Google HackNU 2021

Local search engine

Searching through a large number of documents sounds like a familiar problem, right? This hackathon will expose you to a mix of technologies that can be used to index documents and quickly query them afterwards.

You're given a large set of news articles, which you need to make searchable by a number of keywords instantly[1]! In the following we'll give you a high level structure of milestones that you could encounter during your implementation. We'll leave technologies and approach completely to you, with the exception that you should work on building a local index (also see rules).

During the presentation we'll ask you to use your local search engine to search for a set of keywords. Some examples for these keywords are: "work desk", "presidential election", "Olympic closing ceremony" or "documentary". Your search engine should output a ranked list of 5 articles which are the best match based on the given keywords and the time it took for the query to execute.

Based on a few different searches[2] we'll try to judge the winner based on 1) query accuracy and 2) query speed as objectively as possible!

Dataset

You need to use this open Kaggle news articles dataset for building and testing your Local search engine. Access and download this dataset directly from Kaggle. You will need a Kaggle account to do so (registration is free and easy :)). This dataset is available to all the Kaggle users under Kaggle's Terms of Use.

This dataset[3] contains 3 .csv files, each having roughly 50,000 entries. Each entry corresponds to one news article, which results in around **150,000 news articles in total** you need to work with. Each .csv file contains 8 columns: `[article_num, article_db_id, title, publication, author, date, year, month, url, content]`. You should mainly focus on the `content` column for this task.

[1] A few seconds, but faster is better!
[2] These searches will be reasonable (only real words, no searches like "a a and or")
[3] There is a link in the dataset description to a larger version of this dataset that contains 2.7 million news articles in total. You probably do not want to use that larger dataset as it will take too much time to process it.

1st Stage Preprocessing

The articles you're given are in a raw form and you'll need to parse them in order to create an index of them. There are many existing NLP libraries that you can make use of (e.g. NLTK, SpaCy) which will make this significantly easier.

Questions that you should ask yourself:

- What should we use as keys in our index (e.g. words)?
- How should we store the index persistently for later usage?
- Optional: What kind of additional preprocessing could make our index perform better?
    - How can we rank or weigh the keys?
    - Should some words be filtered out?
    - Can we combine words of the same meaning (e.g. walk, walking)?

Tip: Depending on your solution, building an index can take a while (up to multiple hours), so make sure you save them in advance of the presentation!

2nd Stage Querying

Once you have preprocessed your data and stored it in a persistent index, you'll need to build a query engine on top of it. The query engine should use the index to find the 5 most relevant articles based on a set of words. How you want to implement this is up to you, however a graphical/pretty interface is not necessary and does not give bonus points (command line tool is fine). The main point is to ensure that querying demonstrates that your Local search engine works.

You'll need to ask yourself how you can use your index to rank articles. For example one possible factor is word/text frequency within articles.

Rules

- Your solution needs to be executable without an internet connection. As in, you shouldn't use online APIs in your solution.
- You CAN use any type of offline libraries like NLTK, SpaCy, etc. to make your life easier.
- Be fair, write your own solution! This hackathon is a fun learning experience, treat it as such :)