



Final Project - Spring 24'
[SPEC TOPIC DATA SCIENCE 16:964:694:01]

Final Report

Group-26

Amrutha Karuturi - ak2508

Glory Ekbote - ge144

Ruth Sharon - rsr172

Kunche Suneeth - ks1983

Github URL : [Twitter Search Engine](#)

Github Usernames:

Gleek231997

amrutha2508

suneethkunche

ruth1445

Abstract

This project focuses on analysing a Twitter dataset comprising JSON documents containing diverse information such as user details, tweet content, media attachments, and timestamps. The dataset is split into user information stored in a relational database (PostgreSQL) and tweet information in a non-relational database (MongoDB). After data processing and loading, we aim to develop a search application enabling retrieval based on various parameters including most used hashtags, tweets with highest retweet counts, and those with the most quote counts. Filtering options include user ID, username, hashtag, tweet content, and date range. Additionally, we plan to implement cache mechanisms to swiftly retrieve critical data, integrating error handling and file management for efficient cache management.

Introduction

The Twitter Search application aims to construct and develop a search engine capable of retrieving pertinent tweets or users in response to user queries. It manages vast volumes of Twitter data to distil relevant information and features a search interface facilitating effortless data querying.

The project's scope encompasses designing and implementing a robust data model for storing user and tweet information, optimising data storage and retrieval through suitable datastores, and crafting a search interface enabling users to search tweets based on keywords, hashtags, or usernames. Additionally, the project entails fine-tuning the search process for enhanced speed and efficiency and integrating caching mechanisms to boost performance.

Ultimately, this initiative aims to deliver a potent tool for searching and analysing Twitter data, catering to various applications such as social media marketing, sentiment analysis, and academic research.

Dataset

This project utilises a dataset of tweets retrieved from the Twitter API specifically designed for a Twitter Search application. The data is rich in detail, encompassing information about individual tweets themselves (text content, timestamp, user details) and related metadata like hashtags and mentions. Additionally, it provides details about the users who posted the tweets, including usernames, follower/following counts, and even profile descriptions. The dataset's substantial size, boasting over 88,000 users, 50,000 tweets, and 61,000 retweets, necessitates the development of efficient storage and search methods to unlock the valuable insights it holds.

Persisted Data Model and Datastores

The Twitter Search application prioritises fast search functionality. To achieve this, we implemented indexing techniques for quick data retrieval. This optimization comes at a cost, however, with increased memory usage, a more complex system design, and potentially higher costs. To counteract this, our data models were meticulously designed to strike a balance between storage efficiency and retrieval speed for the massive Twitter dataset. These models hold essential fields for both users and tweets. User models include information like user ID, location, and account creation date. Tweet models encompass the tweet content itself, relevant hashtags, and the tweet type. Type, here, indicates whether it is a tweet or retweet. If type = 1, it is a retweet and if type=2, it is a tweet.

Processing Tweets for Storing in Datastores

As shown in *Fig 1.1* the process starts with reading lines of the JSON text file. The data was then parsed into a JSON document split into three parts: tweets, retweets, and users. The tweets were then formatted and stored in a MongoDB collection named “tweet collection” and the retweets were formatted and stored in a separate MongoDB collection named “retweet collection”. User data was formatted and stored in a PostgreSQL database. After successfully loading the JSON data to respective databases, IDs inserted were retained in order to check for duplicates when entering again. Moreover, indexing strategies were utilised for quick and efficient data retrieval from the respective datastores.

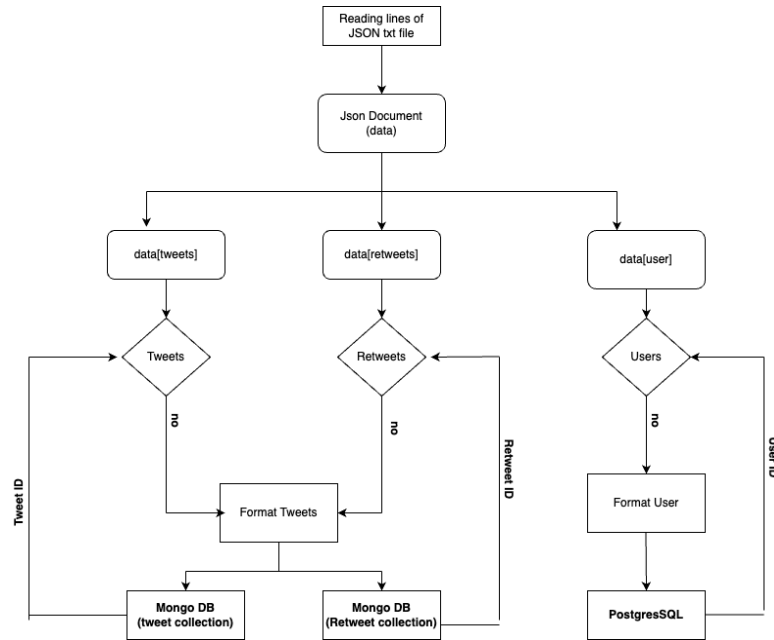


Fig 1.1 Data Processing and Loading Architecture

Cache

Efficient Information Retrieval with Caching

The Cache class serves as the foundation for efficient data retrieval within the application. It initialises a dictionary specifically designed for caching, containing various cache types like "string," "hashtag," and "user." Each cache type acts as a separate dictionary with a predefined maximum capacity of 30 elements. This ensures efficient storage and prevents the cache from becoming overloaded. Additionally, the Cache class provides methods for practical management. It allows you to load cached data from a file for retrieval purposes and even offers a clear cache function to reset it to an empty state. Fig 1.2 shows our vision for caching.

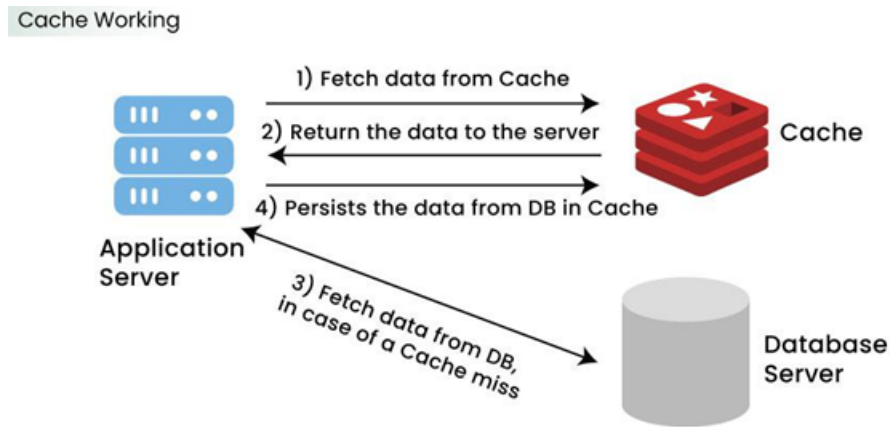


Fig 1.2 Cache Architecture for Search Application

Streamlined Tweet Filtering with CHECKPOINT

The CHECKPOINT function is a crucial component for filtering tweets based on their creation time. It streamlines this process by converting the provided start and end times into timestamps, allowing for flexibility in handling unspecified values (represented as infinity). The function then iterates through each tweet, meticulously checking if its creation timestamp falls within the designated time range. Tweets that meet this criterion are meticulously added to a designated filtered data list. Finally, the function delivers a dictionary containing the filtered tweets along with a timestamp that reflects the specific filtering operation performed. This timestamp helps maintain a record of when the filtering occurred.

Search Application Design

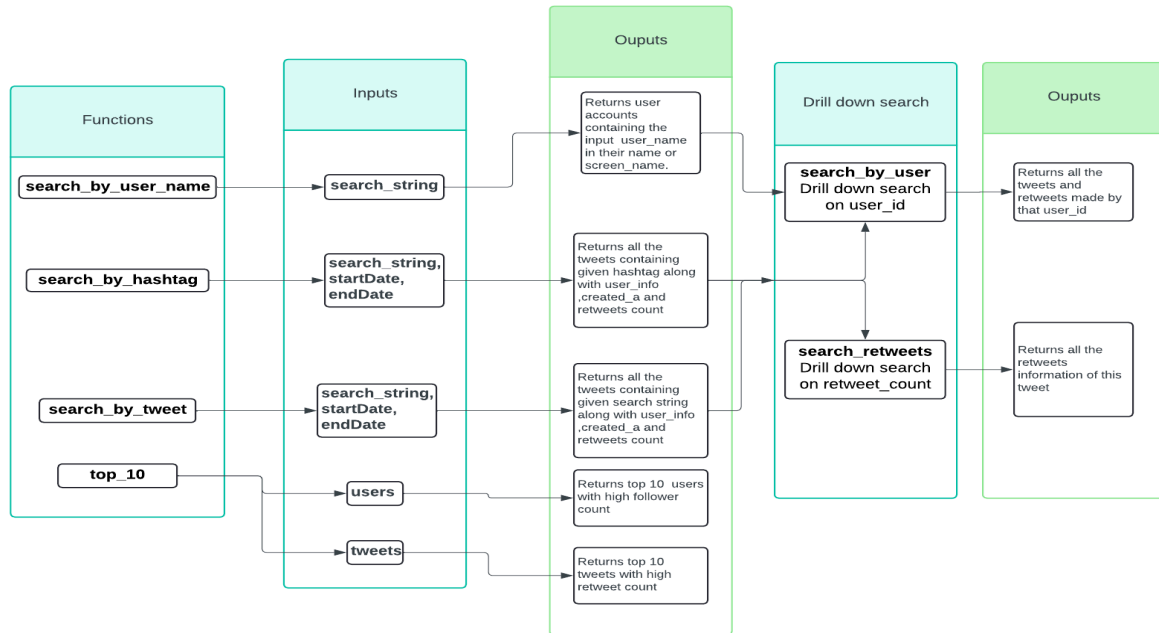


Fig 1.3 Search functions in our search application

Figure 1.3 gives an overview of all the search functions provided for users, their inputs and outputs and what further drill down searches can be performed. The following functionalities are provided:

Fig 1.4 User search options : users, hashtags, string(for tweets), get top_10 metrics

As shown in Fig 1.4, users are provided with 4 search options: User , Hashtag , String along with which the user provides a keyword to be searched, and Top Metrics.

1. For User option : it returns all the user accounts that have name or screen_name like that of the keyword.
2. For Hashtag option : it returns all the tweets containing the keyword as hashtag(sorted by retweet count).

3. For string option : it returns all the tweets that contain the keyword(according to their relevance score(which is a textscore provided in the metadata of MongoDB)).
4. For Top Metrics : it returns top 10 users (sorted by follower count) / top 10 tweets (sorted by retweet count).

Search Query Performance :

The search query performance is improved by indexing databases as shown in Fig 1.5. It improves search query performance by reducing the number of documents examined during searches, consequently decreasing overall execution time.

<i>Indexes on tweet db</i>					<i>Indexes in retweets db</i>				
Name and Definition	⌵≡	Type	⌵≡	Size	Name and Definition	⌵≡	Type	⌵≡	Size
> hashtags_1		REGULAR ⓘ		94.2 KB	> user_id_str_1		REGULAR ⓘ		217.1 KB
> text_text		TEXT ⓘ		2.8 MB	> org_tweet_id_1		REGULAR ⓘ		143.4 KB
> user_id_str_1		REGULAR ⓘ		217.1 KB					

<i>Indexes on user db (relational DB)</i>					
Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
id_str	character varying ▾	20		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Fig 1.5 Indexes

Results

Search Twitter Engine

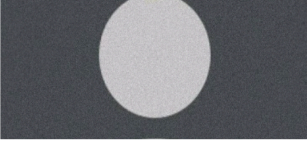
Top Metrics

String Start: End:

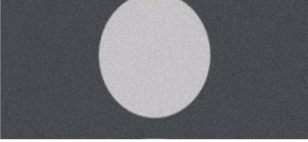
User Name	Twitter Handle	Screen Name	Follower Count	Friend Count	Verified User	Description	Location
Narendra Modi	18839785	narendramodi	55782751	2364	true	Prime Minister of India	India
The New York Times	807095	nytimes	46361159	904	true	News tips? Share them here: http://nyti.ms/2FVHq9v	New York City
Amitabh Bachchan	145125358	SrBachchan	41596464	1833	true	'तुमने इतने पूरा पूरा कर पावर कर छाटा ; वे जो हमपर तुमने कसते हैं हमें जिंदा तो समझते हैं' ~ हरिवंश राय बच्चन	Mumbai, India
Shah Rukh Khan	101311381	iamsrk	40026760	77	true		
PMO India	471741741	PMOIndia	34461808	486	true	Office of the Prime Minister of India	India
Hrithik Roshan	113419517	iHrithik	28170371	90	true	Man on mission- to live the best life possible come what may.	
Arvind Kejriwal	405427035	ArvindKejriwal	18335920	221	true	सब हमारा बराबर हैं, चाहे वो किसी धर्म या जाति के हों। इसे देश भारत बनाना है जहाँ सभी धर्म और जाति के लोगों में भाईचारा और मोहब्बत हो, न कि नफरत और बैर हो।	India
TIME	14293310	TIME	17057740	494	true	Breaking news and current events from around the globe. Hosted by TIME staff.	
detikcom	69183155	detikcom	15927868	28	true	Official Twitter of http://www.detik.com , redaksi@detik.com promosi@detik.com Android: http://detik.com/android iPhone: http://detik.com/iphone	Jakarta, Indonesia
J.K. Rowling	62513246	jk_rowling	14608046	721	true	Writer sometimes known as Robert Galbraith	Scotland
Search Time		2.621173858642578 ms					

Top Followed Users

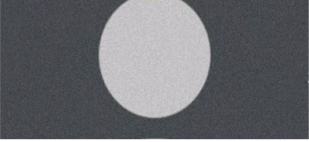
Fig 1.6 UI of the Search Application and Top 10 Users result

HtownBabyG


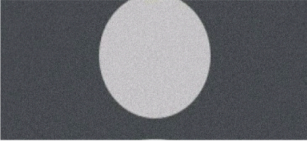
corona virus enters my body The 4 Flintstone gummies I ate in 2005.
<https://t.co/3STfdlQuT>

Khydill


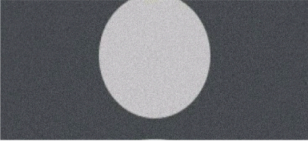
When this Corona shit passes we have to promise each other that we're going to tell our kids that we...

AyeeCarlos


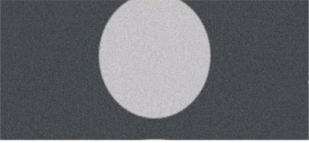
THIS MAN IS A GENIUS he figured out the Corona virus problem 🧐
<https://t.co/EZP7lqTnV>

nichellexnicole


"corona time" 🤔🤔🤔🤔 <https://t.co/iXBMHVcFoY>

SJPeace


This is Dr. Usama Riaz. He spent past weeks screening and treating patients with Corona Virus in Pak...

LyanneSavage


Nobody: Me and the homies on our \$41 corona trip to the Bahamas:
<https://t.co/UOoRRJLGr>

Fig 1.7 Top 10 tweets Display

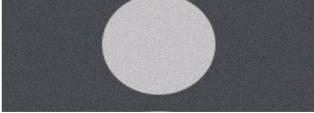
Search Twitter Engine

Top Metrics

String corona
Start: dd/mm/yyyy
End: dd/mm/yyyy
Search

Search Time: 5.040399074554443 ms


2020-03-13 00:43:40



[Author Profile](#)
[Retweets](#)
 Liked: 811062
 Twitter Handle: HtownBabyG

corona virus enters my body The 4 Flintstone gummies I ate in 2005:
<https://t.co/3STf6lQuT>

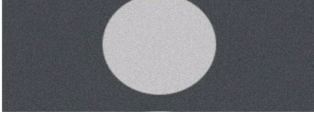
2020-03-18 17:51:18



[Author Profile](#)
[Retweets](#)
 Liked: 764405
 Twitter Handle: Khydiill

When this Corona shit passes we have to promise each other that we're
 going to tell our kids that we...

2020-03-10 17:52:14



[Author Profile](#)
[Retweets](#)
 Liked: 515867
 Twitter Handle: _AyeCarlos_

THIS MAN IS A GENIUS he figured out the Corona virus problem 🤔
<https://t.co/EZP7lqTxV>

Fig 1.8 Searching by string “corona”

Search Twitter Engine

Top Metrics

String corona
Start: dd/mm/yyyy
End: dd/mm/yyyy
Search

Search Time: 0.07828617095947266 ms

Close

Screen Name: Khydiill

Created At: 2020-03-18 17:51:18

ID: 1240334979701395458

Tweet: When this Corona shit passes we have to promise each other that we're going to tell our kids that we survived a zombie apocalypse in 2020

Retweet Count: 181584

Fig 1.9 More details about the author of the tweet

Conclusion

This project has demonstrated success in creating an efficient search engine for managing and retrieving user and tweet data, bolstered by a user-friendly interface. With optimised data models and datastores, the system is capable of handling large data volumes while maintaining rapid response times to search queries.

Through experimentation, caching mechanisms were found to significantly enhance response times, and further optimizations in data modelling and indexing further improved performance. Introducing drill-down search functionalities, such as exploring tweet and user metadata, significantly enriched the user experience.

This project has provided valuable insights into managing extensive datasets, designing effective data structures, and optimising search mechanisms, underscoring the balance between performance and usability. In summary, the Twitter Search application offers a valuable solution for navigating Twitter data and presents opportunities for future enhancements and refinements.

References

- Learning Flask for Python: https://youtu.be/Z1RJmh_OqeA?si=SgmidfXB5myxsPEf
- How to use MongoDB in a Flask Application: <https://www.digitalocean.com/community/tutorials/how-to-use-mongodb-in-a-flask-application>
- System Design: Twitter: <https://medium.com/@karan99/system-design-twitter-793ab06c9355>
- Best 6 Caching Strategies to Boost Your Website's Performance: <https://rabbitloader.com/articles/caching-strategies-to-speed-up-your-website-performance/#:~:text=There%20are%20different%20types%20of,most%20commonly%20used%20caching%20strategies.>

Team Member Contribution

Member	Work Taken Up
Amrutha Karuturi	Relational Database Loading, Database Indexing , Search queries
Glory Ekbote	Data Loading and Processing for Non-Relational Databases, Flask for Data rendering on UI and Integration.
Kunche Suneeth	Cache Management
Ruth Sharon	User Formatting functions and Documentations