Assignment 1

CA6005

Viraj Pawar
School of Computing
Dublin City University
Dublin, Ireland
viraj.pawar2@email.dcu.ie

ABSTRACT

This work presents an implementation of three retrieval models: the Vector Space Model, BM25 and Query likelihood model with Laplace smoothing. The models are run Cranfield collection, in which there are 1400 documents and 225 associated queries. The document collection was preprocessed, indexed, and queried over the queries for ranking using the Vector Space Model, BM25 and Query likelihood model with Laplace smoothing retrieval models. The retrieval model output containing the relevance of each query and document pair IDs are saved in a text file and evaluated using the TREC evaluation programme. The evaluation output is saved in the evaluation folder in the code repository. The link to the code is: https://github.com/virajpwr/CA6005I-assignment-1.git

KEYWORDS

Vector Space Model, BM25, Query Likelihood, tf-idf, posting, IR.

1 Introduction

The Cranfield collection is a collection of 1400 documents containing abstracts of aerodynamics journal articles and a set of 225 associated queries for precise quantitative measures of information retrieval (IR) effectiveness [1].

A simple IR system is implemented by indexing the Cranfield collection documents using three different IR models: Vector Space Model, Okapi BM25, and it's effectiveness in retrieving relevant documents based on the query is tested.

The Cranfield document collection is structured in XML encoded machine-readable format containing various tags for docno, title, author, bib and text. These tags encapsulate different information about the document. The most relevant and useful information used in the implemented IR system is doc no (doc id) and text containing the main body text of the document. The main body text contains a mixture of punctuation, words and numbers.

The queries in the Cranfield collection are also structured in XML encoded machine-readable format and contain tags containing information of num (query id) and title. The title contains a mixture of words, punctuation and numbers.

Given a query with it's ambiguity, the task of the information retrieval system is to find the most relevant documents.

1.1 Architecture

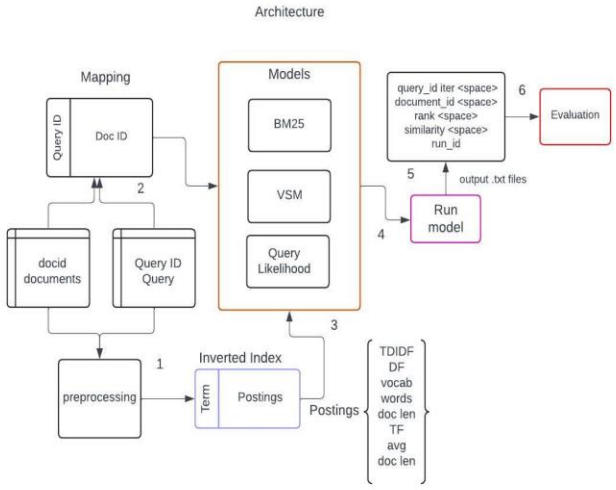

Figure 1: Architecture of the implementation of IR system

The architecture of the implementation of the IR system shown in Fig 1 contains a reading Cranfield collection block, text preprocessing block, query ID and doc ID mapping block, inverted index with posting, IR models and evaluation of the models.

1.1.1 Reading documents and queries: The Cranfield documents and queries are read, and split tags are used to separate the documents and queries creating a list of documents and queries. A dictionary of document ID and corresponding text is created as a key-value pair. Similarly, a dictionary of query ID and query (title) is created as a key-value pair

1.1.2 Mapping query ID and document ID: In mapping, each query ID is mapped to the list of all document IDs which is used for finding the relevant document for a query.

1.1.3 Preprocessing: The text is preprocessed to remove noise by removing irregular characters, normalize and improve relevance of the text. In preprocessing step the text is tokenised, the numbers are converted to words, normalised by removing punctuation and lowercasing, reducing the words to its root form by stemming and removing stop words to reduce noise in the text.

1.1.4 Inverted Index: The inverted index with posting list is created.

1.1.5 Models: BM25, VSM and Query likelihood models were used in the implementation of the IR system.

1.1.6 Evaluation: The output from the models is saved in text file with "query_id iter <space> document_id <space> rank <space> similarity <space> run_id" The models were evaluated for MAP, P@5, and NDCG using TREC evaluation programme.

2 Indexing

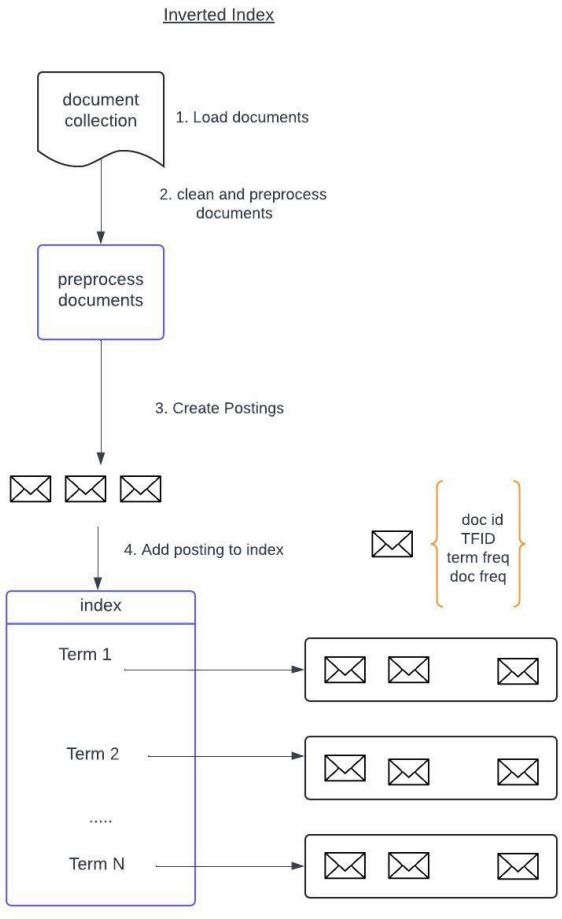

Figure 2: Inverted Index construction

The index is implemented using a key-value pair of term and postings list. The postings list contains the document ID, the term frequency and the tf-idf of the mapped term. Other information such document frequency of the term is also added to the inverted index. Other attributes such as word frequency, average document length, vocabulary count, and word count can also be accessed from the inverted index class.

3 Ranking

3.1 Vector Space

In the vector space model, the documents and queries are represented as sparse vectors where Term-Frequency times Inverse-Document-Frequency is used as a weighting scheme [2]. The term-frequency is within document frequency, and the inverse document frequency is the negative log of the document frequency [2].

$$tf\text{-}idf_{i,j} = tf_{i,j} \times idf_{i,j}$$

The vectors for documents and queries are initialised with zeros with the size of the vocabulary. The tf-idf values for document vocabulary are pulled from the postings. The tf-idf for the queries is calculated. The document relevance is calculated by calculating cosine similarity between the tf-idf vector representations of the documents and queries. The cosine similarity is given by

$$\cos(\angle(\vec{d},\vec{q})) := \frac{\vec{d}\cdot\vec{q}}{\sqrt{\vec{d}^2}\cdot\sqrt{\vec{q}^2}}$$

3.3 BM25

BM25 is a probabilistic model where the search results are order based on the probability of the relevance to the query. The IDF of query is calculated using

$$\mathrm{IDF}(q_i) = \log\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5},$$

The BM25 score is calculated using the formula:

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

The constants k1 = 1.2 where k = [1.2, 2.0] and b = 0.75 [2].

3.4 Query Likelihood with Laplace smoothing

The language model used is Query likelihood with Laplace smoothing was used. The query likelihood is a probabilistic model where the probability is query for a document is given by $P(d|q)/P(d)$.

4 Evaluation

The model is passed with mapping data of query id and document id, and the inverted index. For each query the model ranks the documents and return relevance score for each document for a query id. The result is stored in dictionary where each query id has a corresponding relevant document ids, score for each document id and ranks

The table below shows the evaluation of the three models.

	VSM	BM25	LM
P@5	0.0093	0.0079	0.0093
MAP	0.0120	0.0111	0.010
NDCG	0.1464	0.1450	0.1450

Table 1: IR model evaluation

REFERENCES

[1] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. Introduction to Information Retrieval. Cambridge University Press, USA.

[2] Roelleke, T., 2022. Information Retrieval Models: Foundations & Relationships. Springer Nature.