Information Retrieval

Assignment – 1, Group - 25

Here the assignment for retrieving the document in which their content match according to the Boolean query.

There are some steps for solving this problem

- Preprocessing of file content
- Maintaining the Unigram inverted index data structure for the file content
- Implement the merging Algorithm for the operator (OR, AND, OR NOT, AND NOT)

Methodology

So, we have written 3 .py script (MainA1.py, unigramIndex.py, Qprocessing.py)

MainA1.py file

- Reading the file content from the dataset directory,
- Then preprocessed the file content one-by-one,
- Then creating the Unigram Data structure for each unique words and their posting list(here we have used linked list as a data structure), this linked list containing all the docID, in which the particular word is present.
- After creating the unigram data structure then serialize the class object into a binary file.

unigramIndex.py file

- Maintaining the linkedlist (create or add node to the linked list) used in Unigram data structure.

Qprocessing.py file

- First Deserialized the MainA1.py class object from the binary file, in which their preprocessed unigram data structure stored of the dataset file content, that will be used for the query processing.
- After that taking input (query string, query operator) from the users.
- Then preprocessed the query string, then query processed from left to right in a query, according to the query operator by merging Algorithm.
- After the completint the merging algorithm a final resultant postinglist(linked list) in which the node containing all the docID for a particular query.

- Then print the relevant final output (No. of document matched, No. of comparison done in merging algorithm and list of matched documents name)

Preprocessing

In a preprocessing part we have used these processing for both file content and query string

- Converting all the character in the Lower case
- Removing all punctuation from a string
- Tokenizing the string
- Removing the stopping words
- Applying the Lemmatization

Assumptions

- Input of query and query operator from the user, after preprocessing of the query string, the length of tokenizing list of a query is *m*, then the length of query operator list should be (*m*-1).
  For example: - Input query: **lion stood thoughtfully for a moment**, so this query after preprocessing of tokenization would be [‘lion’, ‘stood’, ‘thoughtfully’, ‘moment’], here the length of this list is 4, so the Input query operator list’s length should be 3.

- Input of query string from the user, after completing all the preprocessing steps of query string, then all the preprocessed term of query should present in the unigram data structure.
  For example: - Input query: **Beautiful rose**, so after completing all the preprocessing steps of this input query, then query becomes [‘beauti’, ‘rose’], then these words should be present in the unigram data structure