Information Retrieval of Relevant Documents given a Query

Miguel Melchor

Abstract

This project focuses on Information Retrieval of Relevant documents retrieved from a set database given a user query or queries. The Snowball Stemming algorithm and an Inverted Files algorithm were used to simplify and speed up searching for retrieving documents. The results showed a fifty percent or higher precision, meaning that more than half of the documents returned were relevant. Results may vary and improve by changing the current output file threshold and by improving the gold standard.

Introduction

As technology improves and documents become digital, information retrieval systems that can search large databases fast and efficient have become more useful. Information Retrieval focuses on the structure, analysis, organization, storage and searching of information (Sec. Information Retrieval). Now a days, many libraries, business and universities use IR systems to provide fast access to all types of documents (Baeza Ch.1).

In order to provide the most relevant documents on a given search, keywords and frequencies should first be generated to improve searching speed. A number of techniques have been proposed and implemented for efficient information retrieval systems such as Signature Files (Baeza). Signature Files are based on the idea of the inexact filter. Using hash-coded bit

patterns, each file generates a signature, which is then saved into a signature file. When searching a query, this signature file is scanned to determined relevant files. The Signature File algorithm provides a quick search and are good with dynamic databases but are slow with large databases (Baeza Ch.4), which in turn leads us to look for an algorithm that is just as good but works well with large databases.

The best known and most widely use technique for information retrieval is the Inverted File (Sec. Information Retrieval). An Inverted File is used to store keywords or some content identifier to the corresponding set of documents. It has been implemented using sorted arrays, Binary Trees, Tries and other data structures. This paper implements an Inverted File using Hashing as proposed by Baeza Ch.13.

Method

In order to speed up searching of multiple keywords, each document was processed to remove all unnecessary information such as special symbols, punctuation and new line characters. The reason for doing this is the following, when searching a keyword such as “time”; the results might not be as accurate because each occurrence of “time” attached to some punctuation will not be counted towards the frequency of “time” in a given document, thus removing punctuation will solve this problem. This was also done to facilitate the creation of bigrams for phrase searching (2 words). After cleaning each file, every word in the document was stemmed using the Snowball Stemmer provided by the NLTK Package in Python. Stemming was used to improve retrieval effectiveness and to reduce the size of the hash table later created (Figure 1).

Term	Stem
Engineering	Engineer
Engineered	Engineer
Engineer	Engineer

Figure 1.

To implement the Inverted File, the processed documents, outputted by the cleaning algorithm, were used. A hash table was created with all the words in each of the documents where the keys in the table were the alphabet letters from a-z and numbers 0-9. Each word was placed in the hash table according to their first letter or number. Similarly, a hash table of bigrams was also created and each bigram was added to the table using the first letter of the first appearing word in the tuple. In other words, the hash table was simply a dictionary of dictionaries of list where every keyword, in both the unigram and bigram, was linked to each document in which it appeared along with the frequency of that keyword in that given document.

After creating the Inverted File, the searching time increased. Original and Processed Documents were no longer needed for searching since it only required to search a small dictionary and list now.

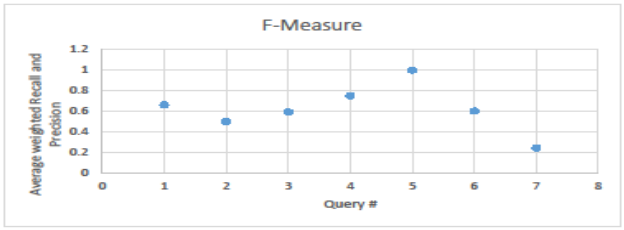
The original data set used to test this algorithm, was obtained from Project Gutenberg. A set of books from different genres were used and reviewed to create a Gold Standard for the expected results of a given query as suggested by Prof. Joydeep (Prof. Joydeep).

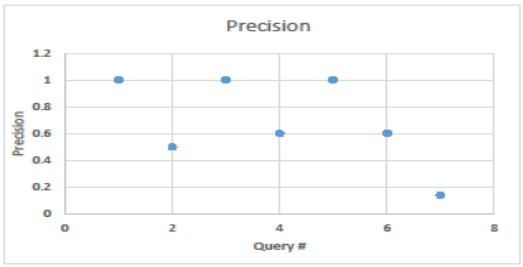
Results

Using 8 to 10 given queries, a Gold Standard was created to use for the evaluation of retrieved files. The number of retrieved documents was determined by a static threshold set to half of the

number of occurrences of the first highest file containing a given keyword. Due to the low number of queries use to create the Gold Standard, the results were not as good as expected. The recall and precision of each search was calculated and using both, the F-measure was calculated (Figure 2). The results showed that my implementation of Inverted Files using hashing returned more than 50% of relevant documents.

Recall	Precision	F-Measure (2PR/P+R)
0.5	1	0.66
0.5	0.5	0.5
0.42	1	0.59
1	0.6	0.75
1	1	1
0.6	0.6	0.6
1	0.14	0.24





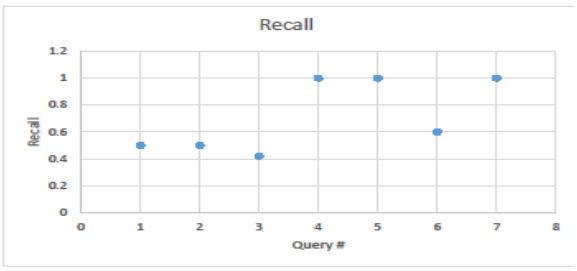


Figure 2.

Conclusion

The results of the Inverted File with hashing show that more than 50% of relevant documents are being returned. According to my results, this algorithm tends to work better when searching for a keyword in which the documents that are relevant have a significant difference from all other documents, in terms of genre. For example, if most of the data set consist of poetry or romance, and the term being search is “computer” and the number of relevant computer documents is low

compare to the others, then the precision and recall are more likely to be higher. This is due to the big difference in topics. Since the frequency of the keyword “computer” is likely to be zero in all other documents and the number of relevant documents is low, then only files containing that keyword are going to be returned. To improve results in the future, a Gold Standard with more exhaustively label relevant documents for each query must be created.

Bibliography

Raghavan, Prabhakar, and Hinrich Schütze. "Chapter 1.1,2.2,3,4." Introduction to Information Retrieval. By Christopher D. Manning. New York: Cambridge UP, 2008. N. pag. Print.

Baeza-Yates, R. "Chapters 1-4,8,10,13." Information Retrieval: Data Structures & Algorithms. By William B. Frakes. Englewood Cliffs, NJ: Prentice Hall, 1992. N. pag. Print.

“Sec. Information Retrieval” File Organization and Search Strategies. Print

Kak, Avinash. "Section 1-2." Evaluating Information Retrieval Algorithms with Significance Testing Based on Randomization and Student’s Paired T-Test. N.p.: Purdue U, 2013. N. pag. Print.

"Evaluation in Information Retrieval." Http://nlp.stanford.edu. Cambridge UP, 1 Apr. 2008. Web. 25 Apr. 2014.

Prof. Joydeep Ghosh (UT ECE). “Performance Evaluation of Information Retrieval Systems.” University of Science and Tech, Hong Kong. PowerPoint.