CS 6322: Information Retrieval

Ashika Prakash Acharya
axa190084

HomeWork- 3

Program Description

The program implements a simple statistical relevance model based on the vector relevance model, using the term-based index that was built in the last assignment. The vector representations of queries and documents are used to determine scores that inform the ranking of documents against the queries. The scores are obtained by computing the cosine similarity for every query-document vector pair.

FOR each query:

1. Turn in the vector representation of the query (10 points per weighting scheme), and the top 5 documents ranked for the query under both weighting schemes. You are also required to present the vector representations for each of the first 5 ranked documents.

The output is available in the text files

output_w1.txt
output_w2.txt

2. Indicate the rank, score, document identifier, and headline, for each of the top 5 documents for each query.

The output is available in the text files

output_w1.txt
output_w2.txt

3. Identify which documents you think are relevant and non-relevant for each query by inspecting the documents.

Query	Top 5 Documents using w1 function	Relevant	Non relevant	Top 5 Documents using w2 function	Relevant	Non relevant
Q1: what similarity laws must be obeyed when constructing aeroelastic models of heated high speed aircraft	0486 0013 0012 0665 0014	0486 0012	0013 0665 0014	0486 0665 0013 0012 0573	0486 0013 0012	0665 0573
Q2: what are the structural and aeroelastic problems associated with flight of high speed aircraft.	0012 0014 0746 0051 0781	0012 0746 0051	0014 0781	0012 0051 0014 0746 0875	0012 0051 0746	0014 0875

Q3: what problems of heat conduction in composite slabs have been solved so far	0485 0144 0005 0090 0091	0144 0005 0090	0485 0091	0485 0005 0399 0144 0090	0005 0144 0090	0485 0399
Q4: can a criterion be developed to show empirically the validity of flow solutions for chemically reacting gas mixtures based on the simplifying assumption of instantaneous local chemical equilibrium	1061 0166 0488 1189 0185	1061 0166 0488 1189	0185	0166 0488 1061 1189 0185	0166 0488 1061 1189	0185
Q5: what chemical kinetic system is applicable to hypersonic aerodynamic problems	0103 1032 0625 1296 0943	1032	0103 0625 1296 0943	0103 0943 1032 0625 1296	1032	0103 0943 0625 1296
Q6: what theoretical and experimental guides do we have as to turbulent couette flow behavior	0491 0798 0315 0257 0160	0491 0798 0315 0257 0160		0491 0798 0315 0257 0160	0491 0798 0315 0257 0160	
Q7: is it possible to relate the available pressure distributions for an ogive forebody at zero angle of attack to the lower surface pressures of an equivalent ogive forebody at angle of attack	0492 0124 1040 0057 0434	0492 0057 0434	0124 1040	0492 0124 0057 0056 0232	0492 0056 0232	0124 0057
Q8: what methods -dash exact or approximate -dash are presently available for predicting body pressures at angle of attack	0688 0443 0556 0476 0711	0688 0711	0443 0556 0476	0019 0556 0688 0753 0433	0668	0019 0556 0753 0433
Q9: papers on internal /slip flow/ heat transfer studies	0021 0022 1215 0306 0571	0021 0022 0306 0571	1215	0021 0306 1215 0045 0022	0021 0306 0022	1215 0045
Q10: are real-gas transport properties for air available over a wide range of enthalpies and densities	0493 0302 0949 1009 1143	0493 0302 0949	1009 1143	0493 0302 1143 1264 0436	0493 0302	1143 1264 0436
Q11: is it possible to find an analytical, similar solution of the strong blast wave problem in	0495 0025 0654	0495 0025 1327	0654 0556	0495 0025 0654	0495 0025 1327	0654 0472

the Newtonian approximation	1327 0556			1327 0472		
Q12: how can the aerodynamic performance of channel flow ground effect machines be calculated	0624 0966 0650 0704 0941	0624 0966 0704	0650 0941	0624 0966 0650 1232 0506	0624 0966	0650 0506 1232
Q13: what is the basic mechanism of the transonic aileron buzz	0496 0903 0520 0643 0199	0496	0903 0520 0643 0199	0496 0903 0520 0643 0199	0496	0903 0520 0643 0199
Q14: papers on shock-sound wave interaction	0798 0170 0345 0439 0256	0798 0170 0439	0345 0256	0170 0798 0439 0256 1364	0170 0798 0439 1364	0256
Q15: material properties of photoelastic materials	0462 1025 0463 0082 1043	0462 0463	1025 0082 1043	0462 0463 1025 1099 1340	0462 0463	1025 1099 1340
Q16: can the transverse potential flow about a body of revolution be calculated efficiently by an electronic computer	0498 0869 0093 0106 1280	0498 0106	0869 0093 1280	0498 0869 0093 1286 1280	0498	0869 0093 1286 1280
Q17: can the three-dimensional problem of a transverse potential flow about a body of revolution be reduced to a two-dimensional problem	1108 0106 1281 0916 0498	1108 0106 0916	1281 0498	0106 1108 0916 1281 0498	0106 1108 0916	1281 0498
Q18: are experimental pressure distributions on bodies of revolution at angle of attack available	0248 0197 0234 0498 0124	0197 0124	0248 0234 0498	0197 0498 0124 0248 0234	0197 0124	0498 0248 0234
Q19: does there exist a good basic treatment of the dynamics of re-entry combining consideration of realistic effects with relative simplicity of results	0082 0706 0237 1279 0713	0082 0713	0706 1279 0237	0082 0706 0237 1279 0831	0082	0706 1279 0237 0831
Q20: has anyone formally determined the influence of joule heating, produced by the induced current, in magnetohydrodynamic free convection flows under general conditions	0500 0270 0450 0087 0458	0500	0270 0450 0087 0458	0500 0270 0458 0087 0450	0500	0270 0458 0087 0450

4. Describe why the top-ranked non-relevant document for each query did not get a lower score.

The non-relevant documents got a high score because they contained few terms of the query which had relatively more weight. And as these terms had minor importance in the relevance of the matching but due to their high frequency in document, the document got higher score. This is the reason why they received higher weight and they were irrelevant.

5. Briefly discuss the different affects you notice with the two weighting schemes, either on a query-by-query basis or overall, whichever is most illuminating. For example, you can point out that the weighting scheme seems to be working for this query as well as a list of other queries, but not for some other queries you have noticed. Try to explain why it works and why it does not work.

As per the problem statement, the two weighting schemes differ in a way that W1(MAX_TF) uses Maximum Term Frequency and W2(Okapi) weighing scheme uses average document Length and Document Length.

W1 weighting scheme: This is based on the term frequency. So, the weights and score are dependent on the frequency of occurrence of the term. The schema will ensure that if a document contains the tokens of the query, it is given a higher score. The problem with this is that it doesn't see the actual meaning or semantic meaning of the word in document or in query. So, if the word matches it gives in the result even though it is irrelevant.

W2 weighting scheme: This is based on the length of a document. The term frequency is normalized by the document length. As a result of this, long documents which do match the query term can often be scored unfairly. Similar to W1, this schema also doesn't take into account the actual or semantic meaning of the word in query or document

6. Describe the design decisions you made in building your ranking system.

Design and Algorithm

The code is in written in Python and does the following task in the mentioned order.

- The code reads the Index file that was created in Homework2 `'Index_Version1_uncompress.txt'` and stores them in memory as `{lemma_index}.`
- `hw3.queries` file is read and stored in memory as `[queries].`
- The code repeats the following section twice for each weight computation formula.
    - Each query is parsed into tokens and then lemmatized.
    - `get_document_rankings()` uses the vector representation of the queries and documents to determine scores that inform the ranking of documents against the queries. The scores are obtained by computing the cosine similarity for every query-document vector pair.

```
CosineScore(q)
 1  float Scores[N] = 0
 2  Initialize Length[N]
 3  for each query term t
 4  do calculate w_{t,q} and fetch postings list for t
 5     for each pair(d, tf_{t,d}) in postings list
 6     do Scores[d] += wf_{t,d} × w_{t,q}
 7  Read the array Length[d]
 8  for each d
 9  do Scores[d] = Scores[d]/Length[d]
10  return Top K components of Scores[]
```

- `get_top5()` sorts the `{scores}` in descending order of the ranking and returns top 5 documents for the query.

The figure above shows the algorithm for computing cosine score for query-document pair.