Universitat Pompeu Fabra Barcelona

Elena Barrio 252060
Laura Juan 254094
Rocio Gilabert 251217

REPORT PART 4

User Interface and Web Analytics

Github LINK and TAG: IRWA-2024-part-4

User Interface

This last part of the project is based on the implementation of previous parts, since we will work with the processed tweets and the search algorithms that we have developed.
In order to implement an Interface for our search engine that allows users to input queries, search for tweets and visualize some analytics, we have used the startup Flask files for developing a web application that we have been proportioned with.

For the implementation of the first part, the User Interface, we have done several steps to complete what was asked. The first step was to adapt and include all our algorithms in the *algorithms.py* file. The different algorithms are:

1. **TF-IDF Search:** With this method, the documents are returned according to the tf-idf algorithm.
2. **Own Search Method:** This is the method that we designed in the last part of the project. Apart from the tf-idf weights, it also takes into account the number of likes, followers, retweets and if the user is verified or not, as it is an easier way to know the level of popularity of the tweet.
3. **BM25:** With this method, the documents are returned according to the BM25 algorithm that adds the fact of document length normalization.
4. **Word2Vec:** The last method we have used is Word2Vec, where each tweet is represented by an averaged vector of its word embeddings and the cosine similarity between the query and every tweet is computed.

When doing the tweet search at the interface, the user can specify which search method wants to use by selecting it at a dropdown menu.
In the same *algorithms.py* file we can find some auxiliary functions such as `build_terms(line)`,`extract_hashtags(content)`, `get_token_tweets(dic)`, `create_index(dic, clean_tweets)` or `min_max_scaling(tweets, doc_id, common_docs, value)`. They are used at different points in the method search functions.

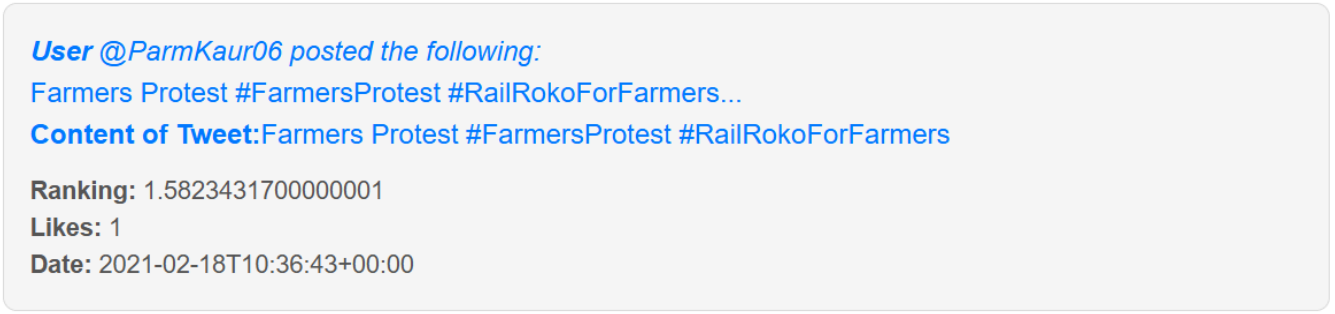
The final output of any search method is the ordered list of relevant documents information and, since we are working with conjunctive queries, no matter the selected method, the documents will always contain all query terms.

Now, adapting the `build_demo_results(corpus: dict, search_id, doc_scores)` function at *search_engine.py* , it receives as input a dictionary of document scores (with document IDs as keys and their relevance scores as values) and a corpus (a

dictionary where each key is a document ID and the value is the document object containing its metadata). The function returns a list composed of *Resultitem* objects, where each represents a document with its attributes and the associated ranking score.

Finally, we would like to mention that, in order to reduce the run time, we have decided to use a pickle file to open different elements that require a lot of time to be generated. Moreover, it would be created each time the user interface is used, something completely senseless.

After the user selects a search method and submits their query, a list of documents is displayed, showing basic information such as the user who posted the tweet, the description, and a related link, between others. Additionally, we have included an option at the top of the page to return to the search tab, allowing the user to perform a new query if desired.



When clicking on the link of each document, it is displayed all the information regarding that tweet which is the following:

- User Id
- Username
- Document title
- Tweet content
- Posted date
- Followers
- Number of likes
- Number of retweets
- If the account is or not verified
- Url: for any URL appearing in the document, we have decided that you could follow the link in order to look at the web page or image related to the tweet.
- Hashtags

On this table, there are also included the option, at the top of the page, to perform a new search, view the current statistics, and access the dashboards.

Universitat Pompeu Fabra Barcelona

Elena Barrio 252060
Laura Juan 254094
Rocio Gilabert 251217

New Query

Stats
Dashboard

User Id: 1362350323797475331 - **Username:** @ParmKaur06
Document title: Farmers Protest #FarmersProtest #RailRokoForFarmers...
Tweet Content: Farmers Protest #FarmersProtest #RailRokoForFarmers

Posted date: 2021-02-18T10:36:43+00:00
Followers: 235
Likes: 1
Retweets: 0
The user account is **not verified**
Url: https://twitter.com/ParmKaur06/status/1362350323797475331
Hashtags: *#FarmersProtest #RailRokoForFarmers*

Web Analytics

For the statistics section, we have collected various types of data to visualize and extract meaningful insights. This data can be particularly useful for web developers to better understand user behavior within our interface. As outlined below, we have chosen to measure these specific metrics because we believe they provide valuable information to improve the web search platform by identifying user preferences. The data we have gathered includes the following:

Session:

- **Time:** We have recorded the exact moment the session was initiated, as well as the duration of the session.
- **User Requests:** We have tracked the user's IP address while using the search engine and counted the number of requests made during the session.
- **Browser/OS:** We have identified the browser and operating system the user is utilizing during the session to access the search engine.
- **Clicks:** We have tallied the total number of clicks the user makes throughout the session.

Document:

- **Visits:** We track how many times each document is accessed and rank them based on the number of visits. Only the top five most visited documents are displayed.

Universitat Pompeu Fabra Barcelona

Elena Barrio 252060
Laura Juan 254094
Rocio Gilabert 251217

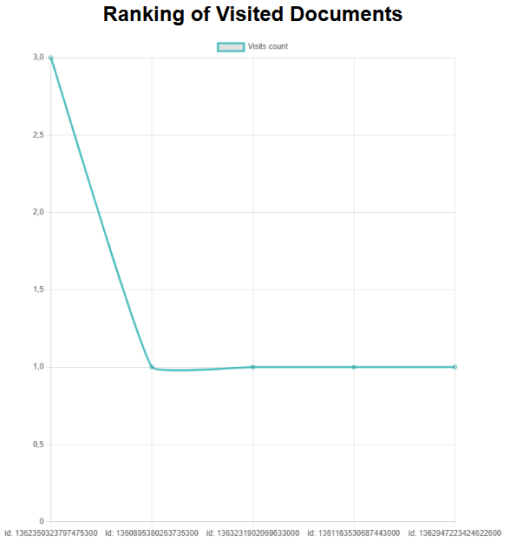
Queries:

- **Most Searched Queries:** We compile a list of all the queries submitted and count how many times each query has been searched. Since a single user can perform multiple searches in a session, we rank all queries by their search frequency and display the top five most searched ones.
- **Average Number of Terms per Query:** We calculate the average number of words in each of the top five queries. Additionally, we perform the same calculation for all queries combined. When computing the average, we consider each query equally, regardless of how many times it has been searched.

As with the other tabs, we have included an option at the top of the page to perform a new search.

Additionally, in this part a Dashboards option is shown, so that the user can move to that page to visualize the data outlined in the Stats tab. This decision was made to simplify the task of the web developer in analyzing and understanding the data generated by user interactions with the interface.

First, we have implemented a **Ranking of Visited Documents dashboard**, which displays the frequency of visits for each document throughout the session. This dashboard shows, in a ranked format, how many times each document has been accessed. Below the graph, we provide a list of the documents displayed in the chart, including their IDs (which are clickable and link to the document's details), titles, and associated queries. This provides a clear and detailed overview of the most visited documents and the context of their usage.



The documents displayed above are the following

(3 visits) — id: 1362350323797475331 — Farmers Protest #FarmersProtest #RailRokoForFarmers **Associated Queries:** farmer protest

(1 visits) — id: 1360895380263735302 — Shame on Indian Government 😡 #FarmersProtest https://t.co/XuLV671Gof **Associated Queries:** indian governement

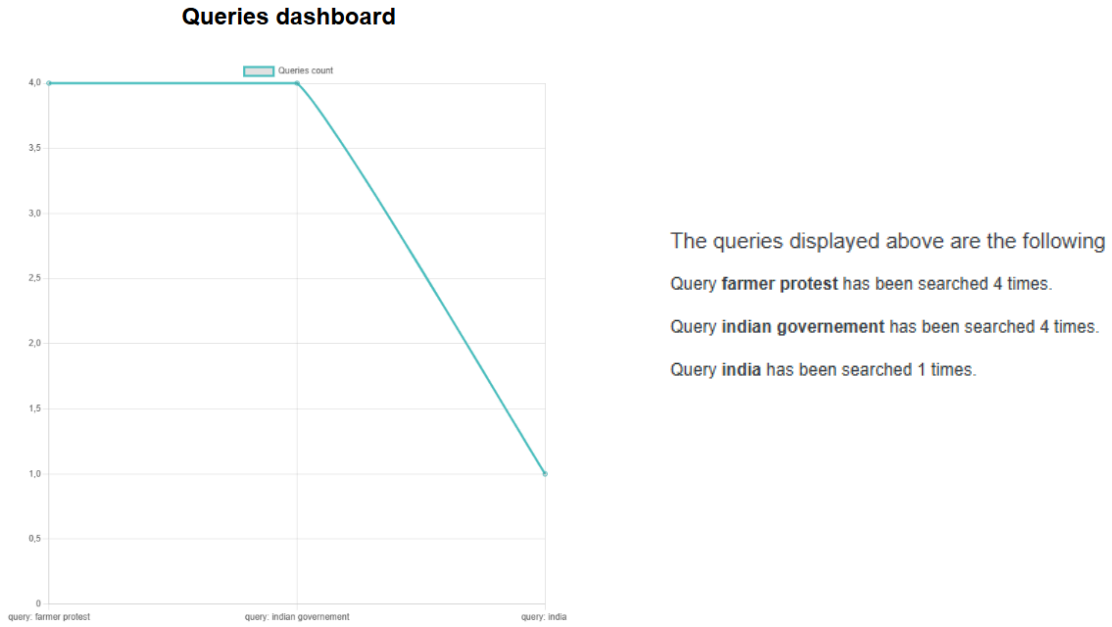
(1 visits) — id: 1363231902069633032 — Shame on Indian government #FarmersProtest https://t.co/khyXi7pNAy **Associated Queries:** indian governement

(1 visits) — id: 1361163530687442949 — So disgusted by Indian government now ! #FarmersProtest #FreeDishaRavi **Associated Queries:** indian governement

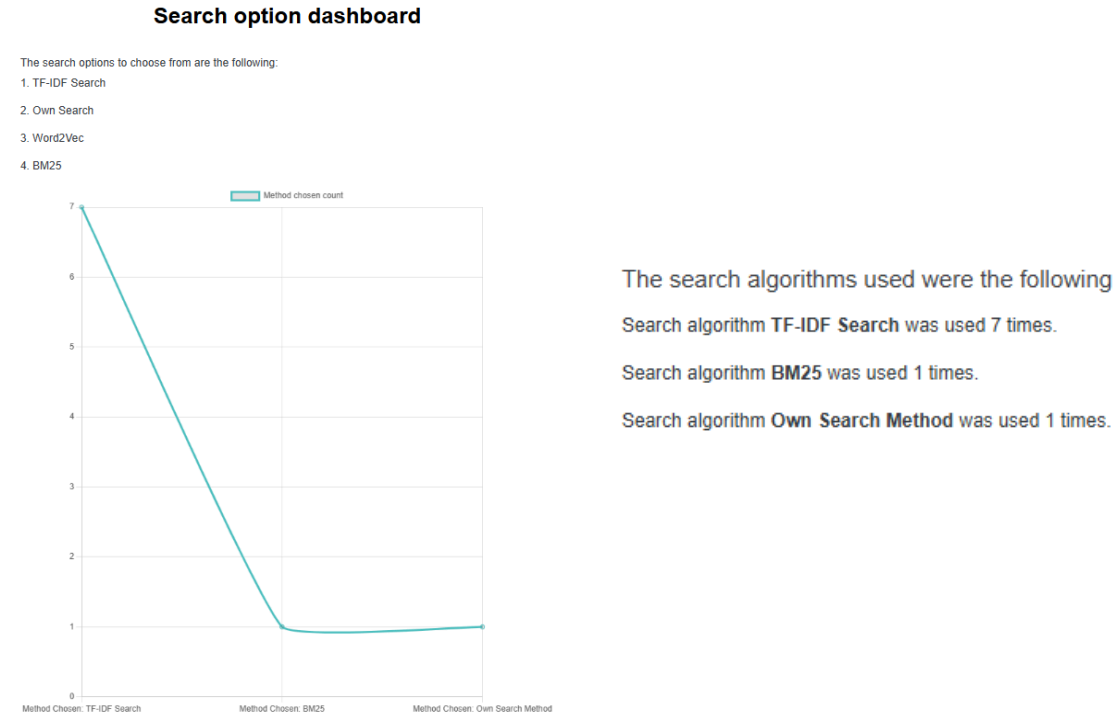
(1 visits) — id: 1362947223424622593 — #FarmersProtest #India https://t.co/NHIwTMQUk4 **Associated Queries:** india

The second dashboard, **Queries Dashboard**, shows the frequency of searches for each query during the session. For every query that has been searched, this dashboard indicates how many times it was explored. Beneath the chart, we include a list summarizing the queries and their respective counts. This dashboard provides an intuitive overview of user

search behavior and is a valuable tool for developers to analyze user preferences and interaction patterns.



The third dashboard, **Search Option Dashboard**, tracks how often each of the four available search algorithms has been used. This metric is particularly useful for developers testing multiple search engines, as it provides insights into which methods users prefer. The dashboard clearly ranks the algorithms based on their usage, helping identify the most favored search options.



Universitat Pompeu Fabra Barcelona

Elena Barrio 252060
Laura Juan 254094
Rocio Gilabert 251217

Finally, the fourth dashboard, **Term Frequency Dashboard**, highlights the word frequency across all queries submitted during the session. It visually represents how often each word has appeared in the searched queries. Below the dashboard, we provide a list of the most frequent query words and their respective counts. This data helps us understand the topics users are most interested in and refine the search engine to better cater to user needs.