Dev Design Doc for MicroForce - Twitter Tweet Recommendations based on Topics

Tyler Wong - - tylercw3@illinois.edu
Rohan Khanna - rohank2@illinois.edu
Cesia Bulnes - cbulnes2@illinois.edu

Introduction on Topic:

We are currently looking at topic modeling tweets and recommending the tweets or users that are most similar in a given tweet. We are looking at using a ranking algorithm such as Okapi BM25 learned in class for ranking tweets by ranking the most similar tweets given a tweet by a user, the category of the tweets that it is related to, the top ranked users in said category, and the top categories/topics in a given sample set. For example, had we borrowed tweets from November 6th,2020, the top topic would have been politics because of the USA elections. Versus November 26 probably having TheGrammys nominations as one of the most trending topics. It will be interesting to recommend other similar tweets/users to a category presented by a single query. We will be ranking and observing the results with a sample set of 2000 users for the time in this project, and observing the past 7 days of tweets by each said user.

Currently:
By the end of the week, we hope to finish the database setup. We decided to use SQLite to avoid any errors on setting up our computers. Simplifying the database will allow us to work on the algorithms needed in the last two weeks of the class.

In addition, we're also working on gathering all the necessary data we need from the Twitter API and importing that data into the database. We have all created twitter developer API and have started planning what we each need to do come the last two weeks of class. We plan on using multiple different entities that the Twitter API provides, such as user data and Tweet annotations and entities.

Concerns:
As of now the concern is having a good ranking model. We want to make sure that the ranking model is correct in terms of how close tweets are to each other. It's imperative that our team take some time to validate returned ranks to make sure there are no discrepancies. For example a query that is exactly like a tweet in our database should have such a tweet ranked as 1 versus one that is highly similar but not exactly the same.

Work to do:

1. Ranking algorithm to rank the query or tweet that the user has typed, displaying the ranked tweets in commonality.

    a. Going off of the query, or tweet, we want to be able to display common tweets from the pool of users and their respective tweets in the past 7 days. Say my query is “I love seafood”. The ranking will be conducted among our pool of tweets and we would show the top tweets that may have similarity in the topic/category.
2. Get 2000 users, and their last 7 days of tweets
    a. For the purpose of the project we want to go ahead and observe ranking in a small-ish pool of people and tweets. I think it would be informative to observe the ranking when using a pool of 1000 users versus a pool of 1000 users for example. The more tweets are available, the more we should see a higher commonality between the tweets given back from our ranking.
3. Process data
    a. We should process the data of tweets that we receive to eliminate words or other elements in the tweet that are not helpful in terms of ranking the tweet for our query.
    b. Source or guidance for processing with python: https://towardsdatascience.com/basic-tweet-preprocessing-in-python-efd8360d529e
4. We should get information from 2000 users and their past 7 days of tweets into the following tables:
    a. original_ tweets table:
        i. We want to obtain the information about a tweet, who wrote it, the time posted, the content type etc. This will be stored in the original_tweets table.
    b. processed_tweets table:
        i. We want to then process the tweet text and content in order to simplify our ranking model.
    c. users table:
        i. We want to create a users table so that we can see the correlated tweets per user. We can do this multiple ways: tie the tweet id’s in an array that correspond to the user. We want to also include the location of the user, number of followers, etc of what’s included in the user-object https://developer.twitter.com/en/docs/twitter-api/v1/data-dictionary/overview/user-object
    d. user_category table:
        i. What kind of topics does this user rank mostly in? We should aggregate counts of topics that their previous seven days of tweets include in. This would be useful to observe at the end in our conclusion. If a user is most likely to post about politics, it would be interesting to see if they rank among the higher ranked tweets about politics in our ranking algorithm.
    e. tweet_category table
        i. Last but not least, we want to have a categories table where we can map the category, we can also create a list of tweets belonging in this table
5. Build an inverted index

    a. We should create an inverted index for every word presented in a tweet to further allow the ranking algorithm to rank a tweet accordingly.
6. Grab the top X users(on ranking BM25) that show up in terms of different categories.
    a. Use all of the categories (display invalid if not in the categories available)
        i. We want to be able to use all of the categories available in order to do ranking with the tweets related. If a query is present where it doesn't fit in with any of the categories, we shall show "no category fits this tweet"
        ii. Users being the top performing tweets for a given category
            1. We want to also display top performing users for a given category. This can be done by collecting the overall topics this user tweeted about and presenting the ones with higher frequency under their id.
        iii. Top categories in sample set
            1. Depending on the content that is aggregated in the past 7 days it would be informative to correlate these tweets with ongoing present news. Would be good to see if the ranking is similar to most searched on twitter charts and trending statistics.
7. Display top X on the terminal/UI.
    a. We will have an interactive terminal asking the user what they would like to display in terms of ranking as show in number 6.