Twitter Keyword Search Project Report

Aryan Gupta, Ehsan Sabouni, Roman Velez, Mayank Yadav, Khang Le

EC 504 Spring 2023

I. Team Information

Román G. Vélez-Alicea
BU ID: 83639966
SCC Username: rgva@bu.edu

Aryan Gupta
BU ID: 67933048
SCC Username:gupta368@bu.edu

Khang Le
BU ID: 23595664
SCC Username:khang@bu.edu

Ehsan Sabouni
BU ID: 37945543
SCC Username: esabouni@bu.edu

Mayank Yadav
BU ID: 23633553
SCC Username: mayanky@bu.edu

II. Abstract

This project involves the implementation of an inverted index to efficiently search for specific phrases within a set of text documents. The program takes in input files containing tweets and prompts the user for a keyword search. The program then outputs the tweet IDs that contain the matching term(s) and their similarity score. Finally, it sorts the tweet IDs based on similarity score and outputs the top 10 most similar tweets. The inverted index data structure which is similar to an unordered map allows for fast search and retrieval of relevant documents, making it an effective solution for text search applications. The reason it is called inverted is because of how the words are the keys and the tweet id are the values stored in a list rather than the other way around. This project aims to implement in the language C and python. The results validate the implementation with various input files, and comment on the performance analysis.

III. Instructions for running the code

Attached on Github is a makefile that compiles: inverted_index.h, inverted_index.cpp and main.cpp. There are a total of 4 .txt files that serve as input files named: basic_input.txt,

tweets1.txt, tweets2.txt, and tweets3.txt. The basic_input.txt only contains 3 tweets and it is the example that was given on Github under SuggestedProjects. The remaining ones (tweets1.txt, tweets2.txt and tweets3.txt) are input files that are extracted from the Twitter API and are actual tweets retrieved from the API database.

Here are the steps to download and run our program:

1. Make sure the source files and input files are in the same directory
2. Compile the programs with the makefile:
    a. make -k
3. Run the program
    a. ./output

When the program is running, it will prompt the user for two things. The first one will ask the input file (make sure to put .txt at the end) and the next one is the keyword search. To get the most out of our program, searching for specific phrases based on the input file will give more meaningful outputs. Here are some suggestions:

tweets1.txt:

- "scared of finals"

tweets2.txt:

- "excited for summer"

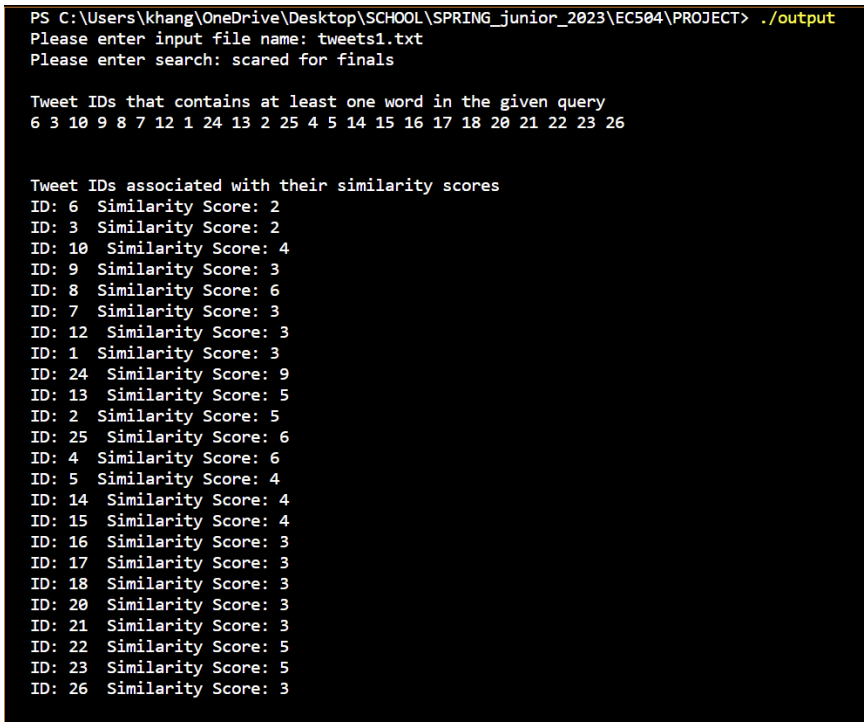
tweets3.txt:

- "we are currently in a recession"

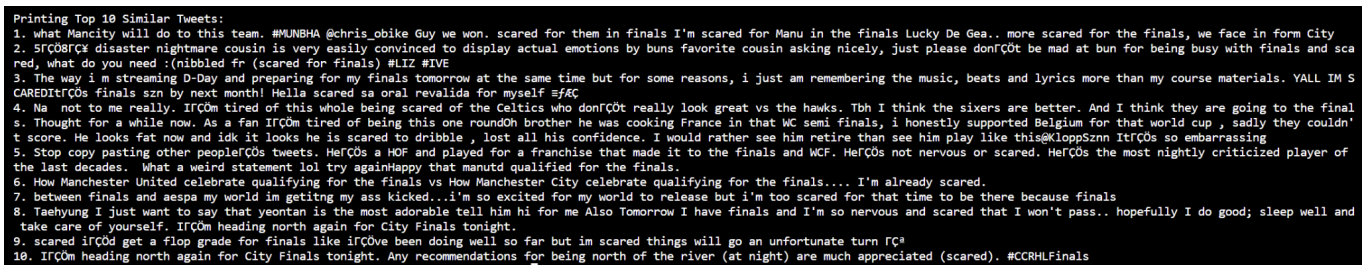
After completing all the steps, the program will first output the tweet ID's that contains at least one matching term with the user's query. The tweet IDs are computed simply by the line number in the input file, since each tweet is designed to be separated by a new line and a line contains the entire tweet. Next, the program will output into a .txt file that has a name format of "*output_<input file name>*" the tweet ID's and their similarity score, which is computed simply by counting the matching terms of the tweet and the user's search. The tweets that do not have any matching terms are not going to get printed. Finally, the program will sort all of the tweet IDs in order of their similarity score, access the tweets based on their ID and output the top 10 most similar tweets.

IV. Results

Figure 1: tweets1.txt as input and “scared for finals” for query



```
PS C:\Users\khang\OneDrive\Desktop\SCHOOL\SPRING_junior_2023\EC504\PROJECT> ./output
Please enter input file name: tweets1.txt
Please enter search: scared for finals

Tweet IDs that contains at least one word in the given query
6 3 10 9 8 7 12 1 24 13 2 25 4 5 14 15 16 17 18 20 21 22 23 26


Tweet IDs associated with their similarity scores
ID: 6  Similarity Score: 2
ID: 3  Similarity Score: 2
ID: 10  Similarity Score: 4
ID: 9  Similarity Score: 3
ID: 8  Similarity Score: 6
ID: 7  Similarity Score: 3
ID: 12  Similarity Score: 3
ID: 1  Similarity Score: 3
ID: 24  Similarity Score: 9
ID: 13  Similarity Score: 5
ID: 2  Similarity Score: 5
ID: 25  Similarity Score: 6
ID: 4  Similarity Score: 6
ID: 5  Similarity Score: 4
ID: 14  Similarity Score: 4
ID: 15  Similarity Score: 4
ID: 16  Similarity Score: 3
ID: 17  Similarity Score: 3
ID: 18  Similarity Score: 3
ID: 20  Similarity Score: 3
ID: 21  Similarity Score: 3
ID: 22  Similarity Score: 5
ID: 23  Similarity Score: 5
ID: 26  Similarity Score: 3
```

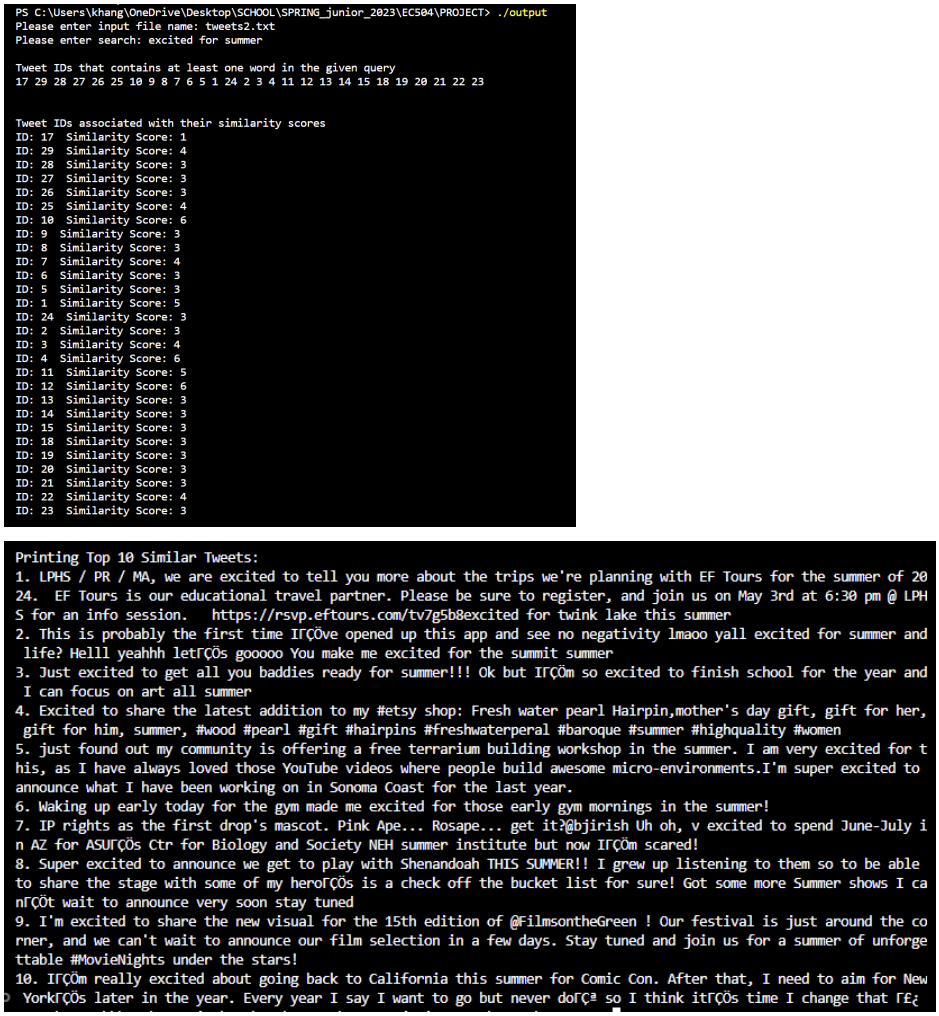


```
Printing Top 10 Similar Tweets:
1. what Mancity will do to this team. #MUNBHA @chris_obike Guy we won. scared for them in finals I'm scared for Manu in the finals Lucky De Gea.. more scared for the finals, we face in form City
2. 5ΓÇÖ8ΓÇ¢ disaster nightmare cousin is very easily convinced to display actual emotions by buns favorite cousin asking nicely, just please donΓÇÖt be mad at bun for being busy with finals and sca
red, what do you need :(nibbled fr (scared for finals) #LIZ #IVE
3. The way i m streaming D-Day and preparing for my finals tomorrow at the same time but for some reasons, i just am remembering the music, beats and lyrics more than my course materials. YALL IM S
CAREDItΓÇÖs finals szn by next month! Hella scared sa oral revalida for myself ≡ƒÆç
4. Na  not to me really. IΓÇÖm tired of this whole being scared of the Celtics who donΓÇÖt really look great vs the hawks. Tbh I think the sixers are better. And I think they are going to the final
s. Thought for a while now. As a fan IΓÇÖm tired of being this one roundOh brother he was cooking France in that WC semi finals, i honestly supported Belgium for that world cup , sadly they couldn'
t score. He looks fat now and idk it looks he is scared to dribble , lost all his confidence. I would rather see him retire than see him play like this@KloppSznn ItΓÇÖs so embarrassing
5. Stop copy pasting other peopleΓÇÖs tweets. HeΓÇÖs a HOF and played for a franchise that made it to the finals and WCF. HeΓÇÖs not nervous or scared. HeΓÇÖs the most nightly criticized player of
the last decades.  What a weird statement lol try againHappy that manutd qualified for the finals.
6. How Manchester United celebrate qualifying for the finals vs How Manchester City celebrate qualifying for the finals.... I'm already scared.
7. between finals and aespa my world im getitng my ass kicked...i'm so excited for my world to release but i'm too scared for that time to be there because finals
8. Taehyung I just want to say that yeontan is the most adorable tell him hi for me Also Tomorrow I have finals and I'm so nervous and scared that I won't pass.. hopefully I do good; sleep well and
 take care of yourself. IΓÇÖm heading north again for City Finals tonight.
9. scared ifΓÇÖd get a flop grade for finals like ifΓÇÖve been doing well so far but im scared things will go an unfortunate turn ΓÇª
10. IΓÇÖm heading north again for City Finals tonight. Any recommendations for being north of the river (at night) are much appreciated (scared). #CCRHLFinals
```

As we can see, the program was able to output the similarity score for all of the tweet IDs that contains at least one matching term with the user’s query: “scared for finals”. Matching the similarity score with the order that the program outputs the top 10 most similar tweets, by comparing the tweet IDs with the tweets in the input file that is provided on Github, they match and are correctly sorted. For instance, we can see tweet ID: 24 matches the tweet on the 24th line

in the tweets.txt file, and that we were able to count that there are 9 instances that the tweet contains a matching term with the user's query: "scared for finals". "*Scared*" appeared 3 times, "*for*" appeared 3 times, and "*finals*" appeared 3 times, resulting in a 3+3+3=9 similarity score.

Figure 2: tweets2.txt as input and "excited for summer" for query



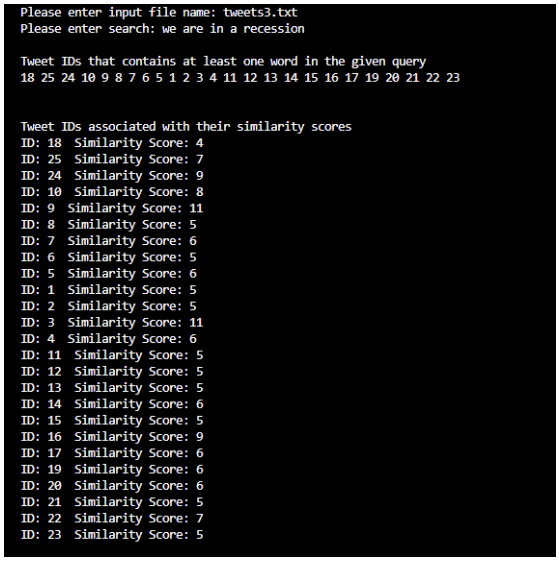
```
PS C:\Users\khang\OneDrive\Desktop\SCHOOL\SPRING_junior_2023\EC504\PROJECT> ./output
Please enter input file name: tweets2.txt
Please enter search: excited for summer

Tweet IDs that contains at least one word in the given query
17 29 28 27 26 25 10 9 8 7 6 5 1 24 2 3 4 11 12 13 14 15 18 19 20 21 22 23


Tweet IDs associated with their similarity scores
ID: 17  Similarity Score: 1
ID: 29  Similarity Score: 4
ID: 28  Similarity Score: 3
ID: 27  Similarity Score: 3
ID: 26  Similarity Score: 3
ID: 25  Similarity Score: 4
ID: 10  Similarity Score: 6
ID: 9  Similarity Score: 3
ID: 8  Similarity Score: 3
ID: 7  Similarity Score: 4
ID: 6  Similarity Score: 3
ID: 5  Similarity Score: 3
ID: 1  Similarity Score: 5
ID: 24  Similarity Score: 3
ID: 2  Similarity Score: 3
ID: 3  Similarity Score: 4
ID: 4  Similarity Score: 6
ID: 11  Similarity Score: 5
ID: 12  Similarity Score: 6
ID: 13  Similarity Score: 3
ID: 14  Similarity Score: 3
ID: 15  Similarity Score: 3
ID: 18  Similarity Score: 3
ID: 19  Similarity Score: 3
ID: 20  Similarity Score: 3
ID: 21  Similarity Score: 3
ID: 22  Similarity Score: 4
ID: 23  Similarity Score: 3
```

```
Printing Top 10 Similar Tweets:
1. LPHS / PR / MA, we are excited to tell you more about the trips we're planning with EF Tours for the summer of 20
24.  EF Tours is our educational travel partner. Please be sure to register, and join us on May 3rd at 6:30 pm @ LPH
S for an info session.   https://rsvp.eftours.com/tv7g5b8excited for twink lake this summer
2. This is probably the first time IΓÇÖve opened up this app and see no negativity lmaoo yall excited for summer and
 life? Helll yeahhh letΓÇÖs gooooo You make me excited for the summit summer
3. Just excited to get all you baddies ready for summer!!! Ok but IΓÇÖm so excited to finish school for the year and
 I can focus on art all summer
4. Excited to share the latest addition to my #etsy shop: Fresh water pearl Hairpin,mother's day gift, gift for her,
 gift for him, summer, #wood #pearl #gift #hairpins #freshwaterperal #baroque #summer #highquality #women
5. just found out my community is offering a free terrarium building workshop in the summer. I am very excited for t
his, as I have always loved those YouTube videos where people build awesome micro-environments.I'm super excited to
announce what I have been working on in Sonoma Coast for the last year.
6. Waking up early today for the gym made me excited for those early gym mornings in the summer!
7. IP rights as the first drop's mascot. Pink Ape... Rosape... get it?@bjirish Uh oh, v excited to spend June-July i
n AZ for ASUΓÇÖs Ctr for Biology and Society NEH summer institute but now IΓÇÖm scared!
8. Super excited to announce we get to play with Shenandoah THIS SUMMER!! I grew up listening to them so to be able
to share the stage with some of my heroΓÇÖs is a check off the bucket list for sure! Got some more Summer shows I ca
nΓÇÖt wait to announce very soon stay tuned
9. I'm excited to share the new visual for the 15th edition of @FilmsontheGreen ! Our festival is just around the co
rner, and we can't wait to announce our film selection in a few days. Stay tuned and join us for a summer of unforge
ttable #MovieNights under the stars!
10. IΓÇÖm really excited about going back to California this summer for Comic Con. After that, I need to aim for New
 YorkΓÇÖs later in the year. Every year I say I want to go but never doΓÇª so I think itΓÇÖs time I change that Γ£¿
```

Similar to the first output, tweet ID of 12 is outputted as the most similar tweet, which makes sense since it has a similarity score of 6 which is the highest out of all the tweets. We can verify the similarity score by noticing that "excited" shows up in the tweet 1 time, the word "for" shows up 3 times and "summer" pops up 2 times. This results in a similarity score of 1+3+2=6 which is what the program outputted. We can also see the sorting works because the second most similar tweet is tweet ID 4, which holds a similarity score of 6 as well. The 10th most similar tweet has a similarity score of 4, which makes sense because there are 3 tweets with a score of 6,

2 tweets with a score of 5, and 5 tweets with a score of 4. This ultimately results in the 10th tweet holding a similarity score of 4 which is what we have.

Figure 3: tweets3.txt as input and "we are in a recession" for query



Similar to the previous results, this run on the program consists of the most matches, both in similarity scores as well as number of tweets that match the search. The tweet that has the most matches with the user's search of "we are in a recession" consists of a similarity score of 11. As we see with tweet ID 3, there are 11 words in this tweet that match the words in the user's

search. There were a total of 2 appearances for “we”, 2 appearances for “are”, 3 appearances for “in”, 2 appearances for “a” and 2 appearances for “recession”. This adds up to the similarity score of 2+2+3+2+2=11.

Overall, the performance analysis is incredibly efficient. Since the time complexity for the entire to program is O(m + K^2), m being the number of words found in the input data, while K is the number of tweets that contains at least one matching term with the user’s search. This was computed because O(m) is used to create the inverted index data structure, as the algorithm requires to go through all of the words in the input data file in order to add it into the inverted index. As for O(K^2), this was computed after we have derived the list of tweets that contains at least one matching term with the user’s query, as well as assign a similarity score to each of the tweets using an unordered map which costs O(K) time, we would have to sort the list of tweets based on their similarity score, which has a worst case of O(K^2).

Due to the fact that it really depends on how the program is ran, for example, if the user’s search has no words that are in the input data file, then the runtime would simply be O(m) since K=0, giving us a linear runtime. However, if the user’s search is costly, giving us a K = number of tweets, then depending on the number of tweets vs. number of words in the file, that will affect our runtime as well.

So overall, the most efficient aspect of our inverted index data structure is that the program runtime analysis does not entirely depend on the input data size, but rather, the user’s input of keywords and how relevant it is compared to the input data. This allows the input data size to be incredibly massive and still could give us a linear runtime.

V. General Context and References

‘get_tweets.ipynb’:

The algorithm implemented in the get_tweets function leverages the Twitter API [1] via the Tweepy library [2] to collect tweet data containing a specific keyword. By utilizing the Tweepy Cursor object, the function efficiently navigates through the paginated results of the API, fetching tweets in batches and iterating over them.

During the iteration, the algorithm filters out retweets and non-English tweets to focus on original, English-language content. For each tweet that meets these criteria, the function extracts and stores the tweet text. The process continues until the desired number of tweets, as specified by the max_tweets variable, has been collected.

Once the collection process is complete, the function organizes the extracted data into a pandas DataFrame [3] for further analysis or manipulation. Additionally, the collected tweet data is written to a text file in a formatted manner for use in our inverted index program.

An additional code cell in the Python notebook is used purely for data generation. Calling the function get_tweets("some keyword(s)") returns 100-1000 most recent tweets containing these keywords, which can then be used by calling the inverted_index executable with the generated tweet text file as an argument.

'main.cpp':

The Inverted Index is implemented by associating all the words found in the input data file with a list of tweet IDs that contains it. It then creates an inverted list that contains all the tweets that have at least one matching term with the user's search, done by accessing all the words from the user's search within the inverted index, each time appending the documents that contain the word. While doing so, it also increments the similarity score for each tweet by having a separate unordered_map that associates the tweet ID with its similarity score (initialized to 0), and as it goes through the inverted list, it increments the similarity score unordered_map by 1. Finally, it sorts the tweets based on their similarity score, and outputs the top 10 most similar tweets.

VI. References:

[1] Twitter Inc., "Twitter API," Twitter Developer Platform, 2021. [Online]. Available: https://developer.twitter.com/en/docs/twitter-api. [Accessed: May. 05, 2023].

[2] J. Roesslein, "Tweepy," GitHub Repository, 2021. [Online]. Available: https://github.com/tweepy/tweepy. [Accessed: May. 05, 2023].

[3] W. McKinney, "pandas: a Foundational Python Library for Data Analysis and Statistics," Python for High Performance and Scientific Computing, 2011. [Online]. Available: https://pandas.pydata.org/. [Accessed: May. 05, 2023].

[4] "Inverted Index." *GeeksforGeeks*, GeeksforGeeks, 4 May 2023, https://www.geeksforgeeks.org/inverted-index/.