Maral Dicle Maral
03.12.2021

**CMPE 493 INFORMATION RETRIEVAL**

**ASSIGNMENT 2: A Simple Search System for Phrase and Free Text Queries**

In this assignment we built a simple information retrieval system for phrase and free text queries with tf-idf and cosine similarity approach.

**Index_building module is working with:**

```
python3 index_build.py ./reuters21578/ ./stopwords.txt
```

**As the index is built, Query_processing module is working with:**

```
python3 query_process.py
```

Assignment consists of 3 main steps:

1- Data Preprocessing
2- Index Building
3- Query Processing, Phrase Search and Free Text Search

## 1- Data Preprocessing

Reuters data was read via an HTML file processing library, beautifulSoup. As the documents are read, they are trimmed from punctuations according to the string.punctuation list and cleaned from stop words specified by the user. Latin-1 encoding was used for the documents. Multiprocessing which creates a process for each document was used to speed up the data reading at data preprocessing step.

Maral Dicle Maral

03.12.2021

## 2- Index Building

I built two different indexes that will be used for phrase and free text queries. For phrase queries I used a nested dictionary structure in which words are hashed to dictionary of documents which are hashed to positions of the corresponding word in the particular document. ({word1:{doc1:[pos1,pos2],doc2[pos1,pos2]}}) For free text queries I built three different dictionaries, one nested structure for tf_idf values for words in documents and another dictionary structure for idf values of all the words and one for the document norms.

I preferred dictionary (hashmap structure in python) for indexes because it uses a hash function to link key and values and it provides an O(1) time complexity to store and retrieve a particular object. I preferred a basic array structure for positions list as it has O(1) complexity for adding new elements. It has O(n) complexity for searching items but as the positions lists of words are not large, I thought it won't create a problem during query search. Also, during the index building step, I didn't use any sorting function, which made the process a lot faster.

```
Last login: Tue Dec  7 16:41:04 on ttys001
maral@192 ~ % cd /Users/maral/Documents/Classes/INFO_RETRIEVAL/Assignment2_tf_idf_cosine_search_engine
maral@192 Assignment2_tf_idf_cosine_search_engine % python3 index_build.py ./reuters21578/ ./stopwords.txt
- Index is created in 7.914770 seconds -
maral@192 Assignment2_tf_idf_cosine_search_engine %
```

**Figure 1 (Index Build step takes 7.914770 second for the given reuters data.)**

## 3- Query Processing

At the last step of the assignment, query processing, a phrase query search engine and a free text query search engine is created. Query input is taken from the user and preprocessed, cleaned from stop words and punctuations before it was sent to the engines. Phrase query search engine performs a conjunction operation in the documents and positions level. Free text query search engine uses the tf-idf values of the query terms and tf-idf values of the same terms in the document. Vectors for the query and the documents is created consisting of the tf-idf values. Finally, vectors are compared in terms of their cosine similarity and the list of documents sorted by their cosine similarity is displayed along with their cosine similarity score. For cosine similarity calculation, I computed the tf-idf values for queries and took the tf-idf values of

documents' query words from the tf_idf_index file. Also, during cosine similarity calculation, because the norm of document vectors are not the same with the query vectors, when I took only the query terms into account, this created a problem with the cosine similarity calculation. Therefore I used the actual norms of documents from norm_index.

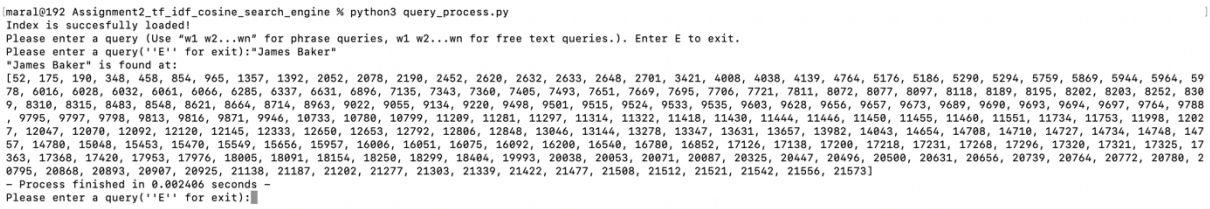


**Figure 2 (Phrase query search for "James Baker", sorted by ID's.)**

```
Please enter a query(''E'' for exit):"fourth quarter performance"
"fourth quarter performance" is found at:
[18, 1019, 2130, 4165]
- Process finished in 0.003970 seconds -
Please enter a query(''E'' for exit):cocoa export shipment tonne
cocoa export shipment tonne is found at:
[(14372, 0.12379653032954416), (1, 0.1133386991237559), (5258, 0.09867527070539589)]
- Process finished in 0.002124 seconds -
Please enter a query(''E'' for exit):E
Thank you for using my search engine!
maral@192 Assignment2_tf_idf_cosine_search_engine %
```

**Figure 3 (Free text query search for cocoa export shipment tonne, sorted by cosine similarity. First element of the tuples represents the document ID's, second element represent the cosine similarity.)**