Information Retrieval Model Evaluation: A Case Study on Cranfield Data

Ayesha Ishrath
Dublin City University
Dublin, Ireland **Student Number: 14224**
ayeshaishrath2@mail.dcu.ie

Abstract

This paper presents the architecture, development, and testing of a sophisticated information retrieval system to index, process, and retrieve documents from the Cranfield dataset. The system integrates three standalone ranking models for document retrieval: the Vector Space Model (VSM), BM25, and Language Model (LM). All the models are coded to rank documents based on relevance to search queries using different mathematical and probabilistic methods.

To ensure a comprehensive assessment of the system's retrieval performance, performance is evaluated with TREC Eval, which is a standard evaluation mechanism in information retrieval. The experimental results provide us with valuable information about the relative strengths and weaknesses of every ranking model. While BM25 is generally supposed to be a good relevance ranking baseline, the Vector Space Model is a simple-to-grasp algebraic framework, and the Language Model has probabilistic modeling for more context-sensitive retrieval. There are significant differences in retrieval performance among these models for various queries.

In addition, the report discusses the overall practical implications of these findings for real-world information retrieval applications, with a particular emphasis on ranking effectiveness versus computational complexity trade-offs. The results contribute to better understanding how different ranking mechanisms influence retrieval performance and provide a foundation for future search technology development.

1 Introduction

Information Retrieval (IR) is one of the fundamental tasks of modern search engines and other digital systems that enable users to effectively get back pertinent documents from huge collections of documents. The efficiency of an IR system is quantified in terms of its ability to fetch documents most relevant to a specific query by the user and least non-relevant documents. To this end, retrieval systems employ a combination of document representation, query processing, ranking models, and evaluation measures.

In this project, we implement and test an information retrieval system on the Cranfield dataset, a model benchmark dataset widely used in IR research. The dataset contains structured document-query relevance judgments and is therefore suitable for testing retrieval effectiveness. The goal of this study is to compare the performance of different retrieval models in terms of ranking documents by relevance to the queries of users. By taking different ranking models and evaluation strategies, we aim to identify the strengths and limitations of each approach in terms of retrieval effectiveness.

The system architecture follows a structured pipeline for a systematic method of document processing, retrieval, and evaluation.

2 Indexing

Indexing in Information Retrieval System (IRS) is necessary to implement efficient and rapid data retrieval. It enhances search speed, accuracy, and storage space employing structures like inverted indexes by indexing terms and their document mappings. Indexing enables Boolean searching, ranking computation, and support for dealing with big-data along with functionalities like fuzzy matching, stemming, and synonym expansion. It also implements relevance ranking

and saves the cost of storage via compression techniques. Without indexing, it would be slow and inefficient to search large amounts of data and therefore an essential ingredient of contemporary search engines and databases. Indexing enhances the efficiency and accuracy of document retrieval. Our approach follows these steps:

2.1 Text Preprocessing

Raw text content is preprocessed prior to indexing to normalize and sanitize the content. Preprocessing of text is primarily done to enhance retrieval precision by removing noise and ensuring consistency in text representation. The following operations are employed:

- **Removal of Non-Alphanumeric Characters:** Punctuation marks, special characters, and numeric data (if not crucial for retrieval) are removed to retain only meaningful words.
- **Stopword Filtering:** It removes common words such as "the," "is," "and," and "of" to conserve index space and increase efficiency since they do not contribute significantly to search relevance.
- **Lemmatization:** Lemmatization transforms words to their base or dictionary form (for example, "running" is mapped to "run") so that different inflected forms of a word are translated to a single term.
- **Lowercasing:** All text is lowercased to eliminate case sensitivity so that indexing can be consistent.

2.2 Tokenization

Tokenization is the process of splitting text into individual words or terms (tokens) that can be indexed. Tokenization provides documents as lists of words in a controlled form. Tokenization is carried out by:

- **Word splitting:** A document is split into a list of tokens by whitespace and punctuation-based tokenization.
- **Compound word handling and hyphenation:** Contractions and hyphenated words are processed to appear consistently in terms.
- **Term normalization:** Term variations frequently appearing, i.e., plural forms, are converted to one form for index consistency.

2.3 Inverted Index Construction

The inverted index is the main data structure for efficient document retrieval. Instead of searching all documents at query time, an inverted index allows the system to retrieve documents that contain a particular term very efficiently. Its construction is the process of:

- **Term Mapping:**Mapping each unique term to the collection of documents that it occurs in.
- **Positional Indexing:** Optionally, the position of a term in a document is indexed to enable proximity and phrase querying.
- **Frequency Information:** The frequency of occurrence of a term in a document (term frequency) is indexed to support frequency-based ranking models and score calculation.

With this indexing method, the system offers effective query execution and reduced computation overhead, enabling the overall effectiveness of the retrieval process. The inverted index is employed as the foundation of the retrieval system for quick access to relevant documents based on search queries.

3 Ranking Models

Ranking in an Information Retrieval System (IRS) needs to rank the search results according to relevance so that the users get a clear idea of the most relevant information. Ranking is biased towards the relevant documents, improving search accuracy, efficiency, and user satisfaction. Ranking in big systems navigates through billions of documents on the basis of algorithms like

BM25, TF-IDF, and PageRank, and advanced techniques utilize machine learning and semantic perception. Several distinct ranking methods, such as lexical and semantic ranking, are applied with specific types of queries in trying to achieve the greatest possible accuracy within the outcomes. Ranking is a very important factor in e-commerce, advertising, and recommendation systems that contributes a lot towards impacting visibility and business performance. Without ranking, search results would be meaningless and disordered, and information retrieval would be laborious and unfruitful. Three different ranking models were implemented:

3.1 Vector Space Model (VSM)

The Vector Space Model (VSM) is an Information Retrieval (IR) model commonly used for ranking and document retrieval, according to the relevance of a document with respect to a user-provided query. Documents and queries are represented as vectors in a high-dimensional space, one dimension for each unique term (word) of the corpus. The similarity between a document and a query is then calculated through TF-IDF weighting and cosine similarity. This model was the first to be utilized model for information retrieval systems.

- **TF-IDF:**
  - **Term Frequency (TF):**This measures how often a term appears in a document

$$TFt, d = \frac{\text{Number of times term } t \text{ appears in document } d}{\text{Total terms in document } d} \quad (1)$$

    Words that appear more frequently in a document get higher importance.
  - **Inverse Document Frequency (IDF):**Reduces the weight of common terms that appear in many documents and boosts the weight of rare terms.

$$IDFt = \log\left(\frac{N}{DFt}\right) \quad (2)$$

    Where:
    - N = Total number of documents in the collection
    - DFt = Number of documents containing term t
  - **TF-IDF Score Calculation:** Combining both TF and IDF:

$$TF\text{-}IDFt, d = TFt, d \times IDFt \quad (3)$$

    This ensures important words in a document are given high weights, while frequent but less meaningful words (e.g., "the", "is", "of") are down-weighted.
- **Cosine Similarity for Measuring Relevance:** Once documents and queries are represented as TF-IDF weighted vectors, we compute cosine similarity to measure their similarity. This determines how closely a document matches a query. Cosine Similarity Formula:

$$\cos\theta = \frac{D \cdot Q}{\|D\| \times \|Q\|} \quad (4)$$

  Where:
  - D = Document vector (TF-IDF weights of terms in the document)
  - Q = Query vector (TF-IDF weights of terms in the query)
  - $D \cdot Q$ = Dot product of the document and query vectors
  - $\|D\|$ and $\|Q\|$ = Magnitudes (lengths) of the vectors

  Cosine similarity normalizes document length, provides relevance scores between 0 and 1, and effectively handles sparse data in information retrieval.

[6]

3.2 BM25 Ranking

BM25 improves upon VSM by introducing term frequency normalization and length correction parameters (k_1, b). BM25+ (Best Matching 25 Plus) is an extension of the BM25 ranking function used in information retrieval systems to rank documents based on their relevance to a query. BM25+ was introduced to address a

known limitation of BM25—its tendency to overly penalize long documents.

BM25 is a probabilistic ranking function based on the Okapi BM25 model, which scores documents using term frequency (TF) and inverse document frequency (IDF), along with document length normalization. The formula is:

$$BM25D, Q = \sum_{t \in Q} IDFt \cdot \frac{ft, D \cdot k_1 \ 1}{ft, D \ k_1 \cdot \left(1 - b \ b \cdot \frac{|D|}{avgD}\right)} \tag{5}$$

Where:

- ft, D is the term frequency of term t in document D.
- $|D|$ is the length of document D.
- $avgD$ is the average document length in the collection.
- k_1 and b are hyperparameters that control term saturation and length normalization.
- $IDFt$ is the inverse document frequency of term t.

A problem with BM25 is that it can over-penalize long documents since the length normalization term is applied multiplicatively.

BM25+ modifies BM25 to reduce the impact of document length on ranking scores. The BM25+ formula is:

$$BM25D, Q = \sum_{t \in Q} IDFt \cdot \frac{ft, D \cdot k_1 \ 1}{ft, D \ k_1 \cdot \left(1 - b \ b \cdot \frac{|D|}{avgD}\right) \ \delta} \tag{6}$$

Where:

- δ is a new positive parameter that ensures long documents are not overly penalized.

[7].

3.3 Language Model (LM)

The language model implemented in this project is a Unigram Language Model with Dirichlet Smoothing. Below is an explanation of how it works and the key aspects of its implementation.

A unigram language model assumes that all the words in a document appear independently. The probability of a query given a document is then simply the product of the probabilities of each term appearing in the document separately.

Raw term probabilities, on the other hand, suffer from zero probability issues—a term's probability is zero if the term does not appear in a document, and this leads to erroneous scoring.

To fix this, Dirichlet Smoothing is applied, which interpolates the word probability by incorporating global collection statistics (i.e., frequency of words across the entire collection).

The probability of a query $Q = w_1, w_2, \ldots, w_n$ given a document D is:

$$PQ \mid D = \prod_{w \in Q} Pw \mid D \tag{7}$$

Since probabilities are very small, we take the log form to avoid numerical underflow:

$$\log PQ \mid D = \sum_{w \in Q} \log Pw \mid D \tag{8}$$

The Dirichlet Smoothing formula for a term w is:

$$Pw \mid D = \frac{fw, D \ \mu Pw \mid C}{|D| \ \mu} \tag{9}$$

Where:

- fw, D = Frequency of term w in document D.
- $|D|$ = Total number of words in document D.
- μ = Dirichlet smoothing parameter (controls the balance between document statistics and collection statistics, typically set around 2000).
- $Pw \mid C = \frac{fw,C}{|C|}$ = Probability of term w in the entire collection (corpus).
- $|C|$ = Total number of words in the entire corpus.

Intuition of Dirichlet Smoothing:

- If a term appears frequently in the document, it gets a high probability.
- If a term does not appear in the document, it still gets a nonzero probability based on its overall frequency in the entire collection.

[8].

4 Evaluation

In this project we have used the TREC (Text REtrieval Conference) Eval for evaluating information retrieval (IR) models because it provides a standardized, reliable, and widely accepted framework for assessing the effectiveness of search and ranking algorithms.

The performance of the above mentioned ranking models was assessed using TREC Eval with the following metrics:

Model	MAP	P@5	NDCG
VSM	0.1092	0.1440	0.2485
BM25	0.1100	0.1529	0.2477
LM	0.0846	0.1191	0.2099

Table 1: Performance Metrics for Ranking Models

5 Analysis of Results

BM25+ handled retrievals better as it effectively balances term frequency, document length normalization, and IDF weighting so that rare but significant terms give meaningful contribution. It also employs non-linear TF scaling and a delta smoothing factor to enhance retrieval quality.

The Language Model (LM) was worse, as opposed to VSM and BM25 because it relies only on term frequency and collection probability, and not on IDF weighting. It is therefore more susceptible to frequent terms, and longer documents can dominate scores and yield worse rankings.

VSM performed well but lacked length normalization, leading to weaker scores. LM was the weakest, as it does not adjust for document length biases.

6 Discussion and Limitations

6.1 Key Observations

- BM25 outperformed other models due to its length-normalization features.
- VSM performed well but struggled with longer documents.
- LM was least effective due to its simplistic probability estimation.

6.2 Limitations and Enhancements

Potential enhancements to the system include:

- Integrating neural ranking models (e.g., BERT, GPT-3) for improved contextual relevance.
- Implementing query expansion techniques for better recall.
- Optimizing indexing strategies for large-scale datasets.

7 Conclusion and Future Work

This study implemented and compared three ranking models for information retrieval. BM25 demonstrated the highest effectiveness, while VSM and LM had respective trade-offs in performance. Future work involves leveraging deep learning-based ranking techniques and improving indexing methodologies.

8 Dataset and Code

For the above project implementation, the Cranfield dataset from this repository was used - **GitHub Repository containing the dataset**.

The code implementation of the models is available here - **Code**.

References

[1] Andrew Trotman, Antti Puurula, and Blake Burgess. *Improvements to BM25 and Language Models Examined*. Proceedings of the 2014 Australasian Document Computing Symposium, 2014. Available at: https://www.researchgate.net.

[2] Hideo Joho and Leif Azzopardi. *A Comparison between Term-Independence Retrieval Models for Ad Hoc Retrieval*. ACM Transactions on Information Systems (TOIS), 2012. Available at: https://dl.acm.org.

[3] François Rousseau and Michalis Vazirgiannis. *Graph-of-Word and TW-IDF: New Approach to Ad Hoc Information Retrieval*. Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM), 2013. Available at: https://lintool.github.io.

[4] José Devezas and Sérgio Nunes. *Latent Dirichlet Allocation Complement in the Vector Space Model for Text Classification*. Computación y Sistemas, 2018. Available at: https://josedevezas.com.

[5] YouTube Video. *BM25 and Language Models Explained*. Available at: https://www.youtube.com/watch?v=y_AGY6AHDP8&list=PLz_RRnOnUTWETaBcpAMOd1evVIT3t8NSH.

[6] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. Available at: https://nlp.stanford.edu/IR-book/.

[7] Stephen Robertson and Hugo Zaragoza. *The Probabilistic Relevance Framework: BM25 and Beyond*. Foundations and Trends in Information Retrieval, 3(4), pages 333–389, 2009. Available at: https://www.nowpublishers.com/article/Details/INR-019.

[8] ChengXiang Zhai and John Lafferty. *A Study of Smoothing Methods for Language Models Applied to Information Retrieval*. ACM Transactions on Information Systems (TOIS), 22(2), pages 179–214, 2004. Available at: https://dl.acm.org/doi/10.1145/984321.984322.