**CS6200 – Information Retrieval, Spring 2017**

**Project Report**

**Team Members:**

**Rakesh Krishna Radhakrishnan**

**Siddharth Talwekar**

**Vivek Nair**

**Instructor:**

**Nada Naji**

# **Introduction**

The goal of this project is to build information retrieval systems using the core concepts and processes of information retrieval learnt throughout the course in this semester. The project scope also comprises of performing optimization techniques such as stopping, stemming, query expansion as well as evaluation of different systems using evaluation metrics such as Precision, Recall, MAP, MRR and P@K.

## **Dataset**

The CACM test-collection containing 3204 raw HTML documents has been used as the corpus along with 64 unprocessed queries, stemmed version of raw documents, corresponding stemmed queries, relevance information for the queries and a list of the most frequent words in the corpus which is used as the stoplist.

This report describes the implementation along with the design choices made during the project implementation.

## **Member’s Contribution**

Rakesh Krishna Radhakrishnan:

Siddharth Talwekar:

Vivek Nair:

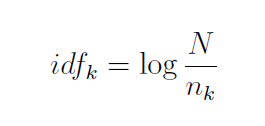
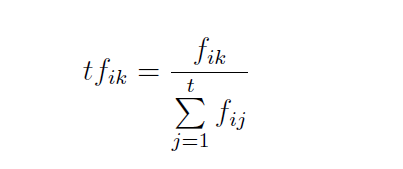
# **Literature and Resources**

Search Engines: Information Retrieval in Practice by Bruce Croft, Donald Metzler and Trevor Strohman

## **Retrieval Models**

TF.IDF (Term frequency – inverse document frequency)

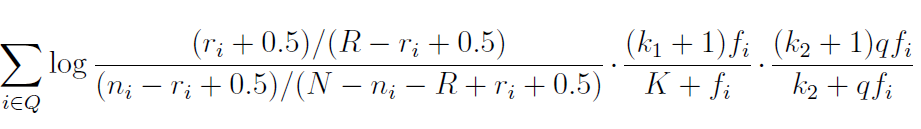
It’s a statistical measure which is used to evaluate the importance of a word in a document. This value increases proportionally to the number of times a word appears in a document but is normalized by the frequency of the word in the collection to adjust for the unimportant words appearing frequently. The formula [] for this is given by



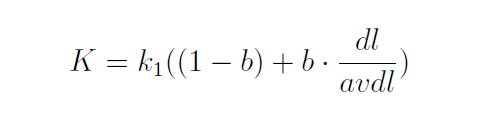
Where *tf* is the term frequency calculated by dividing the frequency of the term *k* over the number of terms appearing in the document and *idf* is calculated by taking the log of number of documents over the number of documents containing the term *k*. We get the value of TF.IDF by multiplying the above terms.

BM25

BM25 is a popular and effective retrieval model based on the binary independence model, the formula[] for which is given



* *k1*, *k2* are constants
* *qfi* frequency of term *i* in the query
* *ri* is the number of relevant documents containing term *i*
* *ni* is the number of documents containing term *i*
* *R* is the number of relevant documents for this query
* *N* is the total number of documents in the collection
* *fi* is the document frequency of the term
* *K* is given by the following formula



* *b* is a parameter
* *dl* is the length of the document
* *avdl* is the average length of a document in the collection

Lucene

Lucene is a popular java based search engine used for a wide range of commercial applications. An open source information retrieval library[], Lucene is widely used in academic and commercial search engine application. Solr and ElasticSearch are both based on Lucene libraries. By default, Lucene uses a TF.IDF similarity for scoring.

Query Expansion

Query expansion is a technique used to add new terms into the query which are contextually relevant to the query. There are various methods through which we can implement query expansion like using Thesauri, Pseudo Relevance Feedback, Inflectional and/or derivational variants. In this project, we have used Pseudo Relevance Feedback. In Pseudo Relevance Feedback, the most frequent words barring the stopwords in the top N documents are assumed to be relevant to the query. In our design, the Top 20 frequent words from the top 50 documents are added to the query and results are then retrieved with the new expanded query.

Stopping

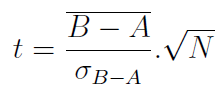
Stopping is a technique used to remove very high frequency words from corpus/query, which are generally not topically relevant and are function words. In this project, we have performed stopping on the corpus as well as the queries using the given list of common words.

Stemming

Stemming is a technique of using stems of words to expand the possibility of finding the multiple variances of a single word. We have used the stemmed corpus and queries provided for the project with TF.IDF and BM25 retrieval models.

T-tests:

T-test is a statistical test used to evaluate the performance of 2 different search engines. We have used T-test to compare the effectiveness of search engine models built in this project. In T-tests, performance score of two systems is determined by the following formula[]



* A and B are the average precisions of 2 systems to be tested
*  is the standard deviation of the differences.
* N is the number of queries.

Snippet Generation

A snippet is a short summary of a document. The snippet is supposed to present a brief description of what the document contains and how it is relevant to the information seeker by highlighting the words which are significant or are topically or contextually related to the query. We have used the *Luhn’s Approach*[2] for generating snippets for the top ranked documents for each query in our project.

**Implementation and Discussion**

We have implemented 9 variants of a search engine in total. Table 1 gives a short summary on the different variants implemented.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Task** | **System** | **Retrieval Model** | **Stemming** | **Stopping** | **Query Expansion** | **Relevance information**  **Used?** |
| 1 | 1 | TFIDF | No | No | No | NA |
| 2 | BM25 | No | No | No | Yes |
| 3 | Lucene | No | No | No | NA |
| 2 | 4 | TFIDF | No | Yes | Pseudo Relevance | NA |
| 5 | BM25 | No | Yes | Pseudo Relevance | No |
| 3 | 6 | TFIDF | No | Yes | No | NA |
| 7 | BM25 | No | Yes | No | Yes |
| 8 | TFIDF | Yes | No | No | NA |
| 9 | BM25 | Yes | No | No | Yes |

**Parsing**

Parsing is the first step in which the raw HTML documents are parsed and tokenized to be used in the indexing stage. During this stage, we have removed all punctuation from the text in the documents except the “-“ and also preserved “.” & “,” if they are between numbers. Also the tokenized text is then case-folded and saved to disk to be used in the indexing stage.

**Indexing**

During indexing, using the tokenized documents, we have built a unigram index which is a dictionary like structure, in which a word is a key and its corresponding value is a list of documents in which the word appears. The frequency of the word in each document is also saved in the index. This index is then used in the retrieval models to find relevant documents for a query. In systems 4,5,6,7 we have used stopping in index creation as well at query time.

**Retrieval Model**

In this project, we have implemented three retrieval models, namely BM25, TF.IDF and Lucene.

TF.IDF

BM25

Lucene

Stopping

Stemming

Query Expansion

# **Results**

The values of the evaluation results (MAP and MRR) for all the 7 systems are presented in Table 2:

The results for precision, recall, P@5 and P@20 are presented in the below files:

Table 3 shows the result of the T-tests:

**Query by Query Analysis**

Below is a query by query analysis of first 3 stemmed queries.

1. Query: portabl oper system

Top 10 documents retrieved by system 7:

1. Query: code optim for space effici

Top 10 documents retrieved by system 7:

1. Query: parallel processor in inform retriev

Top 10 documents retrieved by system 7:

# **Conclusions and Outlook**

# **Bibliography**

[1]

[2]

[3]

[4]