Information Retrieval (CS4051)
Programming Assignment No. 1
Spring 2024

Submission Date: March 10, 2024

Assignment Objective

The objective of this assignment is to make you understand how different indexes work in retrieving different query from a collection. You will create Inverted index and Positional index for a set of collection to facilitate Boolean Model of IR. Inverted files and Positional files are the primary data structure to support the efficient determination of which documents contain specified terms and at which proximity. You also learn to process simple Boolean expression queries through this assignment.

Datasets

You are given a collection of ResearchPapers (File name: ResearchPapers.zip) for implementing inverted index and positional index. This zip file contains 20 papers from some research domains. These are all English language documents extracted from PDF. You need to implement a pre-processing pipeline, to get the meaning full features. It is recommended to first review the given text file for indexing. You need to treat each research paper as a unique document(DocID). This observation offers you many clues for your pipeline implementation and feature extraction.

Query Processing

In this assignment, all you need to implement an information retrieval model called Boolean Information Retrieval Model with some simplified assumptions. You need to treat each review (document or file as a document and need to index it content separately). You need to implement a simplified Boolean user query that can only be formed by joining three terms (t1, t2 AND t3) with (AND, OR and NOT) Boolean operators. For example, a user query may be of the form (t1 AND t2 AND t3). For positional queries, the query text contains "/" along with a k intended to return all documents that contains t1 and t2, k words apart on either side of the text.

Basic Assumption for Boolean Retrieval Model

1. An index term (word) is either present (1) or absent (0) in the document. A dictionary contains all index terms.
2. All index terms provide equal evidence with respect to information needs. (No frequency count necessary, but in next assignment it can be, so keeping in mind you can do it now)
3. Queries are Boolean combinations of index terms (at max 3 operands- a more general idea would be excellent for future uses).
4. Boolean Operators (AND, OR and NOT) are allowed. For examples:
   X AND Y: represents doc that contains both X and Y
   X OR Y: represents doc that contains either X or Y
   NOT X: represents the doc that do not contain X
5. Queries of the type X Y / 3 represents doc that contains both X and Y and 3 words apart.

As we discussed during the lectures, we will implement a Boolean Model by creating a posting list of all the terms present in the documents. You are free to implement a posting list with your choice of data structures; you are only allowed to preprocess the text from the documents in term of tokenization in which you can do case folding and stop-words removal and stemming (Porter Stemmer). The stop word list is also provided to you in assignments files. Your query processing routine must address a query parsing, evaluation of the cost, and through executing it to fetch the required list of documents. A command line interface is simply required to demonstrate the working model. You are also provided by a set of 10 queries, for evaluating your implementation. Coding can be done in either Java, Python, C/C++ , C# or in any programming language. There are additional marks for intuitive GUI for demonstrating the working Boolean Model along with phrase query search.

Files Provided with this Assignment:

1. ResearchPapers
2. Stop-words list as a single file
3. Queries Result-set (Gold Standard- 10 example queries)

Evaluation/ Grading Criteria

The grading will be done as per the scheme of implementations, query responses and matching with a gold standard (provided query set).

Grading Criteria:
Preprocessing (2 marks)
Formation of Inverted and Positional Indexes (1 mark for code complexity 1 mark for saving and loading the indexes)
Simple Boolean Queries (2 marks)
Complex Boolean Queries (2 marks)
Proximity Queries (2 marks)
Bonus: GUI (1 mark for making the GUI 1 mark for good friendly GUI)
The proper clean and well commented code will get 2 more marks.

<The End>