**Citizens’ response to Governments’ public policy on COVID-19**

ICT 1002 P3 Team 16: Tan Jia Jun Clarence (2102420), Pang Ka Ho (2102047), Wang Qixian (2101751), Zarifhaziq Bin Zulqhiffri (2101668), Tan Shireen (2102432)

[2102420@sit.singaporetech.edu.sg](mailto:2102420@sit.singaporetech.edu.sg), [2102047@sit.singaporetech.edu.sg](mailto:2102047@sit.singaporetech.edu.sg), [2101751@sit.singaporetech.edu.sg](mailto:2101751@sit.singaporetech.edu.sg), [2101668@sit.singaporetech.edu.sg](mailto:2101668@sit.singaporetech.edu.sg), [2102432@sit.singaporetech.edu.sg](mailto:2102432@sit.singaporetech.edu.sg)

***Abstract*—This report is the study of Citizens’ response to the Government’s public policies on COVID-19. The steps we took can be summarised into 3 main components: Data Scraping (Gathering data online), Sentiment Analysis (Analyzing the opinion of a sentence to a subject), and Topic Modelling (Detecting words and phrase patterns) using external libraries that are available online. Afterwhich, we will provide some useful insights and discussions, and draw conclusions based on the results gathered.**

***Keywords—covid, extraction, sentiment analysis, natural language processing, latent dirichlet allocation, modelling, citizens, machine learning***

1. INTRODUCTION

Ever since COVID-19 has been affecting our daily lives, the Singapore government has gone through different phases whereby each phase has its own precaution measures policy that are meant to be beneficial for the citizens health and safety.

1. OVERVIEW

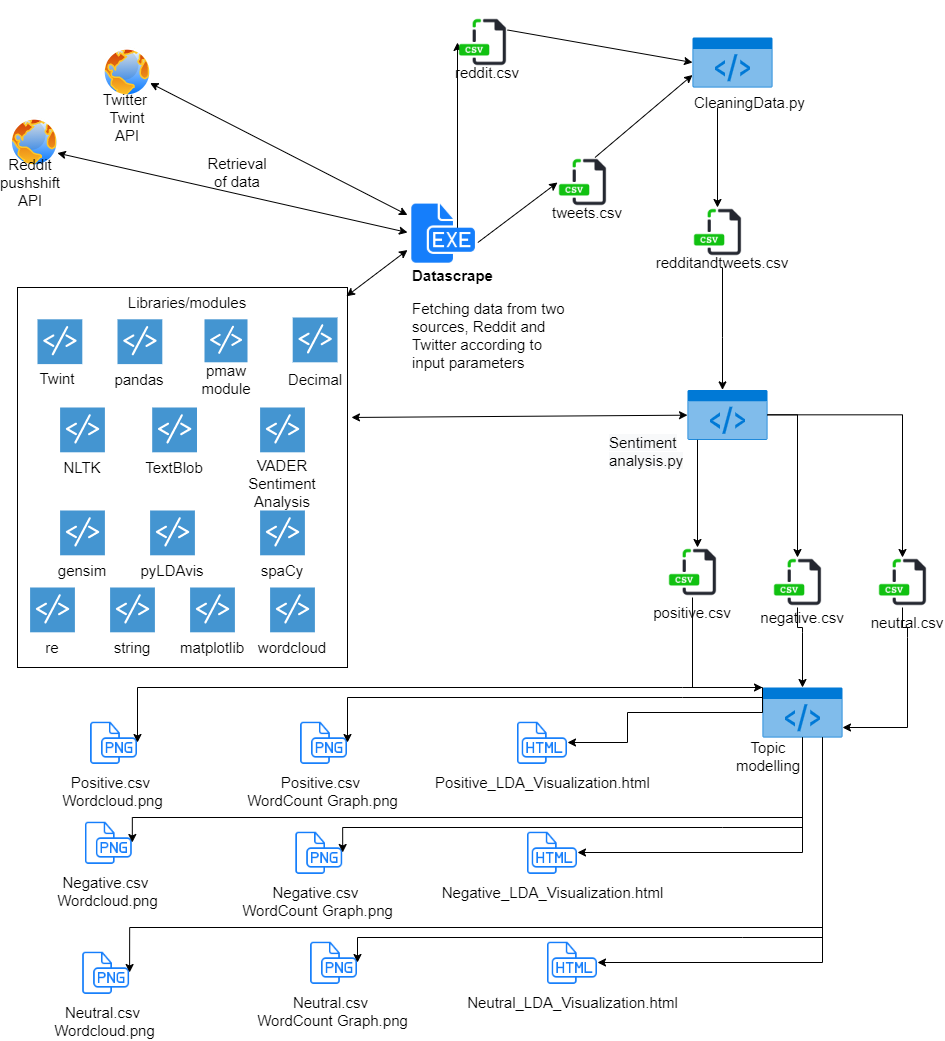
In this report, we will be looking at citizens’ response to Government’s public policies on COVID-19. The topics that we will be looking at are data crawling from social media platforms, using a sentiment analysis library to detect the sentiments of citizens’ discussion, as well as analysing the citizens’ response within each categorization of emotions by using topic modelling.

1. PROJECT IMPLEMENTATION
2. DATA SETS USED

Reddit is a forum website that allows discussion of a range of topics. The subpage r/singapore has over 400,000 subscribed members discussing a variety of Singapore related topics or issues. Its comments were more detailed as compared to other social media sites such as Facebook or Twitter. We plan to extract comments on “covid” topics through the subject header to discover Singaporean sentiments through time.

Tweets are another data set we’ve used. Tweets are generally short due to the word limit but are useful to detect sentiment changes across a set of time frames. We plan to find the keyword “covid sg” in tweets.

1. SYSTEM DIAGRAM



1. EXTRACTION OF DATA

1-5 are tools and libraries used to facilitate the extraction of data. 6 is the step-by-step explanation of extraction of data.

1. *PMAW wrapper for Pushshift API*

The PMAW wrapper uses multithreading to retrieve Reddit comments and submissions. It is significantly faster and better than using Pushshift API alone as it runs asynchronously using multiple threads. [1] Pushshift runs requests sequentially.

1. *TWINT - Twitter Intelligence Tool*

TWINT uses Twitter’s search operator to scrape Tweets relating to users, topics, trends etc. and is able to sort out and filter information. When compared to Twitter’s inbuilt API, it has no limits while Twitter’s API has 3200 tweet limit per fetch. It does not require an account nor API keys and can be done anonymously. [2]

1. *Pandas (library)*

A very popular data manipulation/analysis tool. Data extracted is manipulated through pandas dataframe and stored in a CSV file. It has other useful features such as concatenation of files etc.

1. *Tkinter GUI (In-built)*

A GUI tool for users to insert fields and select data as and when it is needed. This is useful for users who are unsure of command line or modifying fields in a .py file. Everything is simplified into a GUI. Parameters are then wrapped and passed through straight to servers.

1. *PyInstaller*

An installer to package and build executable files. As we have a few second-level imports in the Datascraper.py file, PyInstaller would not detect this. Hence, we need to specify the imports when running the packaging command. We would not want the terminal to be interfering with our GUI as well. The command would be:

*pyinstaller -F --windowed --hidden-import “-insert imports here-” datascraper.py*

1. *Extraction of data process*

Users would first interact with our Tkinter GUI by clicking on Datascraper.exe. They would select the type of scraper to choose (Reddit[default] or Twitter) and enter the necessary fields needed such as topic, subreddit, limits, filters, timeframe, output file. It is better for users to specify which column/data is needed to be downloaded rather than filter them later on. This would save more disk space for users who do not require all of the data fields.

The default fields are:

1. Body/Tweets (To show what are they talking about)
2. Date (The posted date)
3. Link (The direct link to the comment/tweet)

In our scenario, we chose “covid” as our topic and Singapore as our subreddit for reddit. For twitter our search topic is limited to “covid sg”. Limits are set to 100k and the timeframe started in January 2020 when the first few covid cases were detected to the current date.

Validation for fields occurs here. If there are missing parameters or fields that are required, a pop-up message will appear. Input fields such as “limits” also only allow number input from the keyboard. This is a feature of Tkinter.

The results will be sent as parameters to the TWINT or PMAW depending on Twitter or Reddit. Results would be stored in a Pandas dataframe, manipulated and then stored in a CSV file.

When the extraction is completed, a pop-up message will appear to indicate that it is done and would be stored in a CSV. There are a total of 50905 rows for Reddit’s raw data and 20900 Twitter’s raw data, totalling to **71805 rows of data**.

The columns that we found to be useful and kept are “author”, “body”, “date”, “link” and “score”. The score refers to reddit’s total upvotes or downvotes as well as Twitter’s likes count.

The output file names are redditdata.csv and covidtweets.csv respectively.

1. DATA PRE-PROCESSING

1-3 are libraries and tools we used to process our data while 4 is a walkthrough of the processing of data.

1. *Langdetect (Library)*

Language detection is important. We have to filter out non-English characters for our sentiment analysis and topic modelling. This is a re-implementation of Google’s language detection library in Python.

1. *FTFY (Library)*

There is a lot of broken unicode output in the results files. Some of the words have added symbols or missing gaps. It does not change already readable unicode but detects and fixes bad sentences.

1. *Miscellaneous formatting (In-built)*

It is normal for comments to contain HTML links and has HTML encoding such as “&gt” and “&amp”. This is not needed for the processing of data. Using HTML inbuilt function, we applied a function of unescaping those characters. Using regex, we were able to remove links from comments/tweets.

Reddit’s dates are in Unix timestamp format. To make it a more readable user-friendly format, we changed it to YYYY/MM/DD format instead.

Reddit links are in an extension format. It does not display the entire link. We have appended the front part of the URL to ensure that users are able to click on the link in the CSV file directing straight to the respective comment or tweet.

1. *Data processing process*

Data is cleaned each time whenever it is required. We try to make the data readable and plentiful for users, hence we try not to remove whole rows of data whenever we encounter “bad data” such as “ICT isnÃƒÂ¢Ã¢â€šÂ¬Ã¢â€žÂ¢t fun”. We try to make it readable and fix the errors.

When the data arrives in a DataFrame, we remove web links that are contained in comments/tweets. The body column is also applied with HTML decoding, using the unescape function. This solves issues such as “&amp &gt”

The date column is applied with a more readable date by converting UNIX timestamp to a human readable date. For reddit links, it is appended with the front part of the URL, making it clickable for users in the CSV file. Next, we read and concatenate both CSV files together to join their columns and sort it by date posted. Combining both files would make it easier for manipulation for the later tasks.

We then loop through every column through the FTFY library function, to fix unicode encoding issues. While looping, we also detect the language of the row. If it is not in English, the row will be replaced with a NaN value and then deleted thereafter.

After the removal of 9727 rows of data, we are left with **62078 rows** worth of combined Twitter and Reddit data. This dataset is then passed on to sentiment analysis, to give a score/weightage to it.

We considered using the spelling-correction function in a text processing library called “TextBlob” to pre-process the text content from social media posts in order to assist our sentiment analysis as it depends partly on correct word spelling to improve the accuracy of the analysis [6]. However, this turned out to be impractical as with a test dataset of 60 thousand rows, we estimated that it would have taken around more than a day to generate the output files (roughly 15 seconds for 10 rows). We have tried it but ultimately abandoned it for practical reasons.

1. SENTIMENT ANALYSIS

Sentiment analysis is the process of understanding the opinion of an author about a subject. In a sentiment analysis system, there are usually 3 elements [3]:

1. *Opinion and emotion*

Opinions can be positive, neutral or negative. Emotions could be qualitative, like joy, surprise, anger or disgust etc. It could also be quantitative, such as a rating on a movie from 1 to 10. [3]

1. *Subject of discussion*

There are times where two aspects of the same subject. For example, you can praise the screen brightness and colour vividness of a laptop, but criticize the poor battery life. [3]

1. *Opinion holder*

This is the person or entity expressing the opinion.

There are generally 3 types of sentiment analysis. [3] They are:

1. *Automated / Machine Learning Approach*

This is where you usually have existing pre-labelled data of known sentiments, and then using machine- learning algorithms and statistical techniques to train a model, which usually takes a while, based on the labelled data to try and predict sentiments from unlabelled data. [3]

1. *Lexicon / Rule-Based Approach*

This works by manually creating a dictionary of words and values, also known as a lexicon, and assigning a numerical value to a word, for example, “good, +1” or “bad, -1”. [3] Words that do not belong in the lexicon are given a default value of 0.

When these words are detected during the sentiment analysis, they are given a score. When the sentence is done processing, a compound score will be given at the end to evaluate the sentiment of the sentence, ranging from -1 (very negative) to +1 (very positive) (Hutto, 2021). Some libraries also account for subjectivity of the posts, which is more relevant when evaluating the neutrality of fact-based content, such as the reporting of news. [3]

In addition, amplifier words were also taken into consideration. The text “good food” will usually produce a positive score, but if an amplifier word is placed before the word identified in the lexicon, such as “not good food” or “very good food”, the score evaluation is also changed according to the intensity of the amplifier (Hutto, 2021).

1. *Hybrid Approach*

A hybrid approach uses both Machine-Learning and the Lexicon approach. Generally, it tends to perform the best out of the three approaches in terms of accuracy. However, it is also the most complex and costly implementation. [3]

Our project’s implementation is by using our selected library’s sentiment analysis type, which is the Lexicon / Rule-Based Approach. The sentiment analysis library chosen is VADER (Valence Aware Dictionary for sentiment Reasoning).

One of the main reasons that the Rule-Based Approach is chosen is because it is the fastest to implement and the easiest to learn, in terms of utilizing the library.

In addition, there is also a further consideration that there is no pre-labelled data regarding the sentiment of social media posts discussing COVID-19 that is easily available. If such data existed, then it would be possible to train a Machine-Learning Model, if we can build one. Another reason is because we lack the required expertise to custom-build our own Machine-Learning Model.

We also chose VADER because it is built especially for the social media domain, or what VADER describes as attuned to microblog-like context, which uses a combination of qualitative and quantitative methods to produce, and then empirically validate, what they describe as a “gold standard” sentiment lexicon that is especially attuned to short texts (Hutto & Gilbert, 2014).

This is especially relevant due to the nature of social media content as they are usually short and abbreviated, such as the texts we are getting from social media websites, like Twitter. Furthermore, VADER also has the capability to analyse UTF-8 coded emojis, which plays a crucial role in accurately analysing the sentiment of a post (Hutto, 2021).

As such, we thought it was paramount to choose a library with regards to its suitability to the domain of application.

For the sentiment analysis process, the user has to pass in the name of the data .csv file as an argument to the sentiment analysis main script and also run the command in a terminal as follows:

*py main.py dataFile.csv*

This passes the string of the file name to the main script, and the script first validates the user input by checking the length of the argument the user has provided. If the user has entered an incorrect number of arguments, the script prints the message, “Usage: py main.py dataFile.csv”, to the console and informs the user how to use the script, then terminates with an exit code of 1.

If the proper length of argument is provided. The script then tries to load the sanitized data csv file that is built through web crawling from Twitter and Reddit from our web scraping and data cleaning script. We accounted for the possibility that a wrong file name might be passed and the loading of the data failed, in this case, we handle the exception and print the message, “File not found, please check file name input.”, to the console to inform the user as such, and terminate the script with an exit code of 1.

Next, if the data .csv file can be loaded, we will use the pandas library function to load a .csv file into memory and instantiate a pandas DataFrame object to hold the sanitized data. We then proceed to analyse the DataFrame object row by row. This is done using the pandas DataFrame object index attribute, and iterating through it, then using the DataFrame object iloc method, which stands for integer locate method, to identify the row. This essentially allows us to iterate through the .csv file row by row till the end of the data file.

Each row is essentially a pandas Series object, which can be thought of as a Python dictionary, with the key of the dictionary specifying the column of the row, and the value of the dictionary specifying the value of the column at that row. As we iterate through the data row by row, we take the row’s “label”, which is the key for the Series object and the column name of the data for that specific row, and it returns us the value for that column. We take 3 labels in the process, which are the “body” (this column contains the text data of a social media post), the “date” and the “link” of the social media post.

We then check to see if the length of the text exceeds 280 characters (the limit for twitter). We do this because as posts from Reddit can go beyond what would normally be considered as short text, we had set a threshold to check on the length of the text during the sentiment analysis process. We set it to be in accordance with the maximum characters that a Tweet has, which is 280 characters. If such a threshold is exceeded, we use the TextBlob text processing library to break the text data into a list of sentences [6]. Next, we analyse the post sentence by sentence, by passing it into the VADER analyzer, while tallying up the sentiment score for the post, then dividing by the number of sentences to get the average score and use that score to evaluate whether the post is positive, neutral or negative. We had done so as to improve the accuracy of the sentiment and work with the suggested capabilities and limitations of the VADER library (Hutto & Gilbert, 2014).

If it does not exceed the character limit, we pass the whole text into the VADER analyzer, as it can handle shorter text with good accuracy. A score is generated, and we compare it against a suggested threshold to determine its sentiment.

The creators of VADER suggested a threshold score for the sentiment. We accepted their recommendation and set the thresholds for the sentiment as follows (Hutto, 2021):

1. *Positive sentiment: compound score ≥ 0.05*
2. *Negative sentiment: compound score* ***≤*** *-0.05*
3. *Neutral sentiment: compound score > -0.05 and compound score < 0.05*

Once the script finishes its analysis in a row, it then creates a data buffer that contains the analysed text with its emojis removed by calling a function in the script, as well as the score, date and link of the social media post. The emojis are removed as it will interfere with topic modelling. We then check the score against the recommended thresholds, and append it to one of three data file buffers, depending on their sentiments respectively.

When the script finishes its analysis on the entire DataFrame object instantiated from the sanitized data, we instantiate 3 more DataFrame objects for each of the sentiments, and construct the DataFrame object by passing in the data file buffers created during the analysis of the data.

Once the 3 sentiment Data Frame objects are created, we use the pandas library to generate 3 output csv files (positive.csv, neutral.csv and negative.csv), where each file contains social media posts that belong to their respective sentiment. Each row contains the row number, the text content of the social media post with emojis removed, the score computed to calculate the sentiment, the date of the post and the link to the post. The script finally terminates with an exit code of 0.

The output files are then sent off for topic modelling within each sentiment.

1. TOPIC MODELLING

Topic Modelling is a machine learning technique that is capable of going through sets of documents while detecting word and phrase patterns within them, as well as automatically clustering word groups and similar expressions that best characterize them. For this section, we used the LDA (Latent Dirichlet Allocation) topic modelling technique.

With regards to identifying citizens’ response to COVID-19 policies, topic modelling could be used to detect patterns and recurring words online based on the different emotions detected in Sentiment Analysis. By doing so, we are able to detect patterns such as word frequency that are similar within the citizens' feedback. 1-4 denotes the libraries that were used to conduct topic Modelling, while 5 describes the process conducted for topic modelling.

1. *Pandas (library)*

Pandas initializes the 3 output data files from Sentiment Analysis (positive.csv, negative.csv and neutral.csv) to be read from and undergo another round of data cleaning.

1. *NLTK*

NLTK is a platform used for building python programs that work with human language data for application in statistical natural language processing (NLP). [7] It contains text processing libraries such as Corpus, Word Tokenization and Wordnet which we have used to split the data text into smaller sections to be able to categorize and group them easier.

1. *SpaCy, String and Re*

SpaCy is an advanced NLP library that provides analysis and understanding of data and is capable of detecting languages and sentences as well as lemmatization. [8] String is a built-in python library used to interact with strings, such as to lowercase a string etc. [9] Re is also a built-in python library used to provide regular expression matching operations. [10]

1. *Gensim*

Gensim is also a natural language processing package, similar to NLTK, but provides an additional feature of providing algorithms such as LDA or LSI in order to fabricate high-quality topic models. [11]

1. *Process for Topic Modelling*

In order to conduct topic modelling, Pandas was first initialised to read each .csv file that were output from the sentiment analysis phase, based on user’s input via arguments, as shown below,

*py main.py <1, 2 or 3>*

where 1 signifies the program to run on negative.csv, 2 to run on neutral.csv and 3 to run on positive.csv.

Upon reading each .csv file, the data in each tweet or post need to be cleaned. This is because in order to present the results of topic modelling, we require the output to be formatted in single words so as to derive which topics are being discussed for each emotion category. For this process, SpaCy was used in conjunction with string and re in order to clean the data. SpaCy was used to remove non-English sentences, while string and re were used to remove punctuations, URLs, usernames and newlines.

However, the second part of data cleaning involves the use of NLTK’s corpus function to remove unnecessary words such as “this” or “that” or “said”; words that are not relevant to the task. This further narrows down our results to only relevant words and data.

Once data cleaning is completed, Gensim’s function corpora is then used to store the clean data into a dictionary. The LDA model is then generated through Gensim, and then trained using the same dictionary stored earlier, which was converted to a Document-Term Matrix (DTM) through Gensim as well.

A document-term matrix is a representation of the relationship between documents and terms where each row represents a document and each column represents a row. An entry is then added, which signifies the frequency of the term in the document. [13]

Training the LDA model on the DTM allows the LDA model to easily find repeating term patterns within the DTM, resulting in numerous topics being classified based on a number of similar keywords picked up by the model.

Once the LDA model has completed topic modelling, the topics will then be output into a visual format, in order for us to read and gauge which topics are being discussed within each emotion category. This process will be described in the next section.

1. DATA ANALYSIS

Data analysis is the process of collecting, modelling and analyzing data to extract insights that support decision-making. 1-3 denotes the libraries used for this section, while 4 denotes the process of using libraries 1 to 3.

1. *WordCloud*

WordCloud is a tool used to represent text data visually, in which the size of each word signifies the dominance of that word in a text, usually through importance or frequency.

1. *Matplotlib*

Matplotlib is a library used to visualize data in various formats, such as charts, graphs, and provide animation and interaction with visual outputs.

1. *pyLDAvis*

pyLDAVis is a library dedicated to visualize data that has undergone topic Modelling. This library allows users to interpret each topic by inspecting each topic visually, from the keywords dominant in the topic etc.

1. *Process for Data Analysis*

In order to analyse the data, the 3 libraries, WordCloud, Matplotlib and pyLDAvis have been used to create the visualisation based on each of the emotions that we have detected in the sentiment analysis process.

Samples of output visualisation graphs that have used Matplotlib and Wordcloud library are shown in Figures 1 and 2. The trained LDA model will find repeating words and patterns and will then narrow down a total of 10 topics within each emotion in regards to the social media posts regarding citizens’ response towards the government’s public policies on COVID-19.

As a sample output data for analysis, out of the 10 topics generated based on the negative.csv file from sentiment analysis output, Topic 8 was chosen as an example to demonstrate the possible inferences and deductions based on the visualization of topic Modelling via WordCloud. Based on Figure 1, some of the words that have appeared in Topic 8 are “infected”, “severe”, “disease”, “risk” and “vaccine”.

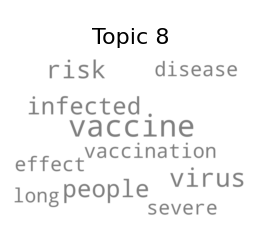
****

Fig. 1. Negative.csv Wordcloud Output of Topic 8

Through this, we can deduce the possibility that some citizens are not as willing to take the vaccine due to some of the risks from it or, the mindsets of the citizens are only based on the negative side of COVID-19 such as the rate of the spreading of virus and the severity and risk of vaccination. The entire output of all 10 topics per emotion category can be exported by the user into a .png format, for more thorough analysis of each topic, possibly by comparing the keywords of two different topics etc.

As a sample output data for analysis, out of the 10 topics generated based on the positive.csv file from sentiment analysis output, Topic 8 was chosen as an example to demonstrate the possible inferences and deductions based on the visualization of topic Modelling via Word Count. In Figure 2, the data shows the total count of the word that has recurred along with the weightages as the axis of the graphs. Based on Figure 2, some of the words that have appeared are “support”, “business”, “work” and “job”.

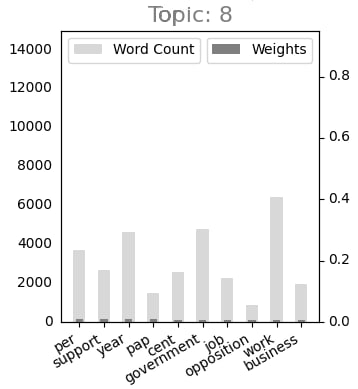
**

Fig. 2. Positive.csv word count graph output of Topic 8

Through this, we can deduce the possibility that the government’s COVID-19 public policies may have provided help and support to the businesses or, have introduced new jobs in the market for those who may have lost their jobs during the pandemic. The entire output of all 10 topics per emotion category can also be exported by the user into a .png format, for more thorough analysis of each topic, possibly by comparing the keywords of two different topics etc.

The visualisation graphs that used pyLDAvis library are shown in Figure 3. It also makes use of the LDA model to find recurring words in the data. However, in contrast to the wordcloud and word count graphs, pyLDAvis visualises the topic using an interactive HTML page which is capable of viewing the different keywords and maps based on the topics. [12]

As a sample output data for analysis, the topics and keywords generated are based on the negative.csv file as shown in Figure 3 and 4. In the left side of the visualisation (Figure 3), it shows the view of our topic model whereby there are numbered bubbles that represent each individual topic and term of the citizens’ response. The size of the bubbles will be determined based on the percentage of tweets in the data between topics. As for the distance between the bubbles, the further the bubbles are away from each other, the more different the topics are. [12]

On the right side of the visualisation (Figure 4), the blue horizontal bars represent each term that are most useful in interpreting the topics based on the overall frequency. To link both left and right side of the visualisation, there is a function at the top left (figure 5) to select the topics which would reveal the useful terms of the individual topic.

The slider tool in the top right of the graph which we are able to interact with to set and change the lambda frequency to different values accordingly is able to sort words based on the most relevant terms and change the rankings of them. By default, the lambda value would be set at 1. However, if we set it at a lower number, users will be able to understand and acquire more specific terms which will then allow us to dive deeper into the data and get more classified terms that are relevant to the topics.

Based on Figure 4, some of the words that have appeared are “people”, “virus”, “health”, “home”, “work” and “medical”.

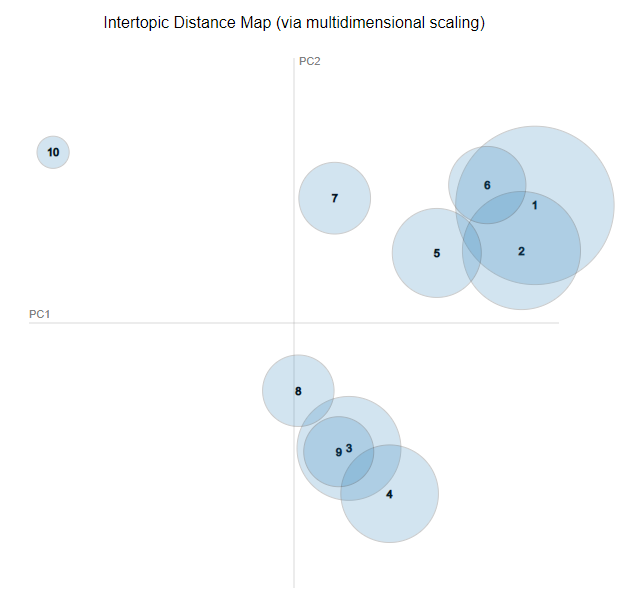
**

Fig. 3. Negative.csv pyLDAvis html bubble graph

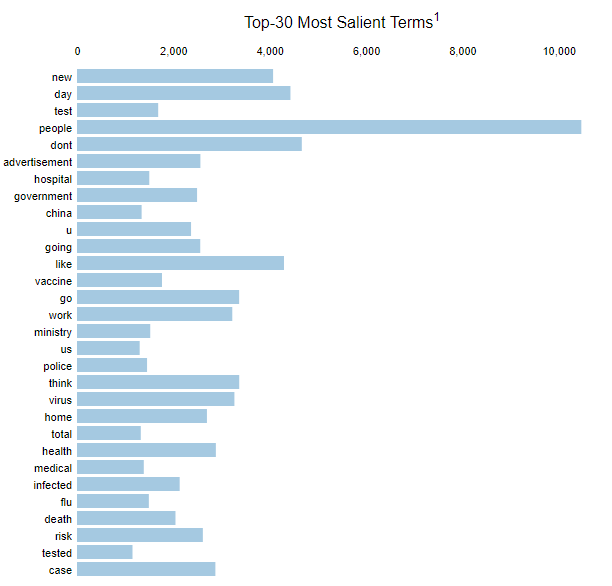


Fig. 4. Negative.csv pyLDAvis html bar graph

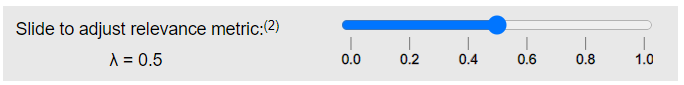


Fig. 5. Negative.csv pyLDAvis html slider tool

Through this, we can deduce the possibility that the citizens are concerned about their own health and the medical concerns due to COVID-19 and do not wish to leave their house to work during this current situation. The entire output of the topics per emotion category will be exported in the folder into a .html format, for more thorough analysis of each topic using the slide tool and interactions with the numbered bubbles.

1. RESULT AND INSIGHT

Both positive and negative sentiments are of similar number of data rows. **27203**(positive) vs **21537**(negative) vs **13338**(neutral). This shows that there are generally mixed sentiments across the board on the internet about the topic of COVID-19.

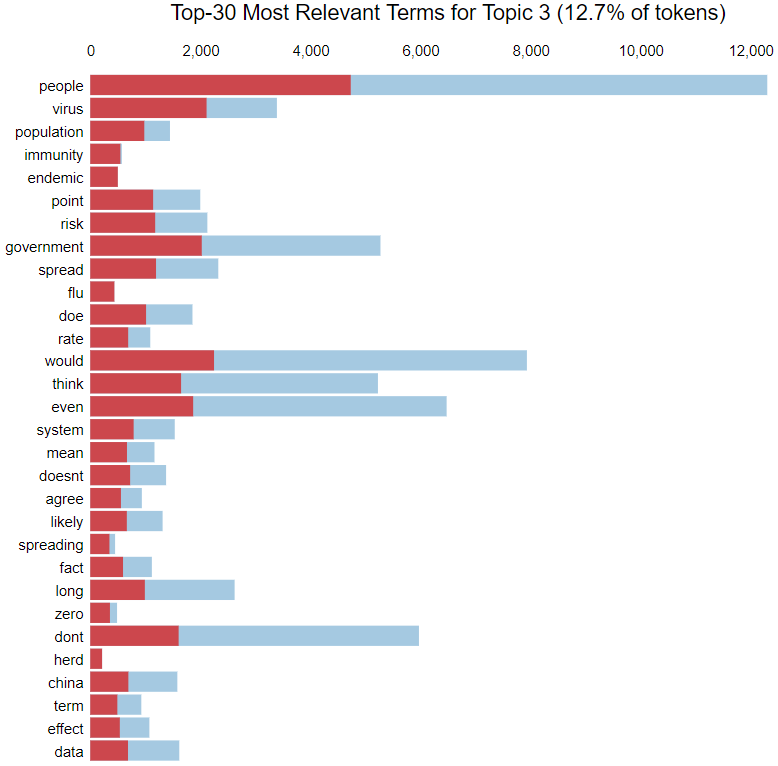


Fig. 6. Positive.csv pyLDAvis Output of Topic 3

Starting off with positive results, from topic 3 of the LDA visualisation in Figure 6, netizens were generally happy about the government's stance on opening up the country by treating it as endemic. This can be shown by keywords such as “immunity”, “herd”, “endemic”, “population”.

Citizens might be tired of social restrictions. Hence, when the Government announced that restrictions would be loosen through the planned endemic roadmap as we transit to the next stage, citizens were favourable to that idea.

With regards to the negative sentiment, we will first be looking at the negative.csv word cloud graph where it will show some of the more frequently used words. Next, we will be looking at the negative.csv word count graph where it has the same keywords as the word cloud graph but with weightage. Lastly, since we are not able to entirely deduce the topic based on the negative.csv Wordcloud and Word count graphs, we will go in-depth and use the pyLDAvis visualisation to determine what the topic is exactly.

Based on Figure 7, we have analysed and determined that topic 9 was the most relevant to our analysis. Some of the most relevant terms include “test”, “doctor”, “fever”, “swab”, “school” and “cough”. By adjusting the relevance slider even further, we are able to conclude that topic 9 is about COVID-19’s medical symptoms. This shows the negative side of how the pandemic has affected the citizens' health and their mindsets of being paranoid for example when they are required to do a swab test if they are suspected with a COVID-19 symptom which was previously, before the pandemic, was just a common illness.

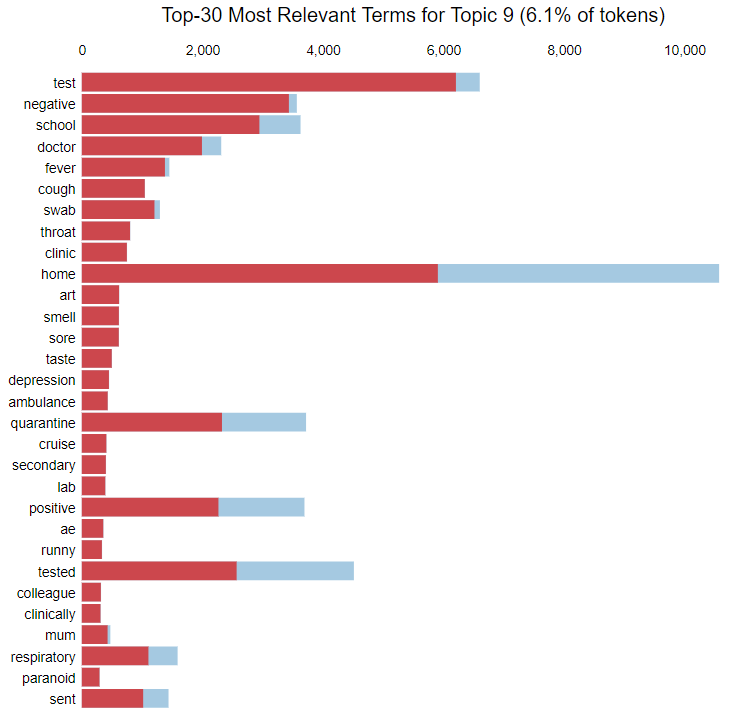


Fig. 7. Negative.csv pyLDAvis Output of Topic 9

Moving on to the neutral sentiment, we analysed all topics and found topic 3 to be the most relevant to our analysis on citizens’ response to government measures on COVID-19. We can see that in Figure. 8, the top 30 most relevant terms include terms such as “business”, “time”, “flight” and “travel”.

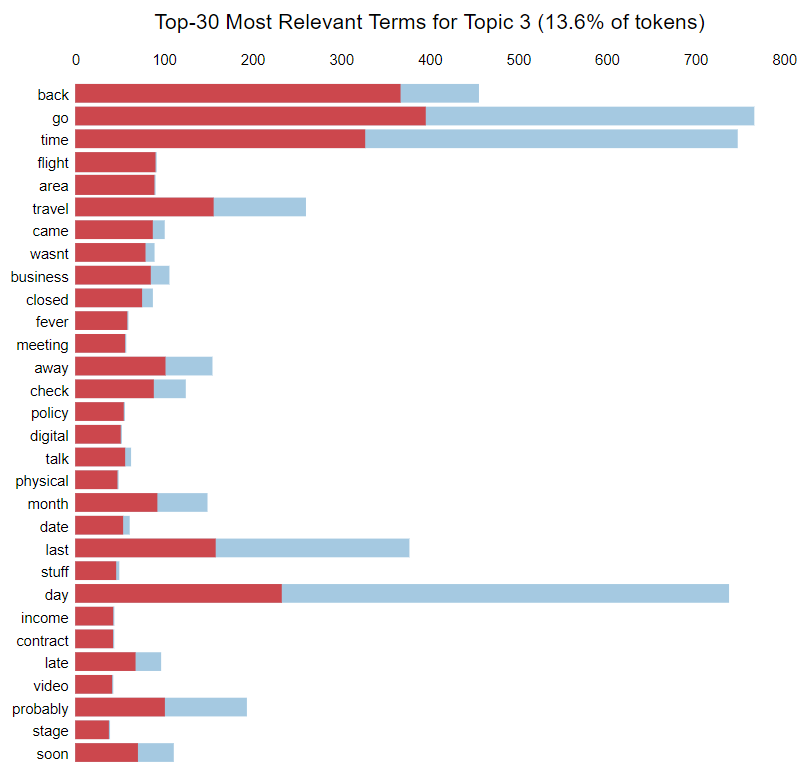


Fig. 8. Top 30 most relevant terms for topic 3 from pyLDAvis

We leaned towards the conclusion that topic 6 is generally about the government's measures around business travel. We suspect that social media users are generally neutral towards the government’s measures towards allowing travel for business because of the need for Singapore to do so (of being an international hub). Our data shows that the general sentiment and discussions are neutral towards these measures.

1. CONCLUSION

Overall, citizens’ sentiments of their response towards the government’s public policies are mixed across different topics. There are times where sentiments are split amongst a single topic (e.g on opening up the country) whereby there are citizens who are for and against it. The Government would need to strike a balance between the groups so that everyone or the majority remains happy and implement policies that would be beneficial for all.

We have learnt a lot of new skills through this project. Examples are:

1. Data Science (Pandas, Data manipulation, NLP etc)
2. GitHub source control
3. Library selection given the task and timeframe, to produce a reasonable project of reasonable quality within the given timeframe
4. Discovering useful insights from the clean data that we have collected.

For our future improvements, we could plot and align sentiment dates according to each public policy the Government has rolled out. This may give a better sense of real-time emotions that are talking about the Government’s actions.

The breakdown of the tasks is as follows:

1. Data scraping (Clarence)
2. Sentiment analysis (Ka Ho and Qi Xian)
3. Topic modelling and plotting (Zarif and Shireen)

ACKNOWLEDGEMENTS

We had put in tremendous effort into making this project. However, this would not have been possible without the dedicated support and guidance of Professor Arthur Loo Wee Yeong throughout our entire project. We would also like to express our earnest thanks to Professor Wang Zhengkui for his consultation as well as Ms Jenty Chou for discussing the submission requirements with the group.

REFERENCES

1. Podolak, M. (n.d.) *Mattpodolak/PMAW: A multithread pushshift.io API wrapper for reddit.com comment and submission searches.* GitHub. Retrieved October 24, 2021 from https://github.com/mattpodolak/pmaw.
2. Cervetto, M. (2019). *TwinT - overview*. GitHub. Retrieved October 24, 2021, from https://github.com/twint.
3. MonkeyLearn. (n.d.). *Sentiment analysis: The go-to guide. MonkeyLearn.* Retrieved October 24, 2021, from https://monke

ylearn.com/sentiment-analysis/

1. Hutto, C. J. (2021, March 16). CJ HUTTO/ Vadersentiment: *Vader sentiment analysis.* GitHub. Retrieved October 24, 2021, from https://github.com/cjhutto/vaderSentiment.
2. 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.
3. Tutorial: Quickstart - TextBlob 0.16.0 documentation. (n.d.). Retrieved October 24, 2021, from https://textblob.readthedocs.

io/en/dev/quickstart.html.

1. Techopedia. (2014, August 2). *What is the Natural Language Toolkit (NLTK)? - definition from Techopedia*. Techopedia.com. Retrieved October 24, 2021, from https://www.techopedia.com/definition/30343/natural-languag

e-toolkit-nltk.

1. *Spacy 101: Everything you need to know · spacy usage documentation.* spaCy 101: Everything you need to know. (n.d.). Retrieved October 24, 2021, fromhttps://spacy.io/usage/

spacy-101.

1. *String - common string operations.* Python 3.10.0 documentation. (n.d.). Retrieved October 24, 2021, from https://docs.python.org/3/library/string.html.
2. *Re - regular expression operations*. Python 3.10.0 documentation. (n.d.). Retrieved October 24, 2021, from https://docs.python.org/3/library/re.html.
3. Prabhakaran, S. (2018, October 16). *Gensim tutorial - A complete beginners guide*. Machine Learning Plus. Retrieved October 24, 2021, from https://www.machinelearningplus.com

/nlp/gensim-tutorial/.

1. Tran, K. (2021, July 20). *Pyldavis: Topic Modelling Exploration Tool that every NLP data scientist should know*. neptune.ai. Retrieved October 24, 2021, from https://neptune.ai/blog/pyldavis-topic-modelling-exploration-tool-that-every-nlp-data-scientist-should-know.
2. Zhao, Y. (2013). *Document Matrix - an overview*. ScienceDirect Topics. Retrieved October 24, 2021, from https://www.sciencedirect.com/topics/mathematics/document-matrix.