Report Part 4: User Interface and Web Analytics

1. INTRODUCTION

The objective of this phase of the project is to develop a web-based search engine interface for retrieving and ranking a collection of tweets. The system will allow users to input search queries through a user-friendly interface and display ranked results using optimized search algorithms and a custom scoring function that incorporates tweet popularity (likes and retweets). The web application, built using Flask, will include functionalities for querying, result ranking, and document exploration.

In addition to the search engine functionality, the project integrates web analytics to track and analyze user interactions with the system. The collected data will provide insights into the search engine's performance and user engagement, which will be visualized through a comprehensive analytics dashboard.

2. USER INTERFACE

We have adapted and extended the initial code from the Toy Search Engine seminar. Our objective was to reuse the previously developed search and ranking functions, integrating them into a functional web application using the Flask framework. The processed data, stored in the processed_tweets.json file, serves as the corpus for performing searches.

<u>Data Loading and Preparation</u>

Tweets are loaded from the JSON file. An inverted index is built to map terms to the tweets where they appear, optimizing the search process. Additionally, tweet lengths are calculated, and TF-IDF values are initialized to prepare for the ranking process.

<u>Ranking with TF-IDF</u>

The rank_tweets_tfidf function implements a ranking algorithm based on cosine similarity between the query and tweet vectors, like in the previous projects parts. The output is a list of doc_id values representing the most relevant tweets, sorted by their scores.

<u>Web Interface with Flask</u>

Home Page: The initial page includes a search box where users can input their queries.

Search Results: Upon submitting a query, terms are tokenized and searched in the inverted index to retrieve relevant tweets. Results are displayed with a title (the first few words of the tweet), a summary (the full text), the date, hashtags, and a link to the document details.

Document Details: Clicking on a result opens a page displaying the tweet's full information, including hashtags, full text, and date.

View analytics button: We are going to explain this button in the following point.

3. WEB ANALYTICS

To implement the web analytics component, we developed a system that captures relevant data about user searches and their interactions with the results. Additionally, we included an analytics dashboard that presents key metrics and statistics on the search engine's usage. The data is stored in memory to simplify the educational implementation, and the system tracks queries, document clicks, dwell time, and user characteristics, among other aspects.

Access to the analytics is provided through a "View Analytics" button located below the search bar. When clicked, users are redirected to the analytics dashboard, where they can view detailed statistics on the application and search engine's performance, facilitating the evaluation of user behavior and improving the search experience.

Thanks to the web app code and the analytics HTML file, we have implemented a dashboard that displays key statistics on the search engine's usage. This panel allows users to view the following:

- Total number of queries performed
- Total number of document clicks
- Most popular documents
- Queries made
- Click data: For each document click, the related document, the corresponding search query, and the user's dwell time on the document before returning to the search results are displayed

4. CONCLUSION

In conclusion, this project has successfully integrated a web-based search engine using Flask, optimized with a ranking algorithm that utilizes TF-IDF to improve the relevance of the results. The implementation of a web analytics system has allowed for the capture of key information regarding user interactions with the search engine, providing valuable data on queries, document clicks, and dwell times. This analysis is crucial for understanding user behavior and enhancing the search experience. Additionally, the analytics dashboard facilitates the visualization of key metrics, contributing to the continuous evaluation of the system. Overall, the project not only improves the efficiency of the search engine but also provides a powerful tool for monitoring and optimizing the system’s performance.

5. CODE LOCATION AND EXECUTION INSTRUCTIONS

The code for this project can be found here:
GITHUB Repository: *https://github.com/davidobrero/IRWA-2024-G102-10*

All related with Part 4 can be found inside the folder IRWA-2024-PART-4.

To execute the project, navigate to the main project folder in the terminal and run the following command: *python web_app.py*

This will allow you to access all the previously explained functions.