figure.3 CouchDB architectural design

It is certainly the case that the analysis can be done entirely within the CouchDB if the research question is simple enough and all of the data are contained within single CouchDB. For example, the sentiment analysis separated by time of the day. For more complicated analysis in which the uses of library are required, it is certainly more appropriate to do the analysis using other tools outside of CouchDB. The code snippets for the MapReduce function is provided in the sentiment analysis section.

The architecture is scalable if the research requires large amount of research data, the VMs for harvesters can be replicated as much as the resources from NeCTAR allows the researchers to. External storage is also attached on the Harvester as well as the replicated analysis VM to expand the capabilities of the system. Multiple harvesters can be configured to store the tweets harvested within its own local virtual machine. Then multiple VMs can use similar MapReduce function to create values of interest. These values can then be aggregated using analytic tool such as Python (or Java, Scala, R, etc) as long as the tools have library to interface with the CouchDB.

While the current architecture and script for auto deployment (described in next section) is designed to fulfill the mentioned architecture. This architecture is developed through the experiences with the system. The actual architecture used during the project was to use single virtual machine doing most of the tasks, with offline backup.

4. User guide

This section guides through the deployment, and the usage of the system presented in this paper. These include the system deployments, end user invocation for the retrieval of the views, and the access to the virtual machines.

4.1 System deployment

As the NeCTAR supports Boto, a Python-based API, for deployment, the deployment of VMs and the creation and attachment of the storage is done using Boto script. The installation and configuration is done using Ansible playbook. The script is provided on the GitHub on the folder called automated deployment. The requirement for the auto deploy is Python 3.5 + and boto library installed prior to execution. This section is written assuming that

To start the deployment, run the shell script provided on GitHub. The script will call both Boto deployment and ansible file to configure the VMs.

- `chmod +x ./deploy.sh`
- `./deploy.sh`

To configure the deployment, configure the access key id and secret access key in the AMOSTSettings.py in the autodeploy/boto/, then after deployment, configure the

IP address of the virtual machines in the hosts file in the /autodeploy/ansible. This process will deploy the harvester to harvest tweets relevant to the groups defined within the PostgreSQL. To change the keywords for twitter harvesting, configure the keywords in the follow file:
twitter-harvester/db_structure/keyword.sql (manually configure the keyword in .sql files).
Note that the deployment deploys Ubuntu and install the stable CouchDB 1.6.1 version using the Ubuntu packages manager. If the repositories are updated[5], the version 2.0.0 will be installed. The harvester will be compatible with CouchDB 2.0 as well.

4.2 End user invocation

4.2.1 Manual deployment of harvester

To start the harvester for twitter data manually, and start doing sentiment analysis and storing it to couchDB, download the store the private key (amost-1-.pem) and store it on local machine, then:

- Step 1: In terminal, type
  `ssh -i path/to/amost-1-private.pem ubuntu@115.146.92.156`
  to access the VM using amost-1 as the passphrase.
- Step 2: Go to the directory
  `/home/ubuntu/amost-twitter-analytics/twitter-harvester`
  to find the main script cli.py.
- Step 3: Type
  `python cli.py <type> <group_name> &`
  to keep the script running in the background.

The parameters here are type and group_name. There are two types to crawl data: search and stream. And six groups exist in PostgreSQL database, with each having its own keyword list.
For example,

```
python cli.py search GROUP1 &
```

4.2.2 Accessing Futon

To access futon, in the terminal, type:

```
ssh -i amost-1-private.pem -L 5984:localhost:5984 ubuntu@115.146.92.156
```

in terminal to access the VM and create a tunnel between local computer and the virtual machine on port 5984 (change the port to something else if the installation configuration is different). Use `127.0.0.1:5984/_utils` in the browser to access Futon and create new views using MapReduce.

[5] Check the ppa here https://launchpad.net/~couchdb/+archive/ubuntu/stable

5. NeCTAR Research Cloud

NeCTAR research cloud operates on the standard openstack platform which are free and open source platform. The transparency is one of the benefits. The support from the free source community is another. This section discusses the pros and cons of using the NeCTAR. The benefits listed below generally corresponds to the benefits of using cloud in general.

5.1 Benefits of NeCTAR Cloud

1. NeCTAR Research Cloud provides a platform for researchers across Australia to easily operate cloud computing.
2. NeCTAR has large storage, instant availability and can easily scale.
3. Availability of the standard images to boot from streamlines the deployment process
4. Availability zone is easily defined.
5. Several users can access one specific VM, or work on the same data at the same time.
6. Users can set up many VMs to build their own infrastructure as long as they have resources.

5.2 Issues with NeCTAR Cloud

1. The potential problem of data losses if the VM goes down persists. Backup is always an important issue.
2. While it is not an issue with NeCTAR per se, redundant images can potentially create problems if the images with same names are configured differently when deployed. Since configuration is abstracted out, it becomes difficult to figure out which components are not compatible with required software.
3. The deployment using Boto, especially during the testing periods, can be a little difficult since the termination of instances do not immediately update the resource constraint (amount of CPUs allowed). We have to wait for a while before testing the deployment scripts again.
4. Security group creation and attachment does not always work immediately. This was particularly a problem for accessing CouchDB remotely, especially on the University Student's WIFI network where the network security policy may have conflicts with this intention.

6. Sentiment Analysis

Sentiment analysis has been applied to in the section of Victoria to see whether people are happier in the morning, or in the afternoon, or during night time. When harvesting the data, the harvester utilizes a library called vaderSentiment to do the sentiment analysis and store the scores as a new field in the tweet to CouchDB (Hutto & Gilbert, 2014). Then built-in Map function provided by couchDB is used to filter tweets which don't have geographical information or are not in Melbourne. Compound score is then used to measure the actual sentiment of the tweet.

Several scenarios of the analysis using sentiment score have been done to demonstrate the capabilities of the system. These are described in the section 6.2. Various results have been visualized and shown on the sites running on top of one of our virtual machines. The site is located at http://115.146.92.156:3001/[6].

6.1 Vader Sentiment Analysis

To directly quote the documentation, "VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media" (Hutto, & Gilbert, 2014). It deals with many complex situations when mining twitter content. Instead of simply searching terms, it solve situations like typical negations, emoticons, slang words etc.

After applying vaderSentiment, the analysis yields a result in the format of, for example,

{'neg': 0.0, 'neu': 0.254, 'pos': 0.746, 'compound': 0.8316}

The pos, neu, neg scores are ratios for proportions of text that fall in each category, which means they should add up to 1 or close to 1 with float operation. The compound is computed using some rules and normalized to be between -1 (most extreme negative) and +1 (most extreme positive).

The compound score is used to classify whether a tweet is positive, neutral, or negative. Typical threshold values (Hutto & Gilbert, 2014) are:

1. Positive sentiment: compound score >= 0.5
2. Neutral sentiment: -0.5 < compound score < 0.5
3. Negative sentiment: compound score <= -0.5

[6] The site should be running for sometimes in 2017.

The script stores all scores, neg, neu, pos, compound to a new field in the tweet document then stores them. Note that there are many documents without sentiment scores attached to them as the initial version of harvesters does not apply the sentiment analysis before storing them.

6.2 Analysis

The analysis makes use of the MapReduce provided by CouchDB. A map function is stateless, which means it takes in and only requires a value or a parameter, and outputs <key, value> pairs. It allows users to run the function against values in parallel. In this case, the parameter is a tweet for one call.

Map function is used to create a view which label timeslot in which the tweet belongs to. Then reduce function creates an average compound sentiment score for each time slot. Map function for this task is as follows:

```
function (doc) {
    var parsedDate = new Date(doc.created_at);
    var time = parsedDate.getUTCHours();
    var compoundScore = doc.sentiment.compound;
    if (time < 14 && time >= 6)
        emit("morning", parseFloat(compoundScore));
    else if (time >= 14 && time < 22)
        emit("afternoon-evening", parseFloat(compoundScore));
    else
        emit("evening-night", parseFloat(compoundScore));
}
```

Reduce function is as follows:

```
function(keys, values) {
  return sum(values)/values.length;
}
```

For the time slots, 6:00 - 13:59 are classified as “morning”, 14:00 -21:59 as “afternoon-evening”, and 22:00 - 5:59 as “evening-night” and the raw result is as follows:

Time slot	Compound Score
"afternoon-evening"	0.11229687347826038
"evening-night"	0.1941539675561806
"morning"	0.13990471096915574

Figure. 4 The compound score of sentiment analysis by time slot using all tweets gathered

The view took 35.01.517 minutes to process, utilizing 1,170,809 tweets relevant to the topics harvested (see section 7 for keywords specified), and it indicates that the time in which people are happiest during the day, on average, is the evening until night. Generally speaking, most tweets have neutral sentiment.

Other analyses have also been done is to classify the sentiment by day and by time slots. The results of analysis, including the total distribution of the sentiment score, is illustrated below:

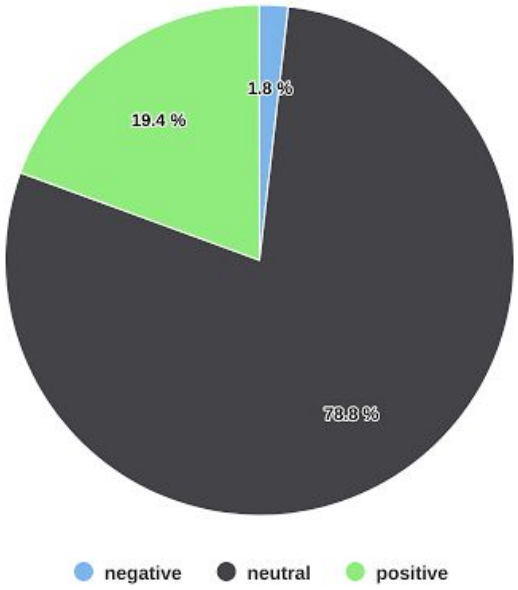

Figure.5 Distribution of Sentiment

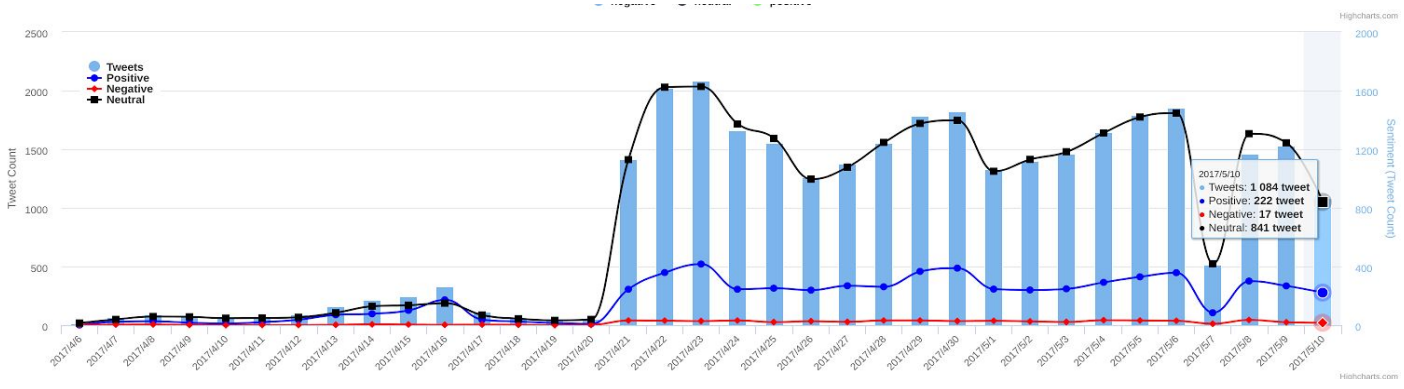

Figure.6 Sentiment Analysis of the tweets by days

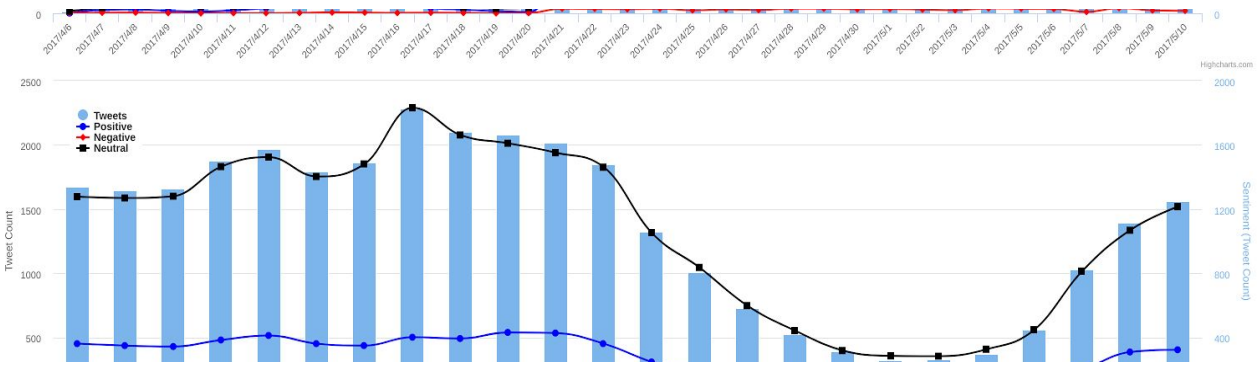

Figure.7 Sentiment Analysis of the tweets throughout the day

The analyses shows that, for the topics harvested in this paper, there are very few tweets during late afternoon until around 22:00 in which the numbers of tweet started to grow.

Additional analyses have also been done with gender computer[7] to estimate the gender of the tweeters. These results are also shown on the site mentioned above.

6.3 Limitation of the analysis

The linguistic characteristic of tweets (or social media in general) are not well-structured. Users tend to use a lot of emojis / signs and, due to the limitation of words in Twitter, unofficial abbreviations. Moreover, twitter is also mainly used as a platform for advertisement which include a large number of links and makes the analysis less accurate. One more thing to notice is that a keyword-only method will create noisy content. For example, when searching “music”, one tweet might be “Justin Bieber’s music is not good.”, which gives no information about the music itself.

VaderSentiment also doesn’t support sarcasm. Vader does classify emojis up to a certain extent, but not all, since vader is still improving and some of the images based emojis, memes are not always supported. A large number of tweets become a neutral sentiment when the tweet itself has a few words to describe the image inside the tweets. Considering the amount of memes are on twitter, it is perhaps not surprising that the neutral sentiment are numerous.

[7] See https://github.com/tue-mdse/genderComputer

7. Extra Scenarios Analysis

Initially, the topics specified at the beginning of the harvesting process was to focus on the keywords related, in general, to food, entertainments, and so on. The keywords included are as follows:

- Food, Music, Health, Entertainment
- #Easter, Easter, happy easter
- Beer, Tv series
- Rock music, Pop music, Jazz music, Music Bar, Music cafe
- Fast food, Hamburger, Burger king, KFC, Fried Chicken.

The twitter harvesting system also gradually receive additional capabilities throughout the time in which harvester were running. With available data, the topics were decided to be correlation between premature mortality and the tweets involving fast foods. To expand the analysis, correlation between premature mortality and the amount of positive and negative sentiment tweets are also explored. The reason for this topic is due to the fact that using social media data to correlate with actual health statistics could potentially lead to interesting research topics in psychological field. Instead of small experiments in psychology, it is possible that big social media data can be the way to replicate and test established theory in psychology as well as food and nutrition which is why the tweets gathered for this paper are revolving around foods and entertainment.

7.1 Research question, methodology, & hypothesis

The main research question is to explore whether there will be a correlation between the average premature mortality per 100,000 person and the amount of tweets related to fast-food, tweets which are considered to be negative, and tweets which are considered to be positive. These questions utilizes the capabilities of the system. We defined the local government area (2011) (LGA) to be the level of analysis. The scope of analysis also focuses only on the Victoria state. This means that there will be only 79 observations, or 80 if unincorporated area is included. For the correlation, the implications can be somewhat limited. Thus, a simple linear regression is also utilized to see the relationship.

Hypothesis

Prior to the analysis, the hypotheses are as follows:

- Negative correlation between average number of premature mortality and positive sentiment. This implies that positive sentimentality in the area the correlates with less premature death. This potentially leads to various scenarios to explore. For example, is positive sentiment reflective of less stressful lifestyle?
- Positive correlation between average number of premature mortality and negative sentiment and food-related sentiment. This implies that the negative sentimentality and more fast food related tweets in the area correlates more premature death in the area. This potentially leads to various scenarios to explore. For examples, does fast food mentions imply consumption or as a part of the diet of the population in the area?, does negative sentiment signals more stress in the area?
- To further extends these questions, replacing premature mortality with premature ischemic heart disease should also results in the same correlation.

Data & Analysis

For the twitter data, the harvester was running since 10 April 2017 until 10 May 2017 while the topics for the extra analysis were being decided[8]. The tweets used for this analysis includes the data from 10 April 2017 to roughly 6 May 2017 and only include data from Victoria area. The coordinates of the tweets and the compound sentiment score were extracted by creating a view using MapReduce function.

As for the data about the premature mortality, the data are provided by the Torrens University Australia, Public Health Information Development Unit (PHIDU) and accessed through AURIN portal[9]. The records calculates the average premature mortality per year per 100,000 person in each of the area in Victoria from the data from 2008 - 2012. The premature mortality used is the total and death from ischemic heart disease (or coronary heart disease). The data can be downloaded as shapefile or JSON or CSV. The JSON file of the data was also uploaded into the CouchDB. The CSV and shapefiles are read into the same Python script to do the analysis on the topic.

These two data stored in CouchDB or loaded into the python script directly allow the analysis to be done on the topic. The script is provided and contains more comments and details as a Jupyter notebook to easily illustrate the process and results and are contained within /analysis folder.

[8] In retrospect, we should have either included more topics or include all other topics considering the size of storage we have available.

[9] Newer version is now available, but it is not updated on the AURIN. The accessed data is "**LGA11 Premature Mortality**"

Potential problems in the model

While it is true that “all models are wrong, some are more useful than others”. The correlation using tweets are filled with flaws and requires assumptions more than other models. The following list display the potential pitfalls of analyses.

1. The timeframe is a mismatched: while the premature mortality data was collected and calculated using the data from 2008 to 2012, the tweets are collected in 2017.
2. Twitter data tend to be more populated in the urban area which affects the correlation since absolute number of tweets, regardless of the sentiment scores, is higher in the urbanized area.
3. The model has no control variables, just a simple multiple regressions and simple analysis.

However, the main focus of the paper is to utilize the system to perform a task and not perfecting a statistical model. Another major assumption put into the analysis is the fact that tweet with compound score = 0 are classified as negative sentiment. This is arbitrary and is not a good form of research endeavour, but this is done to ensure that there are enough observations for the model to be meaningful.[10] This assumes that some of the tweets which are not classified and remain neutral are actually negative. Ideally, if one takes the whole year worth of tweets, then compare them with average premature mortality data for the year, then the result will be more reliable.

7.2 Results

For correlation, we see negative correlation between premature mortality and the all types of the tweets, be it positive, negative, or food related correlation. The results of correlation analyses are illustrated below as a simple scatterplots with fitted regression lines for illustration:

[10] Official recommendation put in compound score of >= 0.5 as positive and <=-0.5 as negative. The rest is neutral, we chose to classify the truly neutral (0) to be negative.

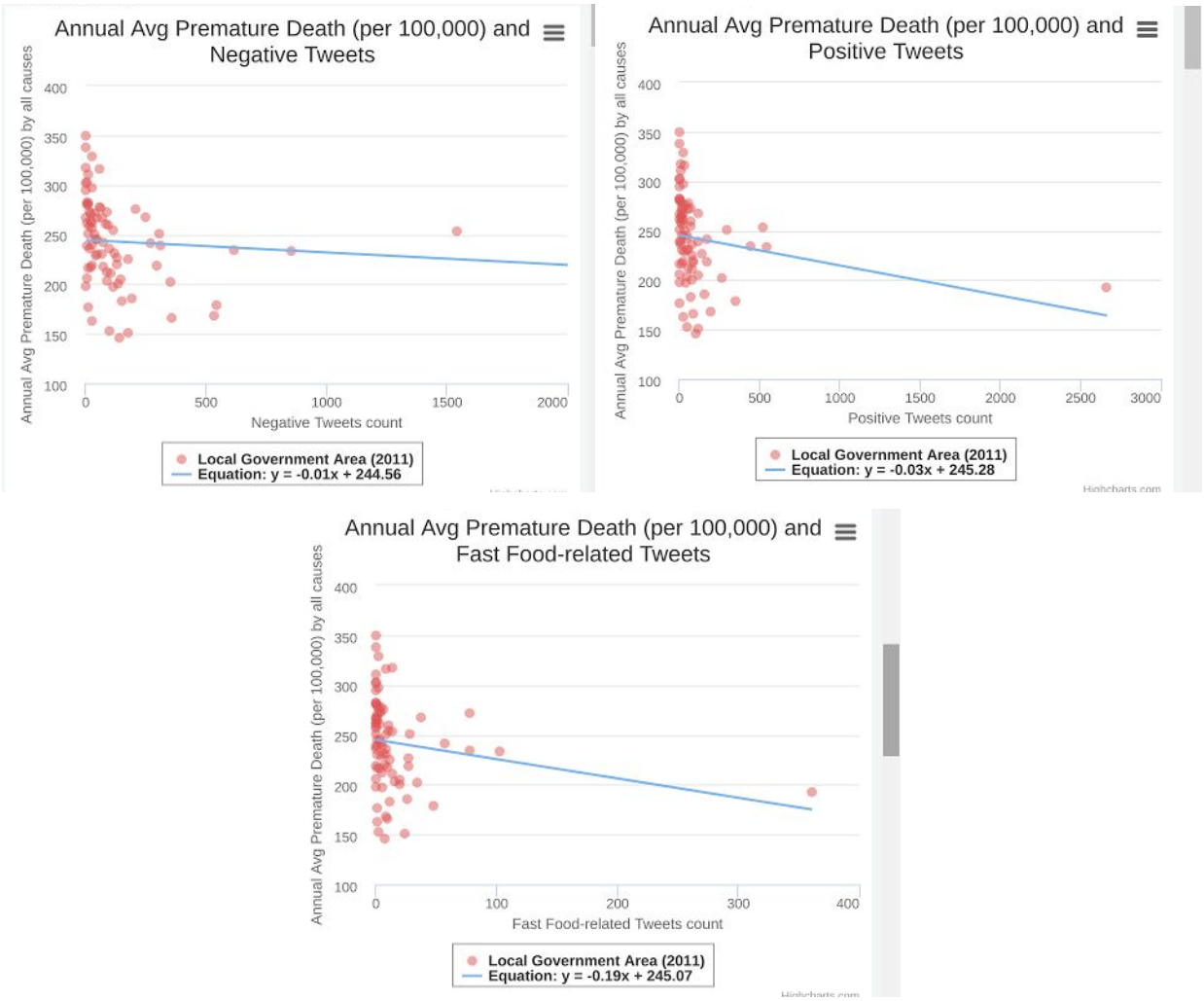

Figure.8 Annual Average Premature Mortality (Death) per 100,000 person in each local government area (2011) and its correlation with Positive, Negative, Fast food related count.

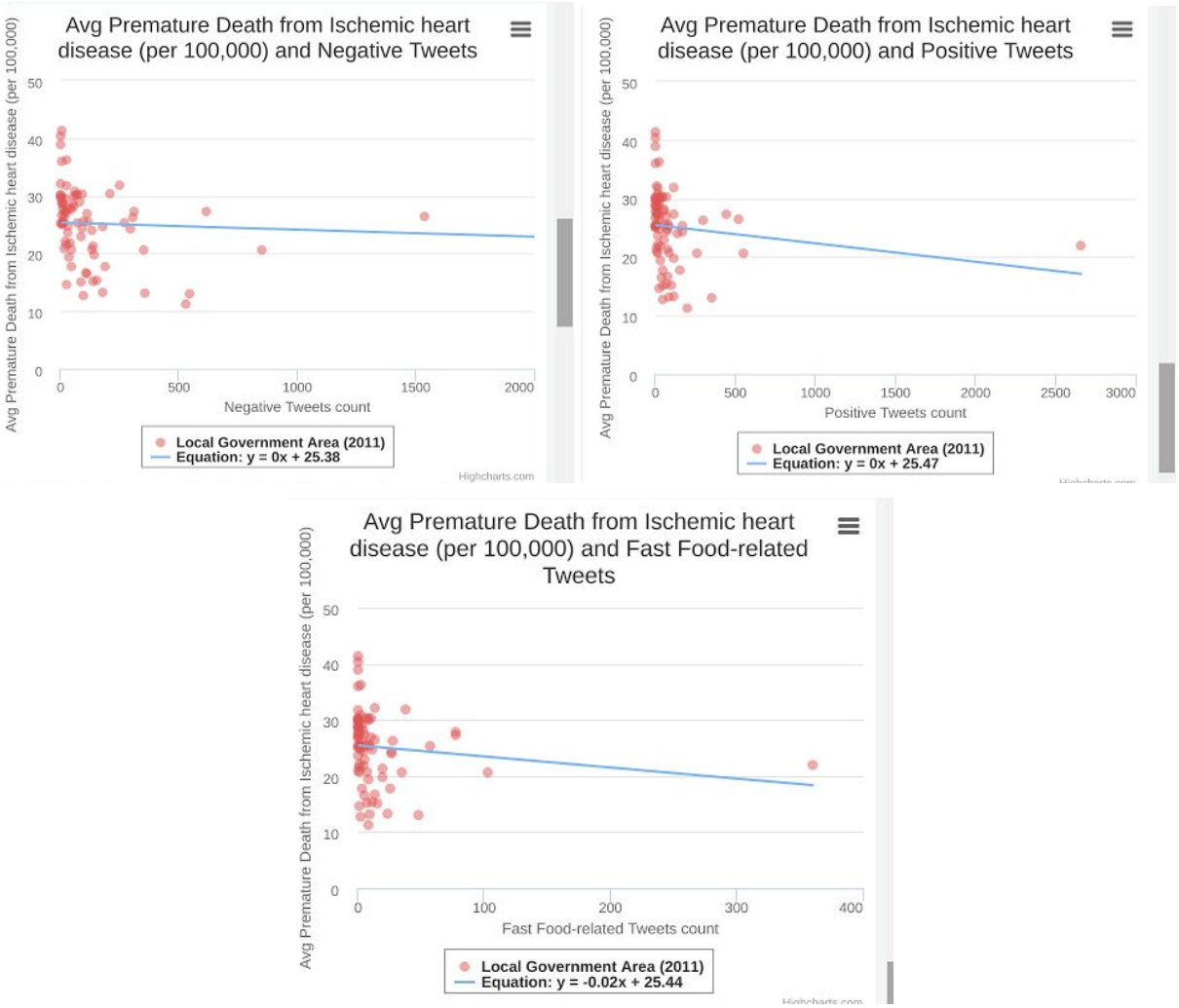

Figure.9 Annual Average Premature Mortality (Death) by Ischemic Heart Disease per 100,000 person in each local government area (2011) and its correlation with Positive, Negative, Fast food related count.

The negative correlation between premature mortality and the negative tweets, and fast food related tweets count, seem counter-intuitive. The problem could be with the fact that more urbanized area tend to have more tweets, regardless of the sentiment. This along with the fact that urbanized area could potentially have better health supports which reduce premature mortality. Note that these correlations are generally very weak, but the increases amount of tweets (running harvesters for longer period) could potentially lead to a more significant result.

To better control relevant information in the analysis, the negative, positive, fast food related tweet counts are used as independent variables in a simple ordinary least square (OLS) regression with either average premature mortality by all causes per year

per 100,000 person or average premature mortality by ischemic heart disease per year per 100,000 person as dependent variable. Raw ordinary least square regressions results are included in the appendix.

Here is the regression result summary in brief with [t-value]:

$$\begin{aligned}\text{Total Death} &= 247.1435 \quad [45.352] \\ &+ 0.0790 * \text{NegativeSentiment} \quad [1.396] \\ &- 0.2690 * \text{PositiveSentiment} \quad [-1.645] \\ &+ 0.4891 * \text{FastFoodTweets} \quad [1.070]\end{aligned}$$

Here is another regression but dependent variable is configured to be premature mortality by coronary heart disease instead.

$$\begin{aligned}\text{Death by Heart Disease} &= 25.4298 \quad [29.909] \\ &+ 0.0128 * \text{NegativeSentiment} \quad [1.440] \\ &- 0.0414 * \text{PositiveSentiment} \quad [-1.614] \\ &+ 0.0782 * \text{FastFoodTweets} \quad [1.090]\end{aligned}$$

Interpretation: with linear regressions, simplest interpretation is to look at coefficients so that it can be interpreted as 'all things being equal, for every one increase in a positive tweet count, you can expect on average -0.2654 decreases, on average, in average annual rate of premature mortality per 100,000 people.' Also, 'all things being equal, for every one increase in a positive tweet count, you can expect on average 0.0408 decrease, on average, in average annual rate of premature mortality by heart disease (ischemic) per 100,000 people.' Note that the statistical significance level is around 89% at best for the positive sentiment, implying that this could be potential for further researches[11].

Note that the model presented here is not properly controlled nor does it have an excellent hypotheses supporting them. Hence, it should not have a high predictive power. The results presented here should be somewhat indicative of the correlation and not taken literally as a causation.

[11] This area of research is probably already well-established, but the measurement of positivity using social media could be novel.

8. Conclusion & Further Improvements

This paper has developed a cloud-based solution to the research using big social network data and demonstrated its capabilities with a simple correlation / linear regression analysis between premature mortality and the amount of tweets count related to fast food, positive, and negative tweets.

The system is designed with scalability in mind with the design to automate the deployment and scale up the twitter harvesting process. At the center of the architecture, is the document-based database CouchDB. The database stores each tweet in a document format which can then be filtered and retrieved through CouchDB native MapReduce function in order to support various research efforts. The system also have built-in sentiment analysis support prior to storing the tweet in the CouchDB which allows for a variety of sentiment related researches to be done. Further improvements include the upgrade of the CouchDB to 2.0 to support sharding and improving the efficiency of the system.

The auto deployment script can easily be modified to scale up the operations and collect more data if resources are available and should be compatible with cloud provider which supports Boto deployment such as AWS. The keywords can also be modified to start harvesting on a different topic. This allows the system to be reasonably flexible and should be able to support a multitudes of research areas which could benefits from the big data from twitter.

9.) Team member & Roles

Abhineet Gupta

- Main system administration
- Ansible playbooks
- Boto deployments
- Architecture design
- Documentation & videos presentation
- Project management

Ao Li

- Harvester developer
- CouchDB views and analyses
- Sentiment analysis code
- Documentation

Martin Valentino

- Harvester developer
- CouchDB views and analysis
- Sentiment analysis code

- Front end developer & Visualization
- Architecture design

Sheng Wu

- Harvester developers
- CouchDB views and analyses
- Sentiment Analyses
- Documentation

Teeraroj Chanchokpong

- Extra topics: correlation, regressions analysis, visualization of correlation
- CouchDB views and analyses
- Support system administration
- Ansible playbooks
- Architecture design
- Documentation and report

References

Cowling, D. "Social Media Statistics Australia - January 2017." Social media news. Accessed on May 2017 from http://www.socialmedianews.com.au/social-media-statistics-australia-january-2017/

Hutto, C.J. & Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014.

Public Health Information Development Unit (PHIDU). Social Health Atlas of Australia (online). At: http://phidu.torrens.edu.au/social-health-atlases/data-archive/data-archive-social-health-atlases-of-australia#social-health-atlas-of-australia-data-released-may-august-december-2016-by-population-health-area-local-government-area-based-on-the-asgc-2011-and-asgs-2011-and-primary-health-network (06 May 2017).

Appendix

Raw OLS output

Total Death

OLS Regression Results

Dep. Variable:	total_death	R-squared:	0.072
Model:	OLS	Adj. R-squared:	0.035
Method:	Least Squares	F-statistic:	1.938
Date:	Mon, 08 May 2017	Prob (F-statistic):	0.131
Time:	20:06:33	Log-Likelihood:	-408.51
No. Observations:	79	AIC:	825.0
Df Residuals:	75	BIC:	834.5
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf. Int.]	
Intercept	247.1435	5.450	45.352	0.000	236.288	257.999
negSentiment	0.0790	0.057	1.396	0.167	-0.034	0.192
posSentiment	-0.2690	0.164	-1.645	0.104	-0.595	0.057
foodTweets	0.4891	0.457	1.070	0.288	-0.422	1.400

Omnibus:	0.273	Durbin-Watson:	2.108
Prob(Omnibus):	0.873	Jarque-Bera (JB):	0.272
Skew:	-0.131	Prob(JB):	0.873
Kurtosis:	2.882	Cond. No.	870.

Warnings:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Total Death from Ischemic Heart disease

OLS Regression Results

Dep. Variable:	pmHeart	R-squared:	0.046
Model:	OLS	Adj. R-squared:	0.008
Method:	Least Squares	F-statistic:	1.219
Date:	Mon, 08 May 2017	Prob (F-statistic):	0.309
Time:	20:06:39	Log-Likelihood:	-265.56
No. Observations:	80	AIC:	539.1
Df Residuals:	76	BIC:	548.7
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf. Int.]	
Intercept	25.4298	0.850	29.909	0.000	23.736	27.123
negSentiment	0.0128	0.009	1.440	0.154	-0.005	0.030
posSentiment	-0.0414	0.026	-1.614	0.111	-0.093	0.010
foodTweets	0.0782	0.072	1.090	0.279	-0.065	0.221

Omnibus:	11.292	Durbin-Watson:	1.703
Prob(Omnibus):	0.004	Jarque-Bera (JB):	14.316
Skew:	-0.643	Prob(JB):	0.000779
Kurtosis:	4.625	Cond. No.	865.

Warnings:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.