Document retrieval system using Apache Lucene

Sasanka Pusapati (sasanka2@illinois.edu)

1. Introduction

Document Retrieval System is a process by which a collection of data is represented, stored, and searched for the purpose of knowledge discovery as a response to a user request (query). With the advent of distributed systems and databases, it became possible to store petabytes of data at a low cost. So, the challenge has been always to find out efficient and useful document retrieval systems that can be used for wide variety of use cases. Implementing an DR system involves a two-stage process: First, data is represented in a summarized format. This is known as the indexing process. Once, all the data is indexed. users can query the system in order to retrieve relevant information. The first stage takes place off-line. The end user is not directly involved in. The second stage includes filtering, searching, matching and ranking operations.

Most of the proposed DR systems are based on the cluster hypothesis [2]. Highly ranked documents relative to a given user query form a cluster that is easy to identify in the case of one simple query. For many complex queries there are query-specific clusters that contain many relevant documents. If those documents are not presented as the top of the result list, this would decrease the retrieval performance. There have been many attempts to propose ranking query-specific clusters techniques. Most of proposed approaches simply compare the representation of the cluster and the representation of the query. Some DR systems make use of additional features such as inter-cluster and cluster-document similarities. Also, Query expansion is another way to boost DR performance.

In this tech review we discuss about a document retrieval system using the Apache Lucene. Then we investigate the relevance of query expansion using parallel corpora and word embeddings to boost document retrieval precision. The next section describes the proposed system and gives details about the expansion process. The third one describes, and analyses obtained results. The last section concludes this paper.

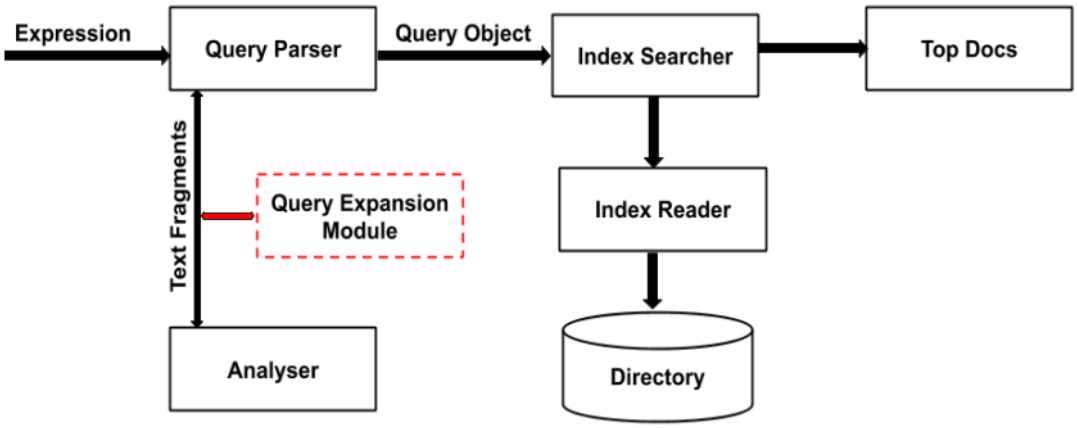
2. METHODOLOGY

In this section, we discuss about the architecture of the Document Retrieval system implemented using Apache Lucene. We will first describe its core functionalities. And some pre-processing operations as well as the evaluation process.

2.1. System Overview

Apache Lucene is an open-source library that provides search and indexing capabilities with high accuracy and performance. It is a java-based library and API that can easily be used to add search capabilities to the applications.

Fig1. Document retrieval system architecture



Generally, a Search engine performs all or a few of the following operations illustrated by the above Figure. Implementing it requires performing the following actions:

- Document Data: it is the first step. It consists in collecting the target contents used later to be queried in order to retrieve accurate documents.
- Analysing the document: In this step raw data is analysed and converted to a given format that can be easily understood and interpreted.

- Indexing the document: The main goal in this step is to index the documents so that the retrieval process will be based on certain keys instead of the entire content of the document.

The above pre-processing steps are performed in an offline mode. Once all the documents are indexed, users can conduct queries and retrieve documents. In this case, an object query is instantiated using a bag of words present in the searched text. Then, the index database is checked to get the relevant details. Returned references are shown to the user. Note that different weighting schemes can be used in order to index documents. The most used ones are tf-idf (the reference of the vectoral model) and BM25 (the reference of the probabilistic model). Typically, the tf-idf [11] [12] weight is composed by two terms: the first one measures how frequently a term occurs in a document. It computes the normalized Term Frequency (TF) which is the ratio of the number of times a word appears in a document by the total number of words in that document. the second term known as the inverse document frequency (IDF) measures how important a term is. It computes the ratio of the logarithm of the number of the documents by the number of documents where the specific term appears. BM25 [13] ranks a set of documents based on the query terms appearing in each document, regardless of the inter-relationship between the query terms within a document (e.g., their relative proximity). It is generally defined as follows:

Given a query Q, containing keywords q_1, ... , q_n the BM25 score of a document D is:

$$score(D,Q) = \sum_{i=1}^{n}(q_i).\frac{f(q_i,D).(k_1+1)}{f(q_i,D)+k_1.(1-b+b.\frac{|D|}{avgdl})}$$

where f(qi , D) is qi 's term frequency in the document D, D is the length of the document D in words, and avgdl is the average document length in the text collection from which documents are drawn. K_1 and b are free parameters, usually chosen, in absence of an advanced optimization, as $K_1 \in$ [1.2 , 2.0] and b = 0.75. IDF(q_i) is the IDF (inverse document frequency) weight of the query term q_i. It is usually computed as:

$$IDF(q_i) = Log\frac{N-n(q_i)+0.5}{n(q_i)+0.5}$$

2.2. Query Expansion using a Comparable Corpora and Word Embeddings

In order to improve system accuracy, two different techniques of query expansion can be used. The first one uses Wikipedia as comparable corpus. The second one uses word embeddings. The main purpose is to make the query more informative while reserving its integrity

2.2.1. Query Expansion using a Comparable Corpus

First, using Wikipedia as a comparable corpus to expand the short queries. For this purpose, two slightly different approaches can be used.

- Query expansion by summary: keywords can be extracted from the query using the Rake algorithm [25]; a domain-independent method for automatically extracting keywords. Keywords can be ranked based on their order of importance; and the most important one can be considered. Then it can be used to query Wikipedia. The data is summarized based on the first returned page; AKA, a short summary of one sentence, and it is concatenated it to the original query.
- Query expansion by content: In this the keywords can be extracted from the query using the Rake algorithm. And then the keywords are based on their order of importance. Then, the most important one is considered, and it will be used to query Wikipedia and the top returned pages are concatenated to the original query.

2.2.2. Query Expansion using Word Embeddings

Word embeddings are also used to expand the queries. We assume that the concept expressed by a given word can be strengthen by adding to the query the bag of words that usually co-occur with it. For this purpose, we can use the Genism implementation of word2vec using three different models: glove-twitter-25, glove-twitter-200, fasttext-wiki-news-subwords-300 and glove-wikigigaword-300 [26].

3. CONCLUSION

In this technology review, a DR system based on the Lucene toolkit is presented. Different weighting schema are discussed. General experiment results have proven that the probabilistic model (BM25) performs the vectoral one (TFIDF). Also, from experiments we can show that query expansion using word embeddings improves the overall system precision. Meanwhile, using a comparable corpus doesn't necessarily lead to the same result. This technical aspect can further be enhanced by:

- Testing an interactive query expansion technique: The query expansion using a comparable corpus may not lead to higher precision rates. The precision rate depends on the efficiency of the Rake key word extractor algorithm. The main idea is to let users validate the automatically extracted keywords used later during the query expansion process.
- Testing a hybrid technique of query expansion: Word embeddings can be applied on the result of the interactive query expansion phase. This may boost the system performance since the interactive query expansion will guarantee the use of significant words of the query. Also, using word embeddings will ensure retrieving relevant documents which do not necessarily contain words used in the query.

References:

[1] Anwar A. Alhenshiri, Web Information Retrieval and Search Engines Techniques,2010, Al-Satil journal, PP:55-92
[2] Fiana R, Oren K., Ranking Document Clusters Using Markov Random Fields, Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval, 2013
[3] Tombros A., Villa R., and van Rijsbergen C. The effectiveness of query-specific hierarchic clustering in information retrieval. Process. Manage., 38(4):559–582, 2002.
[4] Liu X. and. Croft W. B. Experiments on retrieval of optimal clusters. Technical Report IR-478, Canter for Intelligent Information Retrieval (CIIR), University of Massachusetts,2006
[5] Leuski A. Evaluating document clustering for interactive information retrieval. In Proc. of CIKM, pages 33–40, 2001.
[6] Liu X. and Croft W. B. Cluster-based retrieval using language models. In Proc. of SIGIR, pages 186– 193, 2004
[7] Liu X. and Croft W. B., Evaluating text representations for retrieval of the best group of documents. In Proc. of ECIR, pages 454–462, 2008

[8] Kurland O. and Lee L. Respect my authority! HITS without hyperlinks utilizing cluster-based language models. In Proc. Of SIGIR, pages 83–90, 2006.
[9] Kurland O. and Domshlak C., A rank-aggregation approach to searching for optimal query-specific clusters. In Proc. ofSIGIR, pages 547–554, 2008
[10] https://lucene.apache.org/core/
[11] Breitinger, C.; Gipp, B.; Langer, S. Research-paper recommender systems: aliterature survey. International Journal on Digital Libraries.2015. 17 (4): 305338.
[12] Hiemstra, Djoerd. A probabilistic justification for using tf×idf term weighting in information retrieval. International Journal on Digital Libraries 3.2 (2000): 131-139.
[13] Stephen E. R.; Steve W.; Susan J.; Micheline H-B. & Mike G. Okapi at TREC3. Proceedings of the Third Text Retrieval Conference (TREC 1994). Gaithersburg, USA.
[14] Stuart J-R, Wendy E-C. Vernon L-C. And Nicholas O-c, Rapid Automatic Keyword Extraction for Information Retrieval and Analysis, G06F17/30616 Selection or weighting of terms for indexing, USA, 2009
[15] Jeffrey P., Richard S., Christopher D-M, GloVe: Global Vectors for Word Representation.
[16] David M.W (2011). “Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation”. Journal of Machine Learning Technologies. 2 (1): 3763.
[17] https://trec.nist.gov/