CMSC 676 Information Retrieval Project Phase-4

Submitted By- Vinayak Salavi (TL26978)

Implementation

Programming language used: Java

Parser used: Jsoup

Approach

- I have exteded the functionality of phase 3. To tackle the problem I have used Term-at-a-Time. I think this approach is usefule as there isnt any restriction of number of words.

Algorithm and Datasctructures

1. The query weights are fetched from the command line. If only words are provied as an argument, then 1 is considered as default weight. This is stored in a hasmap. While storing the query word, words are stored in lowercase.
2. Then query is iterated over term by term.
3. The query word is compared with the inverted index created in phase 3. If matching word is found corresponding document and tf idf weights map is fecthed .
4. Using this map the numerator (of cosine similarity)values are calculated iteratively and stored in a separate map. Also, as part of denominator of cosine similarity depends on the the same values fetched in the step 3, to save proceesing time have calculated doc weights part and saved in a separate map.

```
for (String key : queryMap.keySet()) {
    double docWeightSquaredSum = 0d;
    TreeMap<Integer, Double> tempMap = weightIndexMap.get(key);
    if (null != tempMap) {
        for (int docID : tempMap.keySet()) {
            double currentWeight = tempMap.get(docID);
            docWeightSquaredSum += Math.pow(currentWeight, 2);
            double product = currentWeight * queryMap.get(key);
        numerator values
            numeratorDocumentWeightsMap.put(docID, numeratorDocumentWeightsMap.get(docID) + product);
         denominator values wt(i,ink)^2 + wt(i, pink)^2
            denominatorDocumentWeightsMap.put(docID,
                    denominatorDocumentWeightsMap.get(docID) + docWeightSquaredSum);
        }
    } else {
        System.out.println("word " + key + " not found in corpus\n");
        continue;
    }
}
```

5. Then the part to calulate the query weights of denominator is calculated.
6. Finally to calculate the cosine similarity, I have created the a reusable function to which values from the numerator and denominator maps are passed. Cosine

similarity is saved in a separate map along with the doc id.

```
private static double calaulateCosineSimilarity(double numerator, double denominator_DocPart,
        double queryWeightSquaredSum) {
    // TODO Auto-generated method stub
    if (numerator == 0 || denominator_DocPart == 0)
        return 0;
    double denom = Math.sqrt(queryWeightSquaredSum) * Math.sqrt(denominator_DocPart);

    return numerator / denom;
}
```

7. To print the top 10 results of, I have sorted the similarityscore map and printed the first 10 values. Also, while doing the same, I have fetched the tf idf weights of a document based on the doc id and printed the top 10 tf*idf weights of docs.

- Along with the this have handled the additional checks such as not printing the zero similarity scores, printing appropriate message when query word is not available in the inverted index.

Complexity

- The complexity of the program depends on the following parameters
  - The number of input documents (N)
  - The terms passed in the query (Q)
  - Inverted index list length (I)

  The overall complexity can be summed up as

  O(N+Q+I)

- Additnal constant can also be added as there are additonal iteration used to sort the ma

Steps to execute:

- Compile
  - javac -cp ".\jsoup-1.14.3.jar;" .\FileParser.java

- Run – Program takes following command line arguments:
  - Path to input files
  - Path to store
  - “Wt”followed by weights and query words

  Eg.java -cp ".\jsoup-1.14.3.jar;" FileParser "Input_File_Path" "Output_File_Path" “wt” 0.3 dog 0.5 cat 0.4 rat
  - query words can also be passed without weights by exluding “wt”

  e.g. .java -cp ".\jsoup-1.14.3.jar;" FileParser "Input_File_Path" "Output_File_Path" dog cat rat

Note: Place ‘jsoup-1.14.3.jar’ under the same repository. If not please provide the path to the jar while compiling and executing code.

Output:

PFB screenshots for the output. Also have shared the stored complete output in separate text file and shared along with deliverables.

- The output displays following parts:
  - Doc ID
  - Cosine similarity score for the document
  - Top 10 tf*idf weights from the same document

Diet

```
DocID   CosineSimilarity                Top 10 TFIDF
######################################################################

9       1.0
                                        5.259313171596415
                                        5.063688381461023
                                        4.9262388435197435
                                        4.889143725212306
                                        4.808117363526723
                                        4.768098655798747
                                        4.649823815681194
                                        4.499452315610765
                                        4.269625827706006
                                        4.23679952669287
18      0.816789720027026
                                        5.878433195030678
                                        5.223411841135542
                                        4.951082774544237
                                        4.587896520749842
                                        4.568390487240406
                                        4.545016042469859
                                        4.330211772819084
                                        4.153793004744186
                                        4.130335954062897
                                        3.967025519024688
50      0.42635679437034896
                                        5.722523341800782
                                        5.594539971190038
                                        5.502247504825492
                                        5.37803826380463
                                        5.2018944190255585
                                        5.18873703023008
                                        5.159992522290957
                                        4.971152237627433
                                        4.932635726564489
                                        4.889143725212306
252     0.42306597341753116
                                        4.9226420299015015
                                        4.716226304013403
```

International affairs

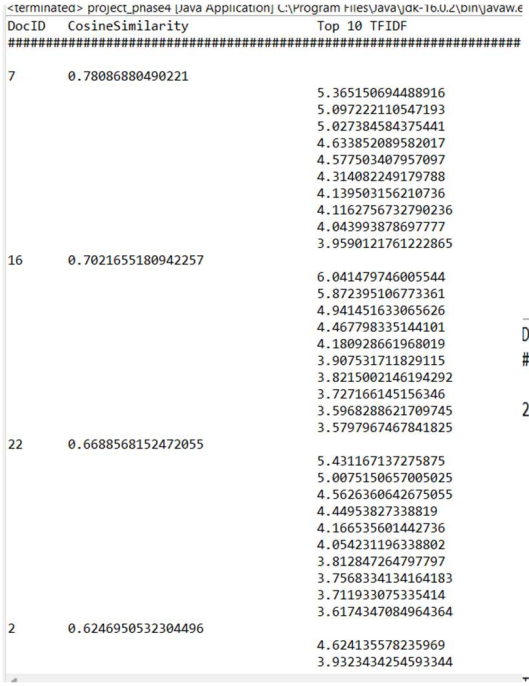
```
DocID   CosineSimilarity                Top 10 TFIDF
######################################################################

1       0.8320503006960566
                                        6.006656457757019
                                        5.826318764879789
                                        5.672240916198829
                                        5.479048417065463
                                        5.3204875149537845
                                        5.207107790974696
                                        4.868529662640034
                                        4.647942367970438
                                        4.17926521466152
                                        4.1300319993932755
10      0.5688611855112126
                                        5.735918368787095
                                        5.520842666156838
                                        5.26073090824144
                                        4.905666867121631
                                        4.409631287960037
                                        4.086163161690186
                                        4.083762259645986
                                        3.8566588737094287
                                        3.678641517864465
                                        3.570781520512883
2       0.5417406235591291
                                        4.624135578235969
                                        3.9323434254593344
                                        3.723713295185555
                                        3.6797607775403245
                                        3.6415888553349762
                                        3.593620321343614
                                        3.4087016467774616
                                        3.37446041607145
                                        3.292213930492086
                                        3.2705965555240732
19      0.5324042477830555
                                        5.061894937623095
                                        4.0010185878218385
```

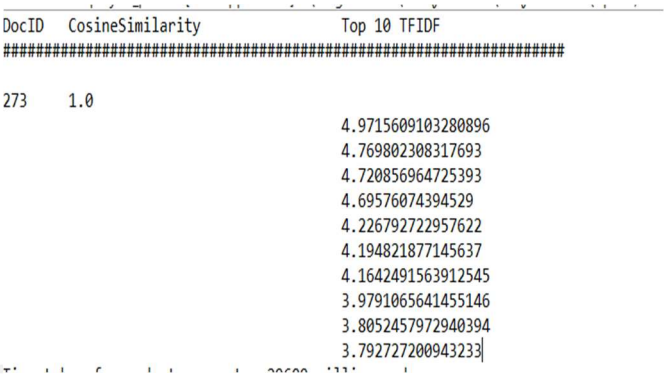
Zimbabwe: This word is not availble in the corpus.

```
word Zimbabwe not found in corpus
```

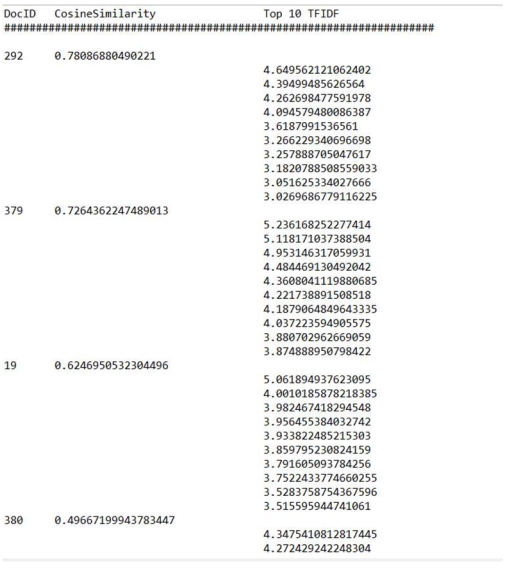
Computer network



hydrotherapy



Identity theft



Query Term Weights

The term which doesn't add any significant meaning to the query, word should be assigned the minimum weights. If the given query contains the weight then query must start with word "wt" or just query words can be passed. If a word is passed two times in a query(when weights are not passed), the default weight 1 is multiplied by the number of occurrence. The higher weight of the word suggests that the doc having such words will be given more priority