COL 764 Assignment 3 - Prior Ranking of Documents

2021 - Version 1.0

Deadline

Deadline for final submission of the complete implementation and the report on the algorithms as well as performance tuning: November 12th 2021, 11:59 PM

Please note that this deadline is not extendible (even with a penalty). You must submit before the specified deadline.

Weightage

This assignment is evaluated against 100 marks. The tentative breakup of marks is given at the end of the document.

Instructions

1. This programming assignment is to be done by each student individually. Do not collaborate -either by sharing code, algorithm, and any other pertinent details- with each other.
2. All programs have to be written either using Python/Java/C/C++ programming languages only. Anything else requires explicit prior permission from the instructor. The machine we use to evaluate your submissions has the following versions:
   - **C**, **C++**: gcc7;
   - **Java**: java 11 (we will not use OpenJDK);
   - **Python:** version 3.7.

   We recommend you use the same versions to avoid the use of deprecated/unsupported functions or ensure required compatibility.
3. A single tar/zip of the source code has to be submitted. The zip/tar file should be structured such that

- upon deflating all submission files should be under a directory with the student's registration number. E.g., if a student's registration number is `20XXCSXX999` then the zip submission should be named 20XXCSXX999.zip and upon deflating **all contained files** should be under a directory named ./20XXCSXX999 only (names should be in uppercase). Your submission might be rejected and not be evaluated if you do not adhere to these specifications.
- apart from source files, the submission zip file should contain a build mechanism *if needed* (allowed build systems are Maven and Ant for Java, Makefile for C/C++). It is the responsibility of each student to ensure that it compiles and generates the necessary executable as specified. **Please include an empty file in case building is not required** *(e.g. if you are using Python)*. **Note that we will use only Ubuntu Linux machines to build and run your assignments.** So take care that your file names, paths, argument handling etc. are compatible.

4. You *should not* submit data files, index files, dictionaries etc. If you are planning to use any other "special" library, please talk to the instructor first (or post on Teams).
5. **Note that there will be no deadline extensions. Do not wait till the end.**

1 Assignment Description

In this assignment, the goal is to simply compute prior similarity-based authority of documents and qualitatively analyse the results. We will use a small subset of the famous `20newsgroups` dataset (commonly used in text classification, but we have repurposed it). The dataset is made available from: `https://www.cse.iitd.ac.in/~srikanta/course/col764-2021/20news-bytedate-test.tgz`

The task consists of the following:

1. Compute a *similarity graph* between each pair of documents in the collection using a similarity function Sim (the choices for this function are given later in this document).
2. On this similarity graph, compute the PageRank scores of all documents.
3. Analyze your results and present them in the report.

20 Newsgroups

As the name suggests, this is a collection of newsgroup documents from about 20 different newsgroups. Some background about the dataset can be found at `http://qwone.com/~jason/20Newsgroups/`. We are only using the **test** subset of this collection containing just 7,532 documents.

1.1 Similarity Computations

You will use the following two different similarity measures for computing the all-pairs similarity graph on the documents:

Term overlap As the name suggests, the similarity function between two documents is given by:

$$Sim_{jaccard}(d_1, d_2) = \frac{|TermSet_{d_1} \cap TermSet_{d_2}|}{|TermSet_{d_1} \cup TermSet_{d_2}|}$$

The $TermSet_{d_i}$ is nothing but the set of terms that appear in a document **after stemming them using Porter stemmer**. Note that you are expected to retain the stopwords.

TF-IDF similarity For a document d_i, its tf-idf vector is constructed by computing the tf-idf scores of all terms in the document and representing it as a vector of the vocabulary size. You should use the following forms of TF and IDF:

$$\forall_{d_i \in D,\, j \in V} \qquad tf_{i,j} = 1 + \log_2(f_{i,j})$$

$$\forall_{d_i \in D,\, j \in V} \qquad idf_j = \log_2\left(1 + \frac{|D|}{df_j}\right)$$

$$Sim_{cosine}(\vec{d}, \vec{d'}) = \frac{\vec{d} \cdot \vec{d'}}{||\vec{d}|| \times ||\vec{d'}||}$$

Where, V is the vocabulary size (after stemming using Porter stemmer), D is the document collection, df_j is the number of documents that j-th term appears in, $f_{i,j}$ is the frequency of j-th term in a document d_i, and each entry of the vector representation of a document d_i, $\vec{d_i}$, consists of the product $tf_{i,j} \times idf_j$ as given above. Hence the length of all $\vec{d_i}$ is the same and equal to $|V|$.

Note that both the similarity functions are symmetric. If a document contains no terms, you can ignore such a document.

2 Computing PageRank

We studied PageRank as being computed on a directed, unweighted web graph. Here, however, we have an undirected, weighted graph between documents where the weights are based on the corresponding Sim score computed between two documents. Hence you need to first explore how you would use the PageRank algorithm on this graph (Hint: a quick Google search :-)).

Somewhat surprisingly, in this assignment, you are not required to implement PageRank from scratch. You are free to use an external library that computes the PageRank scores (there are libraries like NetworkX, scikit-network, snap, etc., which can easily compute these scores). Your only goal is to make sure your input graph is correct.

For PageRank computation, you should use the **damping factor**, $\alpha = 0.85$. Your PageRank scores have to be accurate to at least 3 decimal places.

3 Analysis and Report

In your report, you should have the following details:

- What was the approach taken to adapt the PageRank algorithm to a weighted, undirected graph? Also, describe your method of choice for computing PageRank scores and your experience.
- List of documents and their PageRank scores with top-20 highest PageRank values for each of the similarity functions.
- What specific conclusions you can draw on the kind of documents that have high PageRank scores?

4 Submission

4.1 Program Structure

You are expected to submit the following:

1. `build.sh` is a shell script that is used to build your program.
2. Computation of similarity graph -
   `simgraph_gen.sh [jaccard | cosine] [collection-directory] [output-file]`
   The `output-file` is generated by your program, with each line consisting of a tab-separated entries:
   `<doc1> <doc2> <sim_score>`
   where `<doc1>` and `<doc2>` are the names of the pairs of the documents and `<sim_score>` is the similarity score computed –using the corresponding method– between the two documents. **Document names have to be of the following form "sci.space/61565"**. **You can skip entries for whose the similarity score is 0**. Report similarity score in not more than 4 decimal places precision.

4.2 Submission Plan

All your submissions should strictly adhere to the formatting requirements given above.

- Submission of your source code on 12th November, and the final version of results.
- You should also submit a PDF report with the details given in Section 3.
- The set of commands (roughly) used for evaluating your submissions follows -

Tentative Evaluation Steps (which will be used by us)

```
$ unzip 20XXCSXX999.zip
$ cd 20XXCSXX999
$ bash build.sh
$ bash simgraph_gen.sh jaccard some/coll/dir/ simgraph_jaccard
$ bash simgraph_gen.sh cosine some/coll/dir/ simgraph_cosine
```

- Note that we set the collection directory as read-only, and you should generate your result files within the unzipped submission directory

4.3 Tentative breakup of marks assignment

In general, a submission qualifies for evaluation if and only if it adheres to the specifications given above (arguments, structure, use of external libraries, correct output format, input format adherence, etc.). Given this requirement, the marks assignment for correct implementation is of:

Term overlap similarity graph	30
TF-IDF similarity graph	30
PageRank scores	20
Algorithmic details in the report	10
Analysis of results	10
Total	100