Programming Assignment #3 (due in 4 weeks – day 7 of Module 7)

Computing Document Similarity

The principal reason to create inverted files is to support ad hoc querying where a document collection is relatively static, and a number of *a priori* unknown user queries is evaluated. For this assignment you will put together the pieces from the first two programs (*i.e.,* dictionaries and inverted files) and create a simple but true to practice retrieval engine based the cosine similarity vector space model. For this assignment you will need to process a larger document collection than what we used on the first two assignments. You will then demonstrate the ability to score and rank documents in order of their presumed relevance to queries. For full credit, you need to first build an index on disk (*i.e.,* just as you did in Program #2), and then have your query processing program load the dictionary and retrieve postings lists for terms from the inverted file (as needed) in order to rank documents for a provided set of queries.

Note: you are expressly forbidden from using existing text retrieval APIs or engines for indexing and computing similarity (*e.g.,* Apache Lucene, pyserini). The goal of this assignment is to experience implementing this functionality yourself. If you have doubts about the permissibility of using any open source code on this assignment, post a clarification question in the discussion forum or contact the instructor.

Dataset
On this assignment you will work with a 360 MB sized test collection of biomedical articles related to COVID-19. This data was used in the 2020 NIST TREC COVID shared task.[1] The collection contains 191,175 documents. Some are very short and contain only a title. Others will have abstracts and other paragraphs of text. Compared to the *Yelp* and *Headlines* datasets these documents are generally larger, contain many numbers (possibly including tables), and have many scientific terms. There are often very long lines with multiple sentences. The files are formatted with similar SGML markup as was used in previous assignments. The document collection is provided as a zip file (that has been encrypted[2]) and it is protected using a passcode that I will provide the class with. I encourage you to run your Program #1 code on the collection or perform a similar analysis to see if you are still satisfied with your tokenization for this text.

Indexing (10 points)
Build an index from the document collection. For vector cosine scoring you will also want to calculate document vector lengths after building the inverted file and store these for later use. Video lecture 4E gives some helpful advice.

Query Parsing (10 points)
Along with the single SGML marked-up file containing documents, a link to a similarly formatted set of queries for the TREC COVID dataset has also been provided on the course web page. Your retrieval engine program must automatically process this file of queries and create a bag of words representation for each query. **Print out the processed query terms and their weights** that your system uses for only the first topic (**not** for all 50 topics). The proper way to calculate document similarity scores is by looping over query terms and calculating partial scores for documents by traversing the information in the postings list for each term.

Cosine Scoring (up to 50 points) **or** much simpler **Dot-Product Scoring** (up to 35 points and a 15 point deduction)

Cosine Option (50 points): For full credit on this project, implement cosine scoring. Use TF/IDF term weighting for both documents and the query, and compute cosine similarity for all documents in the collection containing at least one of the query terms.[3] Produce a single output file that contains ranked documents for each of the topics, *carefully* following the detailed formatting instructions below. Important details about implementing cosine scoring correctly include: (a) computing IDF values appropriately (note: IDF weights are learned from the corpus, not the queries); and, (b) properly computing document vector length, and query vector length. As a much simpler alternative you may choose to use dot-product scoring instead, for a 15% deduction.

[1] https://arxiv.org/abs/2004.10706

[2] The data are copyrighted and encryption protects against unauthorized uses; you should not redistribute the data or use it for any non-educational purpose.

[3] Of course you may remove stop words from your documents and/or query vectors. This is quite reasonable, and probably even a good thing to do. Just rank documents that contain at least one of the remaining terms.

Dot-Product Option (35 points): Dot-product still uses the inverted file information but is much simpler than vector consine. Use term frequency information from the query and documents to compute a score for each document that contains one or more query terms, and do not use IDF-weights. Here is a dot product example:

Query (Q): “polio vaccine effectiveness”
Document (D): “jonus salk created a polio vaccine in 1952. There is no known cure for polio.”

The term ‘effectiveness doesn’t appear in D. The query term ‘polio' occurs 2 times and ‘vaccine appears once. Thus the dot product for this query and document would be: (2*1 + 1*1 + 0 * 1) = 3.

Choose either Cosine or Dot-Product scoring and produce an output file following the description below. Dot-product scoring does not normally produce as good of a ranked list as vector cosine. In ranking documents, you may split ties in any fashion. **Additional notes**: **(1)** If you implement cosine scoring, then do not create a dot product ranking; only supply the cosine-based ranked list. **(2)** For both cosine scoring or dot-product scoring it is important to score documents in parallel, computing partial results based on each term until all query terms have been processed.

Batch Processing and Ranked Lists (20 points)
To put all of these pieces together, your program should process an input query file and produce an output file that provides scores of the top-k documents. A web search engine like Bing or Google usually offers the top 10 documents. For this assignment, I want you to provide the top 100 documents for each query. Your single output file must follow ***precise*** formatting instructions. For each query (list them in the same order as the input file) include documents in ranked order (ascending, with 1 being the first rank). Each line should have six columns that are separated by a single space. The first column should be the query id (1, 2, ..., 50); the second should be the string “Q0” (read as 'capital Q zero'); the third should be the integer document id, the fourth should be the rank (1, 2, ... 100); the fifth should be your numerical score (real or integral, but not scientific notation like 1.2e-5); and, the last column should be your JHU JHED ID (e.g., jdoe3, tsmith7). Your ranked list should contain the top ranked 100 documents for each query (unless you cannot find 100 documents with the query terms, which is unlikely), but do not include more than 100. Do not include any blank lines. This file should be named *yourjhed*-a.txt and will be submitted separately in Canvas.

Desired output format (example). Note, fields are to be separated by a single space.

```
 1  Q0  152478   1    0.287899  yourjhed
 1  Q0  68724    2    0.260185  yourjhed
 1  Q0  102318   3    0.228558  yourjhed
 ...
50  Q0  94781    99   0.396122  yourjhed
50  Q0  46937    100  0.387620  yourjhed
```

Length Experiment (10 points)
To explore whether longer queries lead to more accurate retrieval results I want you to produce a second ranked list using a longer form of the topics based on questions instead of keywords. The second file should be named *yourjhed*-b.txt and submit that in Canvas as well. If you like, you may opt-out of doing this second experiment – the penalty is only 10% of the assignment.

Expected Deliverables

Please report the following:

- A brief summary describing your approach and any problems encountered. The summary should describe your tokenization and stemming decisions, how terms are weighted, and, which similarity measure you are sing to rank documents.
- The number of documents successfully indexed
- The size of the vocabulary (*i.e.,* the number of unique indexing terms in the corpus)
- Run-time (minutes in wall-clock time) to build the index
- Disk space consumption (MBs), both for your inverted file and any other data structures
- Your weighted terms for the first query (both keyword and questions)
- Run-time (minutes in wall-clock time) to process all topics (for both keyword and question versions)

You should submit:

1. A file "yourjhed-a.txt" with document rankings (keyword topics).
2. A file "yourjhed-b.txt" with document rankings (question topics).
3. A single PDF file containing the summary, program output (other than the requested ranked lists), any other requested details, and logically arranged source code.

Free Advice

I suggest testing using the toy "animal" corpus to see if your cosine / dot product scores are correct. If you are experiencing performance issues, you may also want to test or debug using smaller portions of the COVID corpus (*e.g.,* 10% or 25%).

If you are unable to index the entire collection (for example, because your programs consume too much time or memory), process as much as you can, and clearly state the number of documents that you were able to index. However, check for correctable inefficiencies before giving up on indexing the whole collection. I wrote a fairly simple Java program[4] that builds an in-memory index for this dataset and writes it to disk; the program used 3.2 GB of RAM and it runs in about 50 seconds on a two year-old laptop. My inverted file takes up about 193 MB. Your numbers may differ substantially from mine and that is okay – I'm just giving you a data point to know that this is feasible.

This assignment requires more effort than the first two programs. If you run short of time you might want to consider implementing dot-product scoring instead of cosine, or choosing not to submit the length experiment. Both options involve reductions in points, but generally less than a late submission.

Sample Queries (Keywords 1-3; Questions 4-5)

```
<Q ID=1>
 coronavirus origin
</Q>

<Q ID=2>
 coronavirus response to weather changes
</Q>

<Q ID=3>
 coronavirus immunity
</Q>

<Q ID=4>
 what causes death from Covid-19?
</Q>

<Q ID=5>
 what drugs have been active against SARS-CoV or SARS-CoV-2 in animal studies?
</Q>
```

[4] Less than 500 lines of pre-1.8 Java code.

Sample Documents

<P ID=9560>

COVID-19 y aparato digestivo

</P>

<P ID=9561>

An Integrated Program in a Pandemic: Johns Hopkins Radiation Oncology Department

</P>

<P ID=9562>

French Sarcoma Group proposals for management of sarcoma patients during the COVID-19 outbreak

</P>

<P ID=9563>

Treatment of Argentine hemorrhagic fever

Argentine hemorrhagic fever (AHF) is a rodent-borne illness caused by the arenavirus Junin that is endemic to the humid pampas of Argentina. AHF has had significant morbidity since its emergence in the 1950s, with a case-fatality rate of the illness without treatment between 15% and 30%. The use of a live attenuated vaccine has markedly reduced the incidence of AHF. Present specific therapy involves the transfusion of immune plasma in defined doses of neutralizing antibodies during the prodromal phase of illness. However, alternative forms of treatment are called for due to current difficulties in early detection of AHF, related to its decrease in incidence, troubles in maintaining adequate stocks of immune plasma, and the absence of effective therapies for severely ill patients that progress to a neurologic–hemorrhagic phase. Ribavirin might be a substitute for immune plasma, provided that the supply is guaranteed. Immune immunoglobulin or monoclonal antibodies should also be considered. New therapeutic options such as those being developed for systemic inflammatory syndromes should also be evaluated in severe forms of AHF.

</P>

<P ID=21107>

In Case You Haven't Heard…

A World Health Organization (WHO) official says the United States has the potential to become the new epicenter of the COVID-19 crisis as a large acceleration of infections is occurring in the nation, Changing America reported March 24. “We are now seeing a very large acceleration in cases in the U.S. So, it does have that potential,” WHO spokeswoman Margaret Harris told reporters when asked whether the United States could become the new epicenter, according to Reuters. At the time this issue of MHW went to press, the United States had more than 46,500 confirmed cases, with nearly 600 deaths, according to Johns Hopkins University data. New York state on March 23 saw an increase of more than 4,000 confirmed cases since the day before, according to The New York Times. At press time, only China and Italy had recorded more cases than the United States. China had tallied more than 81,000 cases, with more than 3,200 deaths, while Italy had more than 63,900 confirmed cases, with more than 6,000 deaths. Worldwide, more than 395,000 cases have been confirmed, with more than 17,000 deaths.

</P>

<P ID=51710>

[Progress and challenge of vaccine development against 2019 novel coronavirus (2019-nCoV)].

The outbreak of 2019 novel coronavirus (2019-nCoV) infection poses a serious threat to global public health. Vaccination is an effective way to prevent the epidemic of the virus. 2019-nCoV along with severe acute respiratory syndrome coronavirus (SARS-CoV) and Middle East respiratory syndrome coronavirus (MERS-CoV) belong to the same β-genus of coronavirus family. Base on the previous experience and the technical platform of developing SARS-CoV and MERS-CoV vaccines, scientists from all over the world are working hard and quickly on the related fields. There are substantial progress in these fields including the characterizing the 2019-nCoV virus, identification of candidate antigens and epitopes, establishment of animal models, characterizing the immune responses, and the design of vaccines. The development of 2019-nCoV vaccines cover all types: inactivated virus vaccine, recombinant protein vaccine, viral vector-based vaccine, mRNA vaccine, and DNA vaccine, et al. As of March 2020, two 2019-nCoV vaccines have entered phase I clinical trials. One is named as Ad5-nCoV developed by the Chinese Institute of Biotechnology of the Academy of Military Medical Sciences and Tianjin Cansino Biotechnology Inc. Ad5-nCoV is based on the replication-defective adenovirus type 5 as the vector to express 2019-nCoV spike protein. The another vaccine is mRNA-1273 developed by the National Institute of Allergy and Infectious Diseases and Moderna, Inc.. RNA-1273 is an mRNA vaccine expressing 2019-nCoV spike protein. Although the rapid development of 2019-nCoV vaccine, it still faces many challenges with unknown knowledge, including the antigenic characteristics of the 2019-nCoV, the influence of antigenic variation, the protective immune response of host, the protection of the elderly population, and the downstream manufacturing process of the new vaccine. The safety and efficacy of vaccines are the first priority for vaccine development and should be carefully evaluated.

</P>