Information Retrieval - Sessional 3

A. Mustafi

August 18, 2023

1. Create an inverted index for the documents in the classic corpus. For this exercise you can assume that the index can be a dictionary stored entirely in the primary memory. You can parse your documents to contain only tokens of size greater than 3 characters and containing only alphabets, digits, hyphens and apostrophe. Also you can drop off the stop words found in the file stopwords.txt. What is the size in bytes of your index? (You can use the function sys.getsizeof()).

2. Plot a curve showing the increase in the size of the vocabulary with increase in the number of documents parsed. Which law is this supposed to follow? Can you calculate the appropriate coefficients?

3. Write a merge based document retrieval for an user query containing M terms. You can assume the query is a conjunction.

4. Store the entire index in the secondary storage of your device. Assuming each new query requires you to fetch the entire index to primary memory what is the difference in time in answering a single term query if the index is in the primary memory *vis-a-vis* th eindex is in the secondary memory. (You can use Python's pickle function for this exercise)

5. Try to plot the Zipf's law estimate for your index. Does it seem to follow what is expected?

Weekend Project

Try to recode the previous exercise where 10% of the total size of the vocabulary may be cached in the memory. Initially the entire index is assumed to be in the secondary storage. It is fetched to the primary memory for the first query and the postings list for the query is retained in an memory data structure (i.e. it is cached). Thus, we can cache 10% of the vocabulary size this way. If the cache is full follow an LRU algorithm to empty one item. For *"k"* iterations of the proposed method using *"k"* random terms from the vocabulary evaluate the performance of this algorithm *vis-a-vis* if the index was retained in memory