

**NANYANG
TECHNOLOGICAL
UNIVERSITY**
SINGAPORE

**CZ4032 Information Retrieval
Project Report**

Name	Matric Number
Poh Ying Xuan	U1821489B
Prem Adithya Suresh	U1820740B
Muhammad Al-Muhazerin	U1921191L
Tan Jian Wei	U1820308E
Kong Alson	U1822899B
Sam Jian Shen	U1821296L

Introduction	3
Crawling	4
Question 1	4
How you crawled the corpus (e.g., source, keywords, API, library) and stored it (e.g., whether a record corresponds to a file or a line, meta information like publication date, author name, record ID)	4
What kind of information users might like to retrieve from your crawled corpus (i.e., applications), with sample queries	6
The numbers of records, words, and types (i.e., unique words) in the corpus	7
Indexing and Querying	8
Understanding SOLR query	8
Define relevant documents and ranking in our context	9
Define our measurement	10
Question 2	11
Build a simple web interface for the search engine (e.g., Google)	11
Write five queries, get their results, and measure the speed of the querying	17
Explore some innovations for enhancing the indexing and ranking. Explaining why they are important to solve specific problems	18
Further preprocessing	18
Problems	20
Handle Code or Company Queries	20
As shown in the 3 result table above, there are irrelevant entries amongst them which are highlighted in yellow. The reason for such results is because the query is only interested to compute the given field. Thus, we have to optimize the queries that also consider other fields as well.	21
Handle Typo Mistakes Queries	21
As you can see based on the above top 10 results. It is not able to query properly due to a minor typo mistake. Thus, has resulted in less fruitful or no results return.	22
Solutions	22
Handle Code or Company Queries	22
Handle Typo Mistakes Queries	27
Classification	30
Question 4	30
Motivate the choice of your classification approach in relation with the state of the art	30
Discuss whether you had to preprocess data and why	32
Build an evaluation dataset by manually labelling 10% of the collected data (at least 1,000 records) with an inter-annotator agreement of at least 80%	39
Provide evaluation metrics such as precision, recall, and F-measure and discuss results	37
Discuss performance metrics, e.g., records classified per second, and scalability of the system	38
A simple UI for visualizing classified data would be a bonus (but not compulsory)	39
Question 5	40

Explore some innovations for enhancing classification. Explain why they are important to solve specific problems, illustrated with examples.	40
Conclusion	49
Appendix A - Submission Links	50
Appendix B - How to use the program 101	51

Introduction

A rising stock market is usually aligned with a growing economy phenomenon and leading to greater investor confidence. As such, Investor confidence in stocks would lead to more buying transactions activity which can help to push prices higher. When stocks rise, people invested in the equity markets gain wealth. This attracts a lot of youngsters or new immature investors to invest in stocks.

This has created a surge in immature investors who spend their personal money unwisely on stocks itself. The consequence resulted in spike cases of immature investors gone bankrupt, domestic violence due to loss of savings over investment. Thus, this has motivated us to create this search engine to help facilitate immature stock picking investors.

In the current Internet of Things (IoT), there have been many solutions provided that help to decide what stocks to invest in. Our project goal is to provide another alternative solution to contribute by seeking basic understanding of the sentiment of the stock market that may influence via Twitter.

Assumption

1. The investor does not acquire deep knowledge of stocks. Thus, we assume that they will search the stock information based on company code or company name of the stock for most of the time.
2. The investor mostly concerns about the sentiment analysis of the stock result which is polarity in our project in order to determine if the stock is positive or negative.

Crawling

We used Twitter as the source of information about stock. We will crawl wikipedia to gather basic information about the different stock companies. Using that information, we will crawl Twitter, gathering information about the user's opinion about a particular stock. These data will then be pre-process, classified and be used for indexing, querying and categorization in the later stage.

Question 1

1. How you crawled the corpus (e.g., source, keywords, API, library) and stored it (e.g., whether a record corresponds to a file or a line, meta information like publication date, author name, record ID)

Symbols	Company
MMM	3M Company
ABT	Abbott Laboratories
ABBV	AbbVie Inc.
ABMD	ABIOMED Inc

A sample of data in stock_companies.csv

First, we crawled for stock companies data on Wikipedia and stored data for 505 companies in stock_companies.csv file. In the CSV file, each stock company's data is separated into 2 fields: symbol and company. An example of the stored data is shown in the table above.

```
{
  "statuses": [
    {
      "created_at": "Sun Feb 25 18:11:01 +0000 2018",
      "id": 967824267948773377,
      "id_str": "967824267948773377",
      "text": "From pilot to astronaut, Robert H. Lawrence was the first African-American to be selected as an astronaut by any na... https://t.co/FjPEWnh804",
      "truncated": true,
      "entities": {
        "hashtags": [],
        "symbols": [],
        "user_mentions": [],
        "urls": [
          {
            "url": "https://t.co/FjPEWnh804",
            "expanded_url": "https://twitter.com/i/web/status/967824267948773377",
            "display_url": "twitter.com/i/web/status/9...",
            "indices": [

```

Sample of the Response

Using our developer account, we utilised Twitters' Standard Search API that allowed us to search for stock related tweets within the last seven days. We used the stock companies from the stock_companies.csv file as the query for our search. For every row, we queried for 50 tweets that were related to either the symbols or the company names. We figured that this combination would provide the most relevant stock tweets to build our corpus and ensure that every stock company has a substantial pool of data for us to work with. A sample of the response is shown in the picture above.

	A	B	C
1	text	datetime	symbol
2	#Madison #tech company @MoxeHealth secures mc	Wed Feb 10 16:40:07 +0000 2021	MMM
3	@ArchLuminous @LevelToPower @rickballan @orn	Wed Feb 10 16:32:41 +0000 2021	MMM
4	#MMM boom something big coming here imo - 40%	Wed Feb 10 16:19:33 +0000 2021	MMM
5	#MMM TR1â€™s 40% of the company now taken ou	Wed Feb 10 16:10:44 +0000 2021	MMM
6	#MAC incredible 5.3m shares traded today or 10% o	Wed Feb 10 15:51:21 +0000 2021	MMM
7	I feel like \$OPTI is slowly becoming another 3M type	Wed Feb 10 14:07:11 +0000 2021	MMM
8	Dividend Increase 3M Company (MMM) https://t.c	Wed Feb 10 13:44:02 +0000 2021	MMM

Sample Data from tweets.csv

The response provided a sizable amount of parameters. However, not all fields are required for our task. Hence, we only store the necessary information, namely the

parameters of the text, created_at, and symbol into tweets.csv. These data will be pre-process later. A sample of the stored data in tweets.csv is shown in the picture above.

2. What kind of information users might like to retrieve from your crawled corpus (i.e., applications), with sample queries

As our targeted audience are beginner investors, the amount of information provided through our system would be sufficient to enrich the investor's knowledge. The information that users would retrieve by using our system are technology stocks information which are classified into the following:

- Company name
- Code
- Company Text (such as a famous name etc.)

The possible queries example are as follows:

1st possible query by Company Name: Walmart

The above query would display all the records that are related to the company name 'Walmart'.

2nd possible query by Code: TWTR

The above query would display all the records that are related to the code 'TWTR', which the code is referring to the Company 'Twitter, Inc.'.

3rd possible query by Company Text: Elon Musk

The above query would display all the records that are related to the famous entrepreneur 'Elon Musk'.

3. The numbers of records, words, and types (i.e., unique words) in the corpus
Our corpus is static. Therefore, using a simple python script, there are **16,595 records**, **391122 words**, and **81927 unique words** in the corpus.

Indexing and Querying

Understanding SOLR query

Solr is a NoSQL database and document structured database that offers SQL support and executes it in a distributed manner. Solr index is a list that holds the mapping of words, terms, or phrases and their corresponding places in the documents stored.

Solr uses fields to index a document. However, before being added to the index, data goes through a field analyzer, where Solr uses char filters, tokenizers, and token filters to make data searchable. Character filters can make changes to the string as a whole. Then, tokenizers break field data into lexical units or tokens that then pass through filters which decide to keep, transform (e.g. setting all the data to lowercase, removing word stems) or discard them, or create new ones. These final tokens are added to the index or searched at query time.

The query is parsed by a URL string and it will return dictionary data which consists of records of the matched document. SOLR also offers features to optimize the indexing and ranking.

```
http://localhost:8984/solr/test_server/select?
fq=polarity%3A2&
q=text%3Abitcoin%2C%20text%3Aelon%20musk
```

- It consists of local address, port number, the domain name and selected queries
- SOLR offers different variations of query to enhance the query result such as "q=" and "fq="
- These are operation key characters encoded as defined by SOLR. For example %3A refers to ':' character
'&' character performs AND operation
- ■ These are document key-value pairs to query the field and its value respectively.

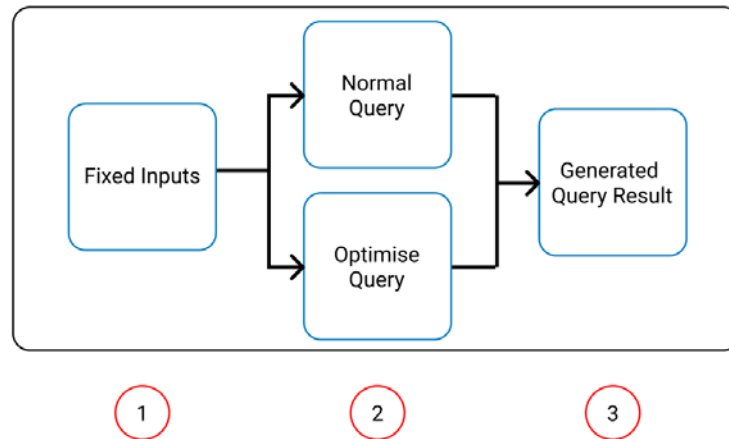
Reference: https://solr.apache.org/guide/6_6/function-queries.html

Define relevant documents and ranking in our context

To improve our query, it is important to understand what is considered a relevant document to us. Firstly, we have to put ourselves into our target audience's mindset. Below are the factors based on our research and understanding of stocks concerning our target audiences. Thus, our query should optimize only based on these factors.

1. The tweets have a probability to influence the stock market such as Elon Musk's tweets about dogecoin has resulted in a surge in the stock price value.
2. The company code and name are typical keywords used to communicate about stocks. For example, Gamestop (GME)
3. The stock price is a numeric value to observe their trends and worth.
4. The target audience is not expected to search for specific phrases about the tweet but rather the keywords that influenced the stocks only. Such as names of influential people, trendy stocks, etc.

We end up using a manual way to do it. This is because our context has limitations to achieve the other approaches. Instead of looking at the entire entries we selectively do it based on the factors the user likely will search. Thus we are only testing our effectiveness of query on a sample scale size. Below shows the figure about how we are going to evaluate the result and improve them.



1. The inputs are selected based on the likelihood of a known result or . The inputs are a list of strings like how the user will query into the search bar.
2. The inputs are then parsed into 2 classified types of query, one is an optimized query and another is a normal query. In this way, we can easily use them to compare how good our performance can be. The type of query can be configured via the “mysite/static/home/views.py” configuration section.
3. The query result is written into CSV format for observation and comparison. Below is an example of the generated results.

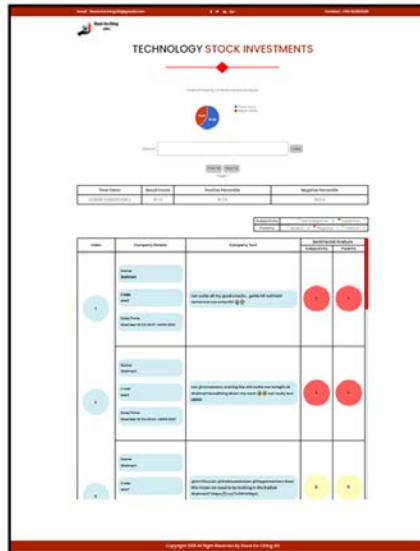
s/n	test_case	rank	records	pos%	neg%	time_taken	text	company_name	code	subjectivity	polarity	date_time
20	Elon Musk	1	5	50	50	0.012020588	DogeCoin Spikes 140%, Dogecoin, Elon Musk, Cryptocurr	Tesla, Inc.	TSLA	0	0	Tue Mar 09 23:43:17 +0000 2021
20	Elon Musk	2	5	50	50	0.012020588	DogeCoin Spikes 140%, Dogecoin, Elon Musk, Cryptocurr	Tesla, Inc.	TSLA	0	0	Tue Mar 09 23:43:48 +0000 2021
20	Elon Musk	3	5	50	50	0.012020588	#Bitcoin/xad/xad/xad/xad/xad/xad/xad/xad/777N?77777	BlackRock	BLK	0	0	Wed Mar 10 00:23:01 +0000 2021
20	Elon Musk	4	5	50	50	0.012020588	#Bitcoin Musk just hit a new milestone: He made a record \$1	Tesla, Inc.	TSLA	1	2	Wed Mar 10 04:28:44 +0000 2021
20	Elon Musk	5	5	50	50	0.012020588	@DavidWilliams @GregoryDoge Elon sets highly ambig	Boeing Company	BA	1	1	Tue Mar 09 06:46:33 +0000 2021
21	GameStop	1	15	77.8	22.2	0.019379477	@MagkarPhmYs Your gamestop turned into a Walmart	Ventur	VNT	0	0	Wed Mar 03 12:14:31 +0000 2021
21	GameStop	2	15	77.8	22.2	0.019379477	Discovery Communications, Inc. (NASDAQ:DISCA), Game	Discovery, Inc.	DISCA	1	2	Tue Mar 09 05:07:56 +0000 2021
21	GameStop	3	15	77.8	22.2	0.019379477	Discovery Communications, Inc. (NASDAQ:DISCA), Game	Discovery, Inc.	DISCA	1	2	Tue Mar 09 05:07:56 +0000 2021
21	GameStop	4	15	77.8	22.2	0.019379477	@jimcramer Boeing is cool but have you heard about th	Boeing Company	BA	1	2	Tue Mar 09 05:07:56 +0000 2021
21	GameStop	5	15	77.8	22.2	0.019379477	Among the day's winners were Tesla Inc (NASDAQ:TS	Tesla, Inc.	TSLA	0	0	Tue Mar 09 23:21:41 +0000 2021
21	GameStop	6	15	77.8	22.2	0.019379477	Among the day's winners were Tesla Inc (NASDAQ:TS	Nasdaq, Inc.	NDQ	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	7	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	8	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	9	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	10	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	11	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	12	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	13	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	14	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	15	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	16	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	17	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	18	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	19	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	20	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	21	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	22	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	23	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	24	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	25	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	26	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	27	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	28	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	29	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	30	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	31	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	32	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	33	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	34	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	35	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	36	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	37	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	38	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	39	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	40	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	41	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	42	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	43	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	44	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	45	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	46	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	47	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	48	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	49	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	50	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	51	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	52	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	53	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	54	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	55	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	56	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	57	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	58	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	59	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	60	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	61	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	62	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	63	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	64	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	65	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	66	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	67	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	68	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	69	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	70	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	71	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	72	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	73	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	74	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	75	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	76	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	77	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	78	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	79	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	80	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	Tue Mar 09 23:55:03 +0000 2021
21	GameStop	81	15	77.8	22.2	0.019379477	George Meletoe @Yusufverett Caterpillar's Trimsly Sak	Diamondback Energy	FANG	0	0	

Question 2

1. Build a simple web interface for the search engine (e.g., Google)

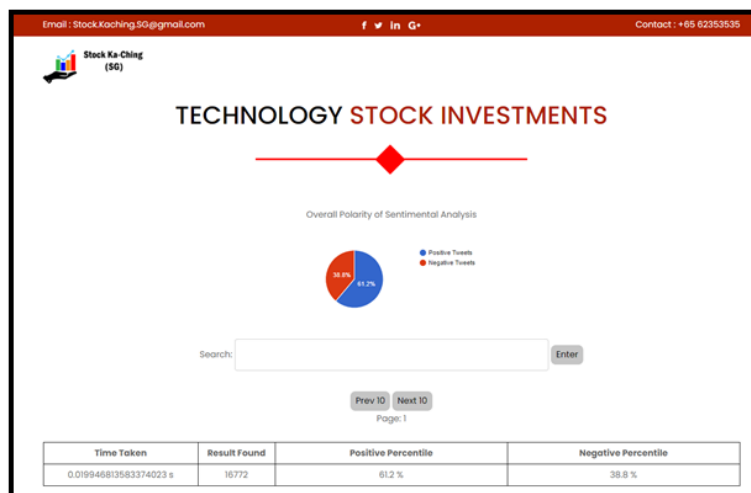
For this project, as the focus is not about the complexity of the design for the search engine, we have decided to go with a simple but yet user-friendly web interface with a light colour theme to enhance the visual experience of our users. All the information would be displayed in one single page for the ease of use and to improve the time taken for each search query. To add on, we have experienced that with the extra graphical content, it would heavily increase the search time of each search query due to the impact of longer rendering speed. As such, we have decided to remove unnecessary images and animated carousel that was previously used to beautify the web interfaces.

Our targeted audience would be anyone who is interested with technology stocks. Through the website, the user would be able to retrieve the basic stock information and be able to visualize how good or bad each of the technology stock is doing through sentimental analysis which will be mentioned in detail in Section 4.3 – Classification. The software used in this project would be Django with Solr, and Django would be used mainly for web development in conjunction with Solr which runs as a standalone full-text search server for implementation of the search engine. The programming language used for designing of the user interface is HTML5 (Hypertext Markup Language), CSS (Cascading Style Sheet) for the presentation of the web interfaces and JavaScript for manipulating the script which creates the interactive portion to capture our audience's attention upon using the web search engine.



Overview of the Web Search Engine Page

Above depicts an overview of the webpage in which the details of each function would later be explained in separate portions.



Home.html (Partial of the Homepage, Top Portion)

Above depicts a partial presentation of the Homepage. (Top portion of the Home page)

To begin with, a navigation bar with a red background colour and information was created at the top of the web portal. A logo named “Stock Ka-Ching (SG)” was also created using adobe photoshop and illustrator software to help simulate an actual feel of a web page. Next, we will dive into details on each of the features in the web page.

Time Taken	Result Found	Positive Percentile	Negative Percentile
0.04887223243713379 s	38	41.7 %	58.3 %

Close up of Search Engine

The figure above shows a simple search engine whereby the search results would be filtered accordingly based on the user’s search query. In this case, the user searched for the company name “3M”. There are three buttons which the enter button would trigger the user search query, the ‘Prev 10’ button would display the previous 10 search results, and the ‘Next 10’ button would display the next 10 search results. To enhance the visual effects, when the user hover over the buttons, the color of the button would also be highlighted accordingly. The results that are retrieved would then be displayed and organized into a precise table format for better delivery of the information to our targeted audience.

The table in the figure above also depicts four main information which are Time Taken, Result Found, Positive Percentile and Negative Percentile and their main purpose is as follows:

1. **Time Taken:** Time taken to search specified results based on user’s query.
2. **Result Found:** Total number of results produced based on user’s query.
3. **Positive Percentile:** Total Positive percentage of the results found based on Polarity.

4. **Negative Percentile:** Total Negative percentage of the results found based on Polarity.

Search:

Elon musk Elon music Elon muck Elon musky

Page: 1

Search Suggestion

As shown above, a search suggestion has also been implemented such that when a user key in a particular search query (e.g., in this case, Elon muskck), the search suggestion bubble(s) would pop up to suggest similar queries that are available.






<div> <div>Subjectivity</div> <div> Not Subjective - 0 Subjective - 1 </div> </div> <div> <div>Polarity</div> <div> Neutral - 0 Negative - 1 Positive - 2 </div> </div>				
Index	Company Details	Company Text	Sentimental Analysis	
			Subjectivity	Polarity
1	<div>Name: 3M Company</div> <div>Code: MMM</div> <div>Date/Time: Tue Mar 09 08:55:02 +0000 2021</div>	3M Innovative Company of the year ! **DENTAL ADVISOR** https://t.co/hafrggtwp8	1	2
	<div>Name: 3M Company</div> <div>Code:</div>	A brilliant story that exemplifies working at 3M.		

Home.html (Partial of the Homepage, Bottom Portion)

Above shows the bottom portion of the web page. Whereby it displays all the results based on the user search query. In this case, as shown previously in an image of the close up search engine, the user searched for the company name “3M” and the results are as shown above. The results are also displayed in a well-organized table in rows and columns, with coloured circles and rounded boxes to create a focus on the information. The main components are Index, Company Details, Company Text, and Sentimental Analysis. Sentimental Analysis would be categorized into two sub-component which is Subjectivity and Polarity.

The 4 main components and their purpose is as follows:

1. **Index:** Provides an index number to the searched record.
2. **Company Details:** The company details are Company Name, Code, and the data/time of the record.
3. **Company Text:** The company text which refers to the information of a tweet.

Subjectivity	 Not Subjective - 0  Subjective - 1
Polarity	 Neutral - 0  Negative - 1  Positive - 2

Legend for Sentimental Analysis

4. **Sentimental Analysis:** Display Subjectivity and Polarity information, As shown above is a legend displayed on top of the table that contains the search records, which allows the user to understand the meaning of each circle and value.
 - 4.1. **Subjectivity:** Display the subjectivity of the record based on 0 being not subjective and 1 being subjective and the background colour of the circle as shown previously in the table that contains the search records, under sentimental analysis – Subjectivity will change accordingly based on the binary number.
 - 4.2. **Polarity:** Display the polarity of the record based on 0 being neutral, 1 being negative and 2 being positive and the background colour of the circle as shown previously in the table that contains the search records, under sentiment analysis – Polarity will change according based on the integer number.

Webpage Footer

To end with the introduction of the user interfaces in the web page, at the bottom of the web page, a footer is created to simulate a real web page.

2. Write five queries, get their results, and measure the speed of the querying

Search: Enter

Prev 10 Next 10
Page: 1

Time Taken	Result Found	Positive Percentile	Negative Percentile
0.03390693664550781 s	16772	79.9%	20.1%

Search: AMZN Enter

Prev 10 Next 10
Page: 1

Time Taken	Result Found	Positive Percentile	Negative Percentile
0.05185961723327637 s	50	80.6%	19.4%

Search: Elon Musk Enter

Prev 10 Next 10
Page: 1

Time Taken	Result Found	Positive Percentile	Negative Percentile
0.04189443588256836 s	5	50.0%	50.0%

Search: Financial Enter

Prev 10 Next 10
Page: 1

Time Taken	Result Found	Positive Percentile	Negative Percentile
0.07931065559387207 s	586	87.2%	12.8%

Search: Abbott Laboratories Enter

Prev 10 Next 10
Page: 1

Time Taken	Result Found	Positive Percentile	Negative Percentile
0.033910274505615234 s	62	68.3%	31.7%

As shown above from the screenshot, five random queries were selected which includes displaying all records by leaving the search box empty, 'AMZN', 'Elon Musk', 'Financial', and 'Abbott Laboratories'. Time taken which is the speed of the query also differs accordingly based on the complexity of each query and also the number of records found.

Question 3

Explore some innovations for enhancing the indexing and ranking. Explaining why they are important to solve specific problems

Further preprocessing

Dataset

After crawling, the text may contain some hashtag (#), user (@) and URL (https://) as shown in Figure below.

```
{
  "text": "One of our own, Jeanine Celeste Pang of @OldNavy, shares her thoughts on growing up as a first-generation American-
  "timestamp": "Mon Mar 08 21:05:13 +0000 2021",
  "code": "GPS",
  "company": "Gap Inc.",
  "subjectivity": 1,
  "polarity": 2
},
```

For every tweet, as shown in Figure below, we are using regex to identify the hidden special character and URL and store it into an array.

```
String after Decode:
One of our own, Jeanine Celeste Pang of @OldNavy, shares her thoughts on growing up as a first-generation American-born Chinese and on how best to
advocate for our AAPI communities in the face of increasing hate crimes. https://t.co/pV5xu5vYXa #StopAAPIHate #StopAsianHate

URL Detected:
['https://t.co/pV5xu5vYXa']

Special Character:
['@OldNavy', '#StopAAPIHate', '#StopAsianHate']

Things to Remove:
['-', 'https://t.co/pV5xu5vYXa', '@OldNavy', '#StopAAPIHate', '#StopAsianHate']

Final Text:
One of our own, Jeanine Celeste Pang of , shares her thoughts on growing up as a firstgeneration Americanborn Chinese and on how best to advocate
for our AAPI communities in the face of increasing hate crimes.
```

After that we use the remove function to remove it from the original text and produce a new text (query) field while keeping the original text for retrace purposes. We will keep the encoded emoticon as it will be automatically decoded in the web user interface.

Furthermore, we created a new field timestamp, which is converted from the date and time of the tweet posted. The type of the timestamp will be in integer.

The final json file after cleaning will be shown in Figure below.

```
{
  "text": "One of our own, Jeanine Celeste Pang of @OldNavy, shares her thoughts on growing up as a first-generation American-born Chinese and on how best t
  "code": "GPS",
  "company": "Gap Inc.",
  "subjectivity": 1,
  "polarity": 2,
  "date_time": "Mon Mar 08 21:05:13 +0000 2021",
  "timestamp": 1615208713.0,
  "query": "One of our own, Jeanine Celeste Pang of , shares her thoughts on growing up as a firstgeneration Americanborn Chinese and on how best to advocat
},
```

User Input

To optimize the user input query, from the raw input, first we will apply case-folding.

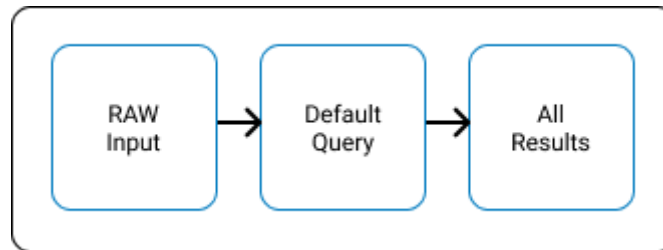
A common strategy is to do case-folding by reducing all letters to lowercase and It will also help on a web search engine. For example, it will allow instances of Amazon at the beginning of a sentence to match with a query of amazon.

Next, we will remove all the stop-words. Removing stop-words can potentially help to improve the search engine performance for fast and relevant retrieval of data from the database.

Next, Spell Correction is also utilized based on Peter Norvig's spell-corrector. Word statistics are utilized to find the most probable candidates.

The above technique is implemented to transform the raw input to optimize the query processing.

Problems



Our default query is directly query tweets content which is referring to our “text” field’s value in our SOLR database. Below is one of the entries listed in the SOLR entries.

Given a SOLR query itself has already been optimized for a query such as tokenization, computing “TF-IDF” score, normalization, and schema similarity scores to generate relevant documents. However, it is still far from perfect. Generally, our default query is decent enough for its intended use. There are still a few problems that can be improved upon. We have identified the problems and tune its query configuration such that it fits best in our context. The results shown below will be selective from our generated predefined test cases.

Handle Code or Company Queries

S/N	Query Syntax	Rank	Result of Company Document	Time Taken(s)
1	text:BA	1	SWK	0.0109925270080566
		2	BA	
		3	BKNG	
		4	BA	
		5	BA	
		6	BA	
		7	SWK	

		8	BA	
		9	BA	
		10	MNST	
2	text:payc	No Result		0.0099947452545166
3	text:waste management inc	1	WM	0.00999116897583007
		2	WM	
		3	WM	
		4	WM	
		5	WM	
		6	WM	
		7	WM	
		8	LYB	
		9	KR	
		10	CLX	

As shown in the 3 result table above, there are irrelevant entries amongst them which are highlighted in **yellow**. The reason for such results is because the query is only interested to compute the given field. Thus, we have to optimize the queries that also consider other fields as well.

Handle Typo Mistakes Queries

S/N	Query Syntax	Expected Query	Time Taken
-----	--------------	----------------	------------

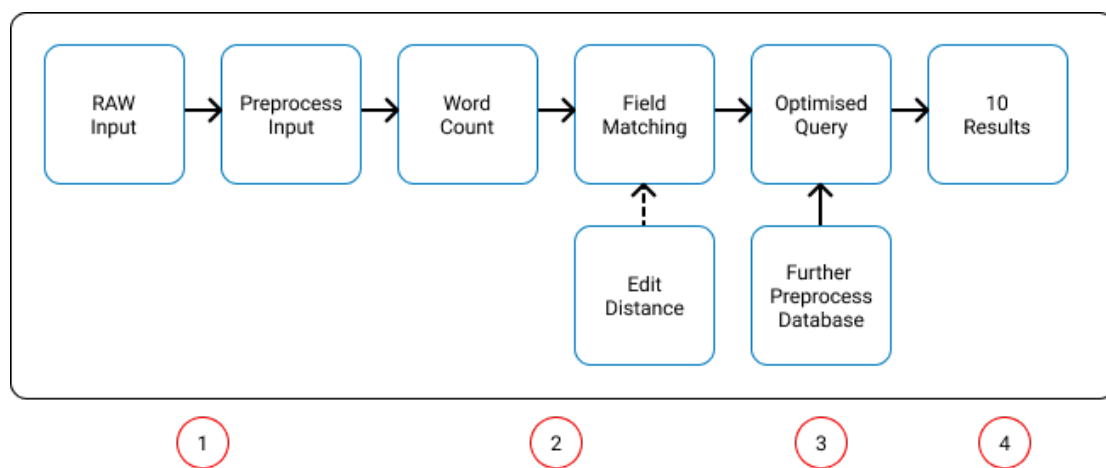
1	text:wamart	walmart	0.009996
2	text:singupora	singapore	0.009995
3	text:meme stoack	meme stock	0.010993

As you can see based on the above top 10 results. It is not able to query properly due to a minor typo mistake. Thus, has resulted in less fruitful or no results return.

Solutions

Handle Code or Company Queries

Optimize Query Flow



1. The RAW inputs are preprocessed to retrieve necessary tokens in a standardized manner as mentioned earlier.
2. Based on the number of tokens to decide which field of interest to query since the characteristic of the 'code' field is a maximum of 1 token, and the 'company' field has a max of 3 tokens. Different fields have different decision requirements. We utilize a double quotation to do an exact match "<query text> " to see if there exists any document(s) return. We start checking from the lowest number of unique terms in the field (code) up to the highest number of unique terms in the field (query), we return the query field name if a match is

found. This helps to improve our query computation as it has fewer terms for matching.

- Code:
 - i. If the number of tokens is one, if not continue the next field of interest.
 - ii. Query the “code” field for an exact match.
 - iii. Check if the documents retrieved is more than zero.
 - iv. If the document(s) is more than zero return, if not continue to the next field of interest.
- Company:
 - i. If the number of tokens is six, if not continue the next field of interest.
 - ii. Query the “code” field for an exact match or within the tolerance value based on edit distance (Levenshtein) version 1 python script¹. The tolerance value is linearly proportional to the number of characters. For example for a 25% tolerance value.

S/N	Characters count	Edit Distance Tolerance Values
1	5	0 and 1
2	7	0 to 2
3	10	0 to 3

- iii. Check if the documents retrieved is more than zero.
 - iv. If the document(s) is more than zero return if not continue to the next field of interest.
- Query:
 - i. No requirement
3. The matched field name will then be used to optimize the query result given the further preprocessing datasets.
 4. Instead of returning all the results by default, we only get 10 of the result at the time based on the given page value. For example, page 2 will return the 10th to 19th index of the documentation assuming the index starts from zero.

¹ Levenshtein distance python script:
https://en.wikibooks.org/wiki/Algorithm_Implementation/Strings/Levenshtein_distance

Enhanced Result

S/N	Query Syntax	Rank	Result of Company Document	Time Taken(s)	Query Speed Comparison with default (100%)
1	code:"ba"	1	BA	0.0139923095703125	127.29%
		2	BA		
		3	BA		
		4	BA		
		5	BA		
		6	BA		
		7	BA		
		8	BA		
		9	BA		
		10	BA		
2	code:"payc"	1	PAYC	0.0139920711517333	139.99%
		2	PAYC		
		3	PAYC		
		4	PAYC		
		5	PAYC		

		6	PAYC		
		7	PAYC		
		8	PAYC		
		9	PAYC		
		10	PAYC		
3	company:waste management inc*	1	WM	0.0159895420074462	160.04%
		2	WM		
		3	WM		
		4	WM		
		5	WM		
		6	WM		
		7	WM		
		8	WM		
		9	WM		
		10	WM		

The result has shown it has been optimized where all the results are reflected correctly based on the given company or the code.

Limitation

1. Longer computation time for better accurate results.
2. Sensitive to character differences since there is no reference to predict the likelihood of an entity's words.

Handle Typo Mistakes Queries

Libraries Used

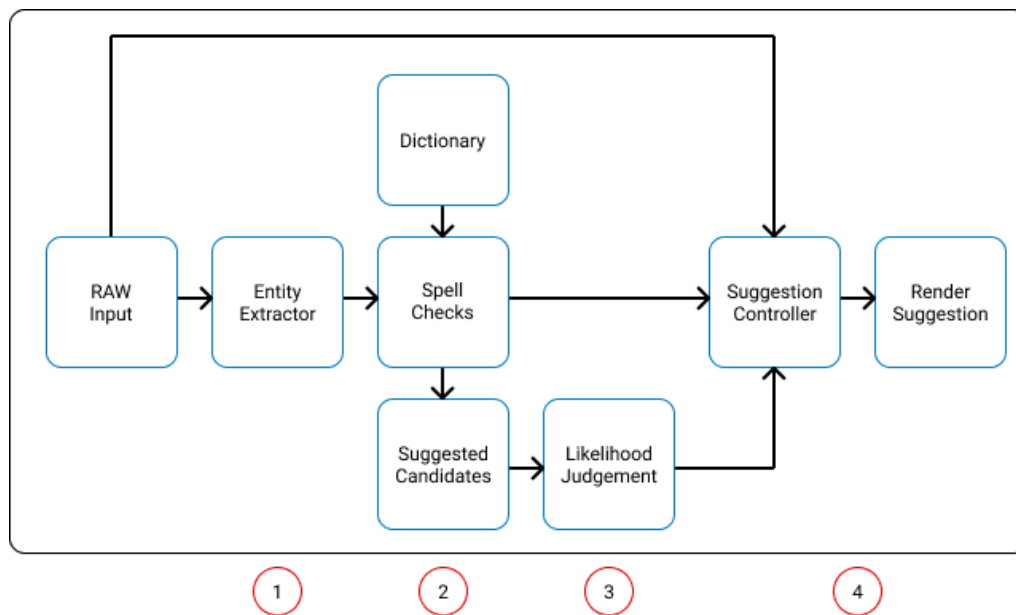
spellchecker

This library adapts Peter Norvig's algorithm and Levenshtein distance algorithm to check for spelling errors effectively². In our context, we are only focusing on the English spelling error.

spacy and pycountry

These libraries focus on removing entities that are considered the noise of spell checkers such as our names, countries or company names as well.

Optimize Query Flow



1. The raw input is parsed into entity extractors, this extractor will remove all types of names such as a name of a person, a name of a company, a name of a country, etc. These names are considered noise of the typo process. Thus, we decide to extract them out. The order of checks is important for efficiency query, the order is arranged based on the likelihood the user will key in this type of entity into the search bar. We assume the user should type the name accurately.

² Peter Novig. Spell corrector algorithm:
<https://norvig.com/spell-correct.html>

2. The remainder input will be separated into words and append into a list. The list will then be traversed by a spell checker with references from its dictionary. We use the library spell checker to evaluate. The spell checker will then return a list of suggested candidates if there exist spelling errors amongst the lists.
3. The suggested candidate's word will be selected based on Ratcliff and Obershelp Gestalt Pattern Matching³ also known as Ratcliff/Obershelp Pattern Recognition, it is a string-matching algorithm for determining the similarity of two strings. It will select the up to four best candidates from the suggested candidates.
4. If there is any error from spell check, it will update the RAW inputs with the best candidates from step 3 and return the suggested result for rendering to the UI.

Enhanced Result

S/N	Query Syntax	Suggestions	Time Taken	Query Speed Comparison with default (100%)
1	query:wamart	walmart	0.043972	439.89%
2	query:singupora	singapore	0.711755	7120.95%
3	query:meme stoack	meme stoick, meme stock, meme stack	0.074952	681.85%

Limitation

1. The typo correction is only subject to a single word. Thus, the suggestion can work better in single word queries as compared to phrases or sentences query.
2. There is a likelihood judgment does not include candidates with severe typo errors.
3. Inconsistent increase of computational time depend on the complexity of the query

³ Ratcliff/Obershelp Pattern Recognition algorithm:
https://en.wikipedia.org/wiki/Gestalt_Pattern_Matching

Conclusion

There is no 1-way ticket for query optimization, there is always a trade-off between speed and accuracy, we have to find the 'sweet spot' through trial and errors to get the optimized result.

Classification

To perform sentiment analysis on the tweets, we took a machine learning based approach with supervised learning. We trained a few models with various configurations on tweets collated from various kaggle datasets⁴. We explored multiclass and multilabel classification to classify tweets based on subjectivity and polarity.

Question 4

1. Motivate the choice of your classification approach in relation with the state of the art

There are many models available to perform text classification and out of those, Bidirectional Encoder Representations from Transformers (BERT) is one of the state of the art models. Many variants of the BERT model exist where each model was designed for more specific problems. The HuggingFace library provides pre-trained BERT based models for transfer learning. We trained BERT (Base, Uncased), BERT (Large, Uncased), Twitter-RoBERTa and ALBERT initially to assess the initial performance of the models. We then chose BERT (Base) and Twitter-RoBERTa out of those for this project as they had a good trade-off between accuracy and training time.

1.1. BERT (Base, Uncased)

This is a transformers model, with about 110 million parameters, pre-trained on a large corpus of uncased data with an automatic generation of inputs and labels in a self-supervised fashion. BERT was trained to randomly mask 15% of the words in the input sentences and predict the masked words allowing it to learn an inner representation of the English language. Given how BERT was trained, for our purpose, we only had to train a classifier using the features extracted by the pre-trained BERT layers.

⁴ Kaggle Datasets: <https://www.kaggle.com/shashank1558/preprocessed-twitter-tweets?select=processedPositive.csv>, <https://www.kaggle.com/crowdflower/twitter-airline-sentiment>, <https://www.kaggle.com/ankurzing/sentiment-analysis-for-financial-news>

1.2. BERT (Large, Uncased)

This variant of BERT is similar to BERT (Base, Uncased) but with many more hidden layers hence having about 340 million parameters. This model can be used with much larger datasets to achieve a higher accuracy as the model is very complex. However, due to the complexity of the model, it takes a much longer time to train on the same amount of data than BERT (Base, Uncased), which is why we did not proceed forward with this model.

1.3. Twitter-RoBERTa

This model is a Robustly Optimised BERT (RoBERTa) model pre-trained on approximately 58 million tweets to classify them as positive, neutral or negative. RoBERTa is built on BERT and trained with larger mini-batches, higher learning rates and with minor tweaks to the tokenizer allowing it to achieve a higher test accuracy in classification tasks. Moreover, since Twitter-RoBERTa is a RoBERTa model pre-trained on tweets, it can be finetuned on our dataset with minimal effort.

1.4. ALBERT

This variant of BERT was developed by Google and Toyota to be a lite version of BERT while achieving a higher accuracy with about 9 times fewer parameters than BERT (Base, Uncased). ALBERT reduced the number of parameters in BERT by sharing all parameters across all layers and improved accuracy by adopting a different training approach with a different scope from BERT. However, the pre-trained weights of this model did not seem to play well with our dataset and it seemed to require a lot more data to be finetuned, which is why we did not proceed forward with this model.

2. Discuss whether you had to preprocess data and why

Labelled Tweets From Kaggle		Crawled Tweets	
Train (75%)	Validation (25%)	Unlabelled (90%)	Manually Labelled (10%)
17,519	5,840	15,102	1,678

The table above shows the distribution of our train, validation and test dataset along with the data to be labelled eventually with the best model.

The tweets crawled included tweets by users and tweets by singular entities where tweets by users were generally noisy and unstructured while tweets by singular entities were cleaner and structured. Therefore, we performed preprocessing to clean the tweets and normalise them. However, the BERT based models that we used, are accompanied by their respective tokenizers which perform extensive data cleaning and preprocessing before tokenization. Hence, preprocessing of the data was not necessary but we still performed some basic preprocessing and evaluated the impact on the accuracy during the ablation study.

	A	B	C	D	E	F
1	text	timestamp	code	company	subjectivity	polarity
2	PepsiCo Inc (PEP	Tue Mar 02 22:52	PEP	PepsiCo Inc.	0	0
3	Teleflex v. Contin	Mon Mar 08 23:2	TFX	Teleflex	0	0
4	Researchers at @	Tue Mar 09 23:01	LLY	Lilly (Eli) & Co.	0	0
5	United Airlines hol	Thu Mar 04 02:31	UAL	United Airlines Ho	1	2
6	At Thermo Fisher	Wed Mar 10 01:4	TMO	Thermo Fisher Sc	0	0
7	\$SGTM 3 Stocks	Mon Mar 08 18:4	CBRE	CBRE Group	0	0
8	vornado realty tru	Thu Mar 04 05:22	VNO	Vornado Realty T	0	0
9	Are you intereste	Fri Mar 05 12:52	DOW	Dow Inc.	0	0
10	"To solidify our fo	Tue Mar 09 21:2	WDC	Western Digital	0	0
11	Of Rs 3,227 crore	Wed Mar 10 00:5	CMS	CMS Energy	0	0
12	Get #NRG to incr	Sat Mar 06 11:36	NRG	NRG Energy	0	0
13	Packaging Corp. (Wed Mar 10 02:2	NWS	News Corp	0	0
14	@Jawanza Article	Wed Mar 10 04:3	UHS	Universal Health S	1	2
15	\$AIG:New Insider	Mon Mar 08 01:0	AIG	American Internat	0	0
16	China wants the E	Mon Mar 08 23:1	BA	Boeing Company	1	1
17	In times of crisis,	Mon Mar 08 18:5	MMM	3M Company	1	2
18	Duke Energy, the	Tue Mar 09 19:25	DUK	Duke Energy	0	0
19	@Mad_Dad2020	Wed Mar 10 00:5	TMUS	T-Mobile US	0	0
20	Amniocentesis Ne	Wed Mar 10 04:2	MDT	Medtronic plc	0	0

Sample Tweets from Test Dataset

Tweet Sentiment	Subjectivity	Polarity
Neutral	0	0
Negative	1	1
Positive		2

Labelling Logic

The figure above shows some sample tweets from our manually labelled test dataset. Every tweet was labelled according to the table above.

The tweets were preprocessed using ekphrasis⁵, a library that uses English Wikipedia and English tweets to perform tasks such as tokenization, word normalisation, word segmentation and spell correction.

Normalisation	
Text	Processed Text
http://www.twitter.com	<url>
hello@twitter.com	<email>
90%	<percent>
\$50.42	<money>
+65 66556655	<phone>
@johndoe	<user>
5:00PM	<time>
12/12/20	<date>

⁵ Ekphrasis GitHub: <https://github.com/cbaziotis/ekphrasis>

87	<number>
:)	<happy>

Annotation	
Text	Processed Text
#sell	<hashtag> sell </hashtag>
BUY NOW	buy <allcaps> now <allcaps>
suuuuuucks	sucks <elongated>
to the moon!!!	to the moon ! <repeated>
sleep on this	sleep <emphasis> on this
dump this sh*t	dump this sh*t <censored>

Unpacking	
Text	Processed Text
#stocksgoup	<hashtag> stocks go up </hashtag>
can't	can not

The tables above illustrate, with examples, the normalisation, annotation and unpacking performed on the tweets using ekphrasis. Emojis were not removed or converted to English text as the vocabulary of the BERT models include emojis.

3. Build an evaluation dataset by manually labelling 10% of the collected data (at least 1,000 records) with an inter-annotator agreement of at least 80%

Three members of our group performed manual labelling on 10% of the data, 1,678 tweets. To measure the inter-annotator agreement, the Fleiss' kappa metric is used

instead of Cohen's kappa as Cohen's kappa measures the agreement between two annotators only.

The definition of Fleiss' kappa is as follows:

Let N be the total number of tweets.

Let n be the total number of labels assigned per tweet.

Let k be the number of labels available.

The tweets are indexed by $i = 1, \dots, N$ and the labels are indexed by $j = 1, \dots, k$.

Let n_{ij} represent the number of annotators who assign the i -th tweet to the j -th label.

$$\kappa = \frac{\bar{P} - \bar{P}_e}{1 - \bar{P}_e}$$

$$p_j = \frac{1}{Nn} \sum_{i=1}^N n_{ij}, \quad 1 = \sum_{j=1}^k p_j$$

$$P_i = \frac{1}{n(n-1)} \sum_{j=1}^k n_{ij}(n_{ij} - 1)$$

$$= \frac{1}{n(n-1)} \sum_{j=1}^k (n_{ij}^2 - n_{ij})$$

$$= \frac{1}{n(n-1)} \left[\left(\sum_{j=1}^k n_{ij}^2 \right) - (n) \right]$$

$$\bar{P} = \frac{1}{N} \sum_{i=1}^N P_i$$

$$= \frac{1}{Nn(n-1)} \left(\sum_{i=1}^N \sum_{j=1}^k n_{ij}^2 - Nn \right)$$

$$\bar{P}_e = \sum_{j=1}^k p_j^2$$

n_{ij}	Objective	Negative	Positive	P_i
1	3	0	0	1.0
2	3	0	0	1.0
3	3	0	0	1.0
4	0	0	3	1.0
5	3	0	0	1.0
...
1674	3	0	0	1.0
1675	3	0	0	1.0
1676	0	3	0	1.0
1677	0	0	3	1.0
1678	3	0	0	1.0
p_j	0.601	0.122	0.277	

Fleiss' kappa was calculated using the definition above and the intermediate results are shown in the table above. The kappa value obtained is **0.833** which is above 0.8 and hence the labels can be considered to have a good inter-annotator agreement.

4. Provide evaluation metrics such as precision, recall, and F-measure and discuss results

BERT (Base, Uncased)			
	Polarity		
	Neutral	Negative	Positive
Precision	0.766	0.543	0.678
Recall	0.851	0.556	0.482
F1-Score	0.807	0.549	0.570
Accuracy	0.721		

Twitter-RoBERTa			
	Polarity		
	Neutral	Negative	Positive
Precision	0.775	0.637	0.623
Recall	0.817	0.599	0.561
F1-Score	0.795	0.618	0.591
Accuracy	0.723		

Twitter-RoBERTa performs better than BERT in terms of polarity accuracy but BERT classifies neutral tweets slightly better than Twitter-RoBERTa as it has a higher f1-score.

5. Discuss performance metrics, e.g., records classified per second, and scalability of the system

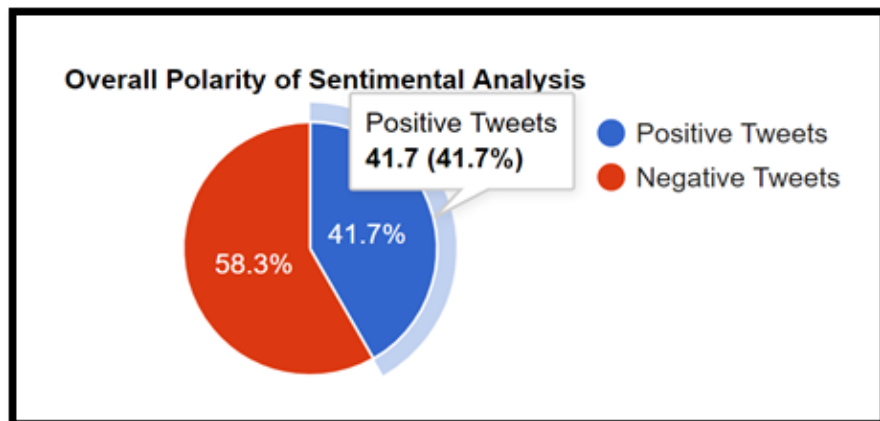
Description	Value
Crawl Time per Tweet	149.73ms
Tweets Crawled per Second	~300
Classification Time per Tweet	9.53ms
Tweets Classified per Second	~105
Model Training Time per Epoch	43.80s
Overall Model Training Time	438s

Since the model has already been trained on a relatively large dataset of tweets, retraining will not be necessary in the near future. However, if over time there is a large shift in the Twitter vocabulary, such as newer generations of Twitter users tweeting with newer slang and abbreviations, the model will have to be finetuned on the newer data crawled from Twitter to maintain its accuracy. In that case, training the model would only take about 438s on TPU given that the amount of training data remains about the same.

If newer batches of twitter data are crawled and have to be classified, the model will be able to classify about 105 tweets per second which translates to classifying 100,000 tweets in about 16 minutes or less.

Since our system is static, crawling, training and classification are considered as offline batch tasks and do not affect the system's live performance in any way. Therefore, the system is highly scalable where ad hoc updates can be scheduled to update the database.

6. A simple UI for visualizing classified data would be a bonus (but not compulsory)



Overall Polarity of Sentimental Analysis

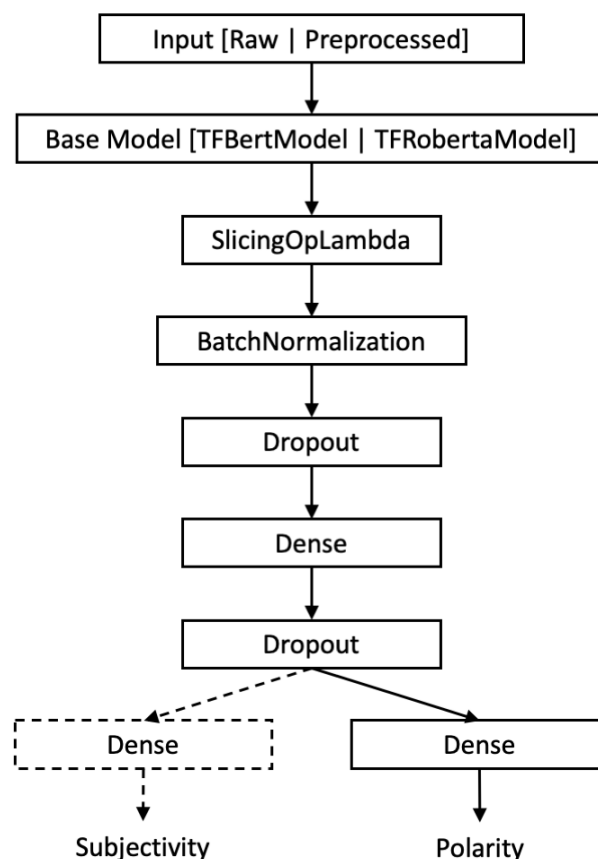
Our visualization will focus based on the Overall Polarity of Sentimental Analysis. As shown above, a Pie Chart is created using JavaScript to provide a simple , elegant and easy-to-understand overview of the Polarity of Sentimental Analysis. In this case, based on the user search query as shown previously in the close up image of the search engine, the positive percentile and the negative percentile would be retrieved to create and visually display a dynamic Pie Chart to provide effective communication aids for the audience. When the user hover over the chart, the user would be able to see the overall percentile of the positive and negative tweets respectively.

Question 5

1. Explore some innovations for enhancing classification. Explain why they are important to solve specific problems, illustrated with examples.

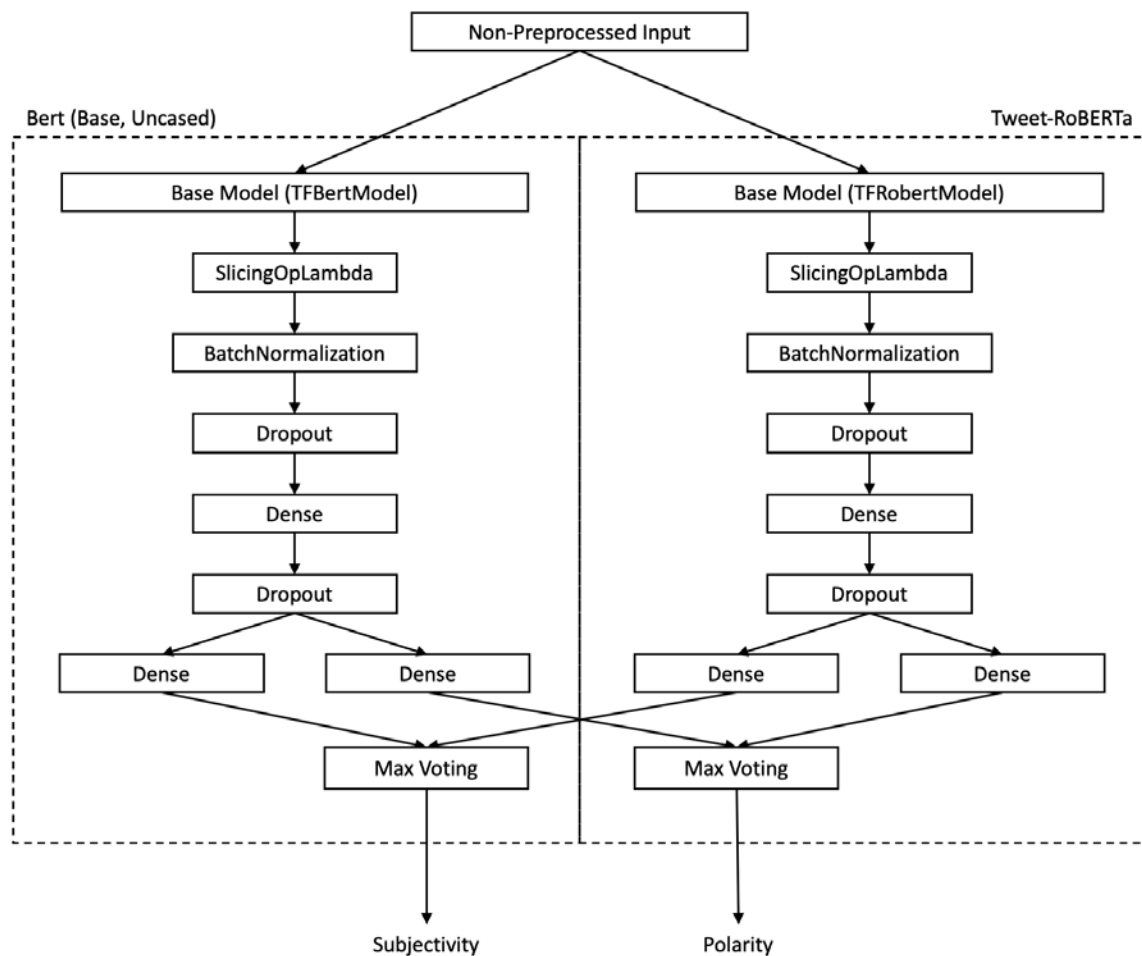
As with any machine learning task, we faced the common problems of overfitting and a jerky accuracy and loss curve. We resolved overfitting by adding dropout layers with a carefully tuned rate of dropout and a batch normalisation layer to provide some regularisation, to allow the model to generalise better. We smoothened the model's accuracy and loss curves by reducing the learning rate and increasing the batch size.

Furthermore, we attempted to improve the classification accuracy by including text preprocessing, multitask classification of subjectivity and polarity and ensemble classification by max voting. We performed an ablation study to measure the impact of the aforementioned techniques on the classification accuracy.



Architecture of Model Variants

The image above shows the architecture of the different model variants. The models were constructed and trained with the tensorflow library on Google Colab TPUs.



Architecture of Ensemble Model

The image above shows the architecture of the ensemble model which is a combination of BERT (Multitask) and Tweet-RoBERTa (Multitask).

BERT			
	Polarity		
	Neutral	Negative	Positive
Precision	0.720	0.545	0.759
Recall	0.897	0.644	0.247
F1-Score	0.799	0.690	0.373
Accuracy	0.697		

BERT (Multitask)					
	Subjectivity		Polarity		
	Objective	Subjective	Neutral	Negative	Positive
Precision	0.772	0.700	0.766	0.543	0.678
Recall	0.836	0.609	0.851	0.556	0.492
F1-Score	0.803	0.651	0.807	0.549	0.570
Accuracy	0.748		0.721		

BERT (Preprocessed)			
	Polarity		
	Neutral	Negative	Positive
Precision	0.793	0.534	0.605
Recall	0.799	0.616	0.547
F1-Score	0.796	0.572	0.575
Accuracy	0.71		

BERT (Preprocessed, Multitask)					
	Subjectivity		Polarity		
	Objective	Subjective	Neutral	Negative	Positive
Precision	0.732	0.719	0.713	0.515	0.769
Recall	0.879	0.488	0.915	0.560	0.215
F1-Score	0.799	0.582	0.801	0.537	0.336
Accuracy	0.728		0.689		

Twitter-RoBERTa			
	Polarity		
	Neutral	Negative	Positive
Precision	0.775	0.637	0.623
Recall	0.817	0.599	0.561
F1-Score	0.795	0.618	0.591
Accuracy	0.723		

Twitter-RoBERTa (Multitask)					
	Subjectivity		Polarity		
	Objective	Subjective	Neutral	Negative	Positive
Precision	0.779	0.672	0.776	0.623	0.638
Recall	0.803	0.639	0.813	0.664	0.545
F1-Score	0.791	0.655	0.794	0.643	0.588
Accuracy	0.740		0.725		

Twitter-RoBERTa (Preprocessed)			
	Polarity		
	Neutral	Negative	Positive
Precision	0.751	0.540	0.489
Recall	0.750	0.530	0.494
F1-Score	0.750	0.535	0.491
Accuracy	0.656		

Twitter-RoBERTa (Preprocessed, Multitask)					
	Subjectivity		Polarity		
	Objective	Subjective	Neutral	Negative	Positive
Precision	0.690	0.717	0.690	0.707	0.644
Recall	0.913	0.351	0.912	0.267	0.351
F1-Score	0.786	0.472	0.786	0.388	0.454
Accuracy	0.695		0.684		

Max Voting Ensemble Model Twitter-RoBERTa (Multitask) & BERT (Multitask)					
	Subjectivity		Polarity		
	Objective	Subjective	Neutral	Negative	Positive
Precision	0.778	0.699	0.782	0.591	0.675
Recall	0.831	0.624	0.835	0.644	0.538
F1-Score	0.804	0.660	0.808	0.616	0.599
Accuracy	0.751		0.734		

The tables above show the metrics, precision, recall, f1-score, accuracy, of the individual models from the ablation study conducted. The last table shows the metrics of the ensemble model of the best BERT model and the best Tweet-RoBERTa model from the ablation study.

Comparison of Metrics								
	Subjectivity				Polarity			
	Prec.	Rec.	F1.	Acc.	Prec.	Rec.	F1.	Acc.
BERT	NA				0.707	0.697	0.662	0.697
BERT (Multitask)	0.744	0.748	0.744	0.748	0.715	0.721	0.713	0.721
BERT (Preprocessed)	NA				0.711	0.710	0.710	0.710
BERT (Multitask, Preprocessed)	0.727	0.728	0.715	0.728	0.702	0.689	0.647	0.689
Tweet-RoBERTa	NA				0.718	0.723	0.720	0.723
Tweet-RoBERTa (Multitask)	0.738	0.740	0.738	0.740	0.721	0.725	0.721	0.725
Tweet-RoBERTa (Preprocessed)	NA				0.656	0.656	0.656	0.656
Tweet-RoBERTa (Multitask, Preprocessed)	0.701	0.695	0.664	0.695	0.680	0.684	0.649	0.684
Max Voting Ensemble	0.748	0.751	0.748	0.751	0.730	0.734	0.729	0.734

The table above shows a comparison of the weighted average of the precisions, recalls and f1-scores along with the accuracies of all the models. From the table above, it can be observed that the max voting ensemble model achieved the highest accuracy for both subjectivity and polarity classification. In fact, this result is not a surprise as the

ensemble model combines the best prediction capabilities of the two models, BERT and RoBERTa, when the final prediction probabilities are compared in a maximum voting manner. Therefore the final class predictions per label are based on the most confident predictions out of both of the models.

Conclusion

In this project, we built an information retrieval system capable of searching for tweets related to stocks. The tweets were crawled using the Twitter API using some search terms. Data manipulation, selection and indexing were performed to ensure top documents remain relevant from Solr software. Multitask classification of subjectivity and polarity was done using an ensemble model of BERT and Tweet-RoBERTa by max voting. The model was trained on a twitter dataset that was put together from multiple Kaggle Datasets and tested on a manually labelled test set. This classification model was selected as the best out of all the other models we trained based on the metrics, precision, recall, f1-score and accuracy.

All in all, this has definitely helped us to understand the basic concepts and building blocks for information retrieval systems.

Appendix A - Submission Links

Presentation: <https://youtu.be/DMnRSuTWyyA>

Crawled Text Data, Queries, Classifications:

<https://drive.google.com/file/d/1YULTTRDpk9vsBmkT4p0qqzdeyrNDR1Gg/view?usp=sharing>

Source Codes:

<https://drive.google.com/file/d/1L1JhiTw2AfmmQvMyKHkQVMShyOGng5pd/view?usp=sharing>

Appendix B - How to use the program 101

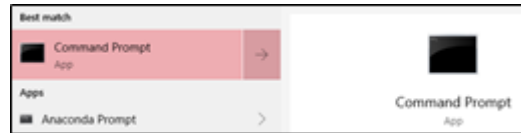
Prerequisite:

1. Setup SOLR and Django properly.
 - SOLR Installation Guide Link: https://solr.apache.org/guide/7_0/installing-solr.html.
 - Django Installation Guide Link: <https://docs.djangoproject.com/en/3.1/topics/install/>.
2. Setup VSCode IDE properly.
 - VSCode Latest Package Link: <https://code.visualstudio.com/>.
3. Setup Python properly.
 - Python Latest Package Link: <https://www.python.org/downloads/>. (Recommended: Ver 3.8.2 and above)
4. Install external packages.
 - “nltk” : <https://pypi.org/project/nltk/>.
 - “pysolr” : <https://pypi.org/project/pysolr/>.
 - “autocomplete” : <https://pypi.org/project/autocomplete/>.
 - “pyspellchecker” : <https://pypi.org/project/pyspellchecker/>.
 - “ekphrasis” : <https://pypi.org/project/ekphrasis/>.
 - “spacy” : <https://pypi.org/project/spacy/>.
 - “pycountry” : <https://pypi.org/project/pycountry/>.
5. Downloaded and extracted our project’s source code file called “test_django.zip” into your local directory workspace.

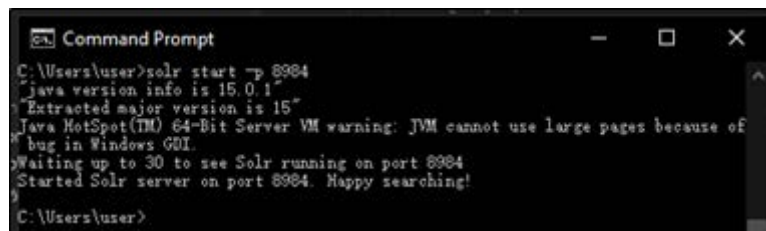
Setup Hosting:

SOLR (Backend - Database)

1. Open your command prompt (CMD) window.



2. Key in and enter the following script in the command line: "solr start -p 8984".
3. Once entered you should be able to see this output as shown in the figure below. This indicates your SOLR server has hosted locally successfully.

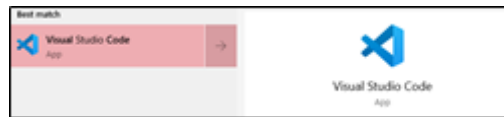
A screenshot of a Windows Command Prompt window. The title bar reads 'Command Prompt'. The command prompt shows the following text:

```
C:\Users\user>solr start -p 8984
java version info is 15.0.1
Extracted major version is 15
Java HotSpot(TM) 64-Bit Server VM warning: JVM cannot use large pages because of
bug in Windows GUI.
Waiting up to 30 to see Solr running on port 8984
Started Solr server on port 8984. Happy searching!
C:\Users\user>
```

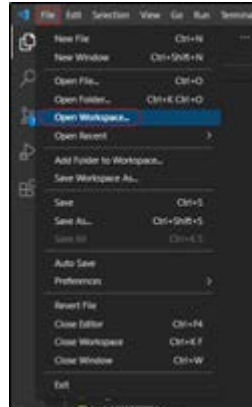
4. Key in and enter the following script at your browser to access the SOLR GUI of the hosting details: "http://localhost:8984".
5. (Optional) Key in and enter the following script in the command line to end hosting the server: "solr stop -p 8984"

Django (Frontend)

1. Open your VSCode IDE



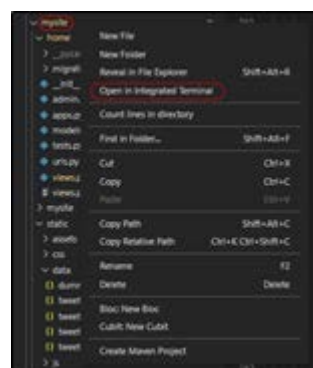
2. Open the workspace of our folder by hovering to the top left of the IDE window as shown in the figure below.



3. Select your extracted directory as shown below.



4. At VSCode's explorer section, open the integrated terminal as shown below.



5. Should be able to see a pop-up "bash" terminal as shown below.



6. Key in and enter the following script: "python manage.py runserver".

7. Key in and enter the local IP address and port number to access the local web page as follows: "http://127.0.0.1:8000/"
8. (Optional) At terminal key "CTRL+C" to end hosting the server.

Misc Configuration:

The configuration helps us to separate our concerns more easily. Thus, improve the workflow. Our configuration code can be located at "mysite/static/home/views.py".

```
60 # !---Configuration-----
61 addDataOp = False      # A toggle to add static crawled data into SOLR in JSON format
62 debug = False          # This remove unnecessary debugging statement in compiled code to maintain efficient query
63 fileName = 'tweets_v4' # Select the file name use to read the data locally at mysite/static/data directory
64 testMode = False       # This generate a csv that contain all the test cases based on a given fixed inputs for testing and analysis uses
65 optimiseQuery = True    # Default - (Query the tweet SOLR) , Optimise - (Query with other feature/configuration SOLR has to offer)
66 # !-----
```