Information Retrieval (CS60092)

Default Project 2022A part-II and III

Scoring and Evaluation, Relevance Feedback

Name: Kistamgari Sri Harika Reddy Roll Number:19EC10035 (Group-9)

Contribution:

I worked on the **first part** of the **second assignment** which is based on ranking the documents based on TF-IDF vectorization and the **second part** of the **third assignment** which is based on identifying the important words from the documents that are closer to the query.

Task 2A (TF-IDF Vectorization):

I first parsed the command line arguments to access the corpus of the document and the inverted index constructed in the first assignment. I then calculated the document frequencies of each term in the vocabulary by accessing the lengths of posting lists of each term in the inverted index. Following this, I calculated the term frequency corresponding to each term in every document by counting the number of instances of each term in each of the processed documents (code from the first assignment has been re-used for pre-processing). Using the DF(t) and TF(t,d) values obtained, I created vocabulary-sized vectors for each document following lnc (1+log tf, df = 1, cosine normalization) and anc (0.5+0.5*tf/max tf[d], df = 1, cosine normalization) schemes. I also resolved bugs in the computation of query vectors (in the lpc and apc schemes) and fixed the function to write the obtained

dictionaries to a CSV file according to the format described in the problem statement

Main Issues encountered and approaches taken:

When the code was run on the entire corpus, I encountered memory errors due to the large size of NumPy arrays needed to store the TF-IDF vectors. To tackle this, we have reduced the size of the vocabulary. After generating the top 50 documents corresponding to each query, I noticed that the retrieved documents aren't matching with the gold standard documents in qrels.csv. Upon closer observation (by generating intermediate vector representations for queries, and documents), it has been observed that many documents with high relevance in qrels.csv either had empty abstracts or weren't found in the document corpus.

Design choice: To reduce the size of the vocabulary, we have included lower casing as one of the pre-processing steps and it removed 20K words from the vocabulary. Following this, we removed all those terms which appeared less than thrice in the entire corpus and this reduced the vocabulary size to ~45k terms (any size >50k was throwing memory error). I have verified that no terms included in the query were removed from the vocabulary (the frequency of the rarest term in queries is 9. Hence, all query terms are retained in the vocabulary and removal of other less frequent words is unlikely to affect the IR system). We also used garbage collection to optimize the memory usage of the program.

Task 3B (Identifying words from pseudo-relevant documents that are closer to the query):

For each query, I retrieved the top 10 documents from the rankings based on TF-IDF scores. By reusing the code from part 2A, I then calculated the document frequencies of each term in the vocabulary and calculated the term frequency corresponding to each term in every document. For the top-rated documents, I obtained the TF-IDF vectors (Following the lnc scheme). Following this, for each

query, I obtained a score corresponding to every vocabulary term by averaging the TF-IDF vectors of the top-rated documents. Once the scores are obtained, the top 5 scores corresponding to each query have been retrieved by sorting the averaged vector values. I then saved the obtained results in the suggested format to a CSV file i.e. corresponding to every query, listed the top 5 words important according to a particular vectorization scheme.