Natural Language Processing Project

Aditya Viraj Rao Ponugoti CS20B005
Venkata Sesha Haricharan Korrapati CS20B086
Kakunuri Shashank Reddy CS21B038

1 Abstract

Our prior Information Retrieval (IR) system depended on the TF-IDF method for document indexing and cosine similarity for judging query relevance with documents. We measured the effectiveness of this system on the Cranfield dataset through various precision-recall and ranking-based metrics. To enhance our system, we explored unigram, bigram, and hybrid models, while also addressing challenges like vector sparsity, synonymy through query expansion, and incorporating document titles for improved retrieval.

2 Introduction

Our current IR system relies on TF-IDF-based indexing, treating documents and vectors as bags of words. While it does a fair job, there are times when no relevant documents are found in the top 10 results. To overcome this, we aim to adopt more advanced word and sentence representation techniques. However, our current model faces several challenges: high-dimensional sparse vectors leading to low cosine scores, reduced recall due to synonym usage in queries and documents, and decreased precision stemming from the bag-of-words representation. In this project, we concentrate on addressing these issues. Initially, we identify the problems in TF-IDF indexing, hypothesize the cases, and propose potential solutions.

3 Dataset

The project utilizes the Cranfield dataset, consisting of documents on automobile aerodynamics. With 1400 documents and 225 queries provided, human relevance scores accompany the queries, aiding in system evaluation. Reference documents are categorized based on their relevance degree, facilitating performance assessment through metrics like nDCG. The positions of reference documents in cran qrels.json represent the following judgments: 1 - Complete answer, 2 - High relevance, critical for research, 3 - Useful background or method suggestions, and 4 - Minimum interest, e.g., historical perspective.

4 Evaluation Metrics

Let's explore the evaluation measures used to assess the effectiveness of the IR systems we're going to develop in the next sections. These measures indicate whether

the search results are satisfactory or not. We categorize them into online metrics, which analyze user behavior, and offline metrics, which assess result relevance. They provide insights into the system's performance and areas for improvement.

Precision

Precision in binary classification is calculated as the ratio of the number of relevant documents retrieved to the total number of documents retrieved.

$$\text{Precision} = \frac{|\text{Retrieved} \cap \text{Relevant}|}{|\text{Retrieved}|}$$

Recall

Recall in binary classification is calculated as the ratio of the number of relevant documents retrieved to the total number of relevant documents in the dataset.

$$\text{Recall} = \frac{|\text{Retrieved} \cap \text{Relevant}|}{|\text{Relevant}|}$$

F1 Score

The F1 Score in binary classification is the harmonic mean of precision and recall. It balances both metrics and is calculated using the formula:

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Mean Average Precision (MAP)

Mean Average Precision (MAP) is a measure of the average precision of the relevant documents retrieved across all queries. It evaluates the quality of the ranking produced by an information retrieval system:

$$\text{MAP} = \frac{\sum_{q=1}^{Q} \text{AP}(q)}{Q}$$

Where $\text{AP}(q)$ represents the average precision for query q, and Q is the total number of queries. Precision is calculated at each position in the ranked list of retrieved documents, and AP is computed as the average of these precision values for all relevant documents.

Normalized Discounted Cumulative Gain (nDCG)

Normalized Discounted Cumulative Gain (nDCG) assesses the effectiveness of a search engine's ranking by considering both the relevance and position of the retrieved documents. It normalizes the Discounted Cumulative Gain (DCG) value by dividing it by the ideal DCG:

$$\text{nDCG} = \frac{\text{DCG}}{\text{iDCG}}$$

Where DCG is the Discounted Cumulative Gain and iDCG is the ideal DCG.

5 Vector Space Model

The Vector Space Model (VSM) is a widely used method in natural language processing for representing text as a collection of features or words, regardless of their order or grammar. It's a straightforward approach for extracting document features, focusing on word occurrences and ignoring word order and meaning relationships. Under this model, texts of varying lengths are converted into fixed-length vectors. The process involves preprocessing the text to eliminate irrelevant details and tokenize the text into individual words. Commonly occurring but less meaningful words, known as stop words, are typically removed to enhance analysis accuracy. Stemming reduces words to their base forms, while lemmatization transforms words to their dictionary forms. The TF-IDF matrix is computed, incorporating term frequency and inverted document frequency, with additional smoothing. The same process is applied to a query. Cosine similarity between the query and TF-IDF matrix is calculated, arranging documents by decreasing similarity. Higher cosine similarity indicates greater resemblance between the document and query. Due to its sparsity, computing the TF-IDF matrix requires significant computational resources. Latent Semantic Analysis (LSA) was attempted to derive meaningful features from the Cranfield dataset, addressing this computational challenge.

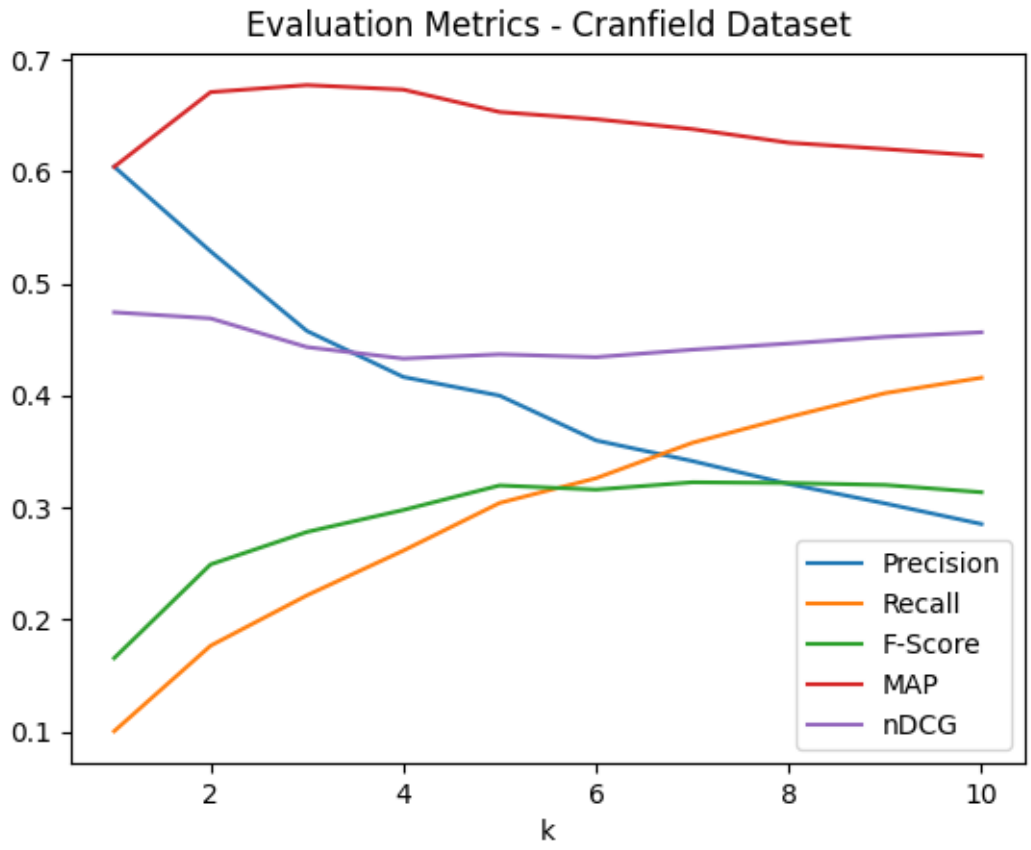

Figure 1: Evaluation Metrics for Vector Space Model on the Cranfield Dataset

6 Latent Semantic Analysis

LSA, or Latent Semantic Analysis, is a statistical technique employed in natural language processing and information retrieval to uncover hidden semantic relationships between words and documents. It utilizes Singular Value Decomposition (SVD) to represent words and documents in a lower-dimensional space, where each dimension corresponds to a latent concept or topic.

The fundamental concept behind LSA is that words appearing in similar contexts often possess similar meanings. By analyzing co-occurrence patterns in extensive text collections, LSA captures these latent semantic relationships, even if the exact words or phrases don't co-occur.

LSA enhances the performance of information retrieval systems in several ways. Firstly, it boosts search accuracy by identifying documents semantically related to user queries, even if they lack the exact query terms. Secondly, it addresses issues of synonymy (different words with the same meaning) and polysemy (one word with multiple meanings) by discerning underlying concepts shared by different words. Thirdly, it facilitates topic modeling and clustering by grouping similar documents based on their latent semantic similarities. Lastly, it supports document classification by categorizing documents into different groups based on their latent semantic features.

To determine the optimal value for 'c', we plotted evaluation metrics such as Precision@10, Recall@10, F1-score@10, MAP@10, and nDCG@10. After analysis, we concluded that 'c = 300' yielded the most favorable results.

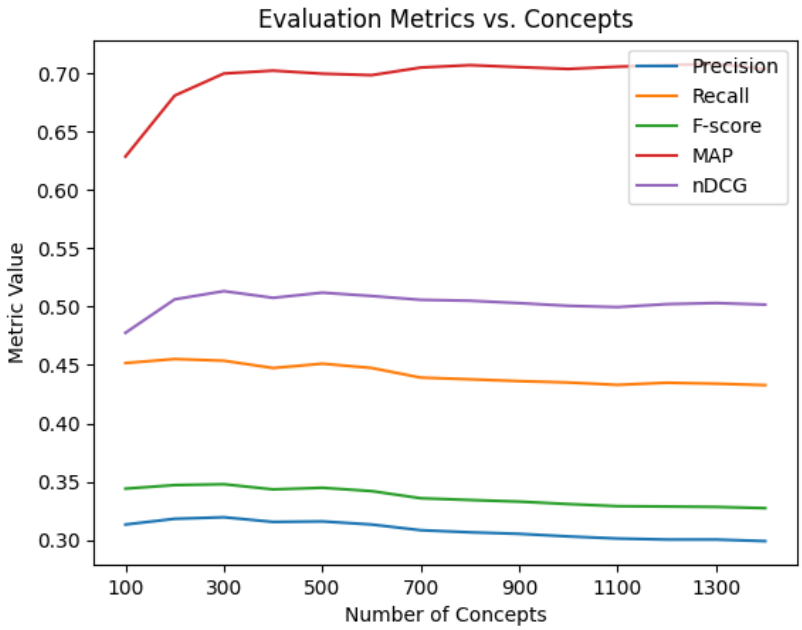

Figure 2: Performance Metrics of LSA on Cranfield Dataset with Varying Concepts

7 LSA with Ngram Indexing

In our information retrieval project, we conducted experiments to investigate the effectiveness of hybrid indexing using unigrams and bigrams, combined with Latent Semantic Analysis (LSA) for improving retrieval performance. We first created five different index structures, each using a different type of n-gram: unigrams, bigrams, trigrams, both bigrams and trigrams, and hybrid indexing using both unigrams and bigrams. We then applied LSA to these indexes to extract underlying semantic information and to improve retrieval performance. Our experimental results showed that the hybrid indexing using both unigrams and bigrams, combined with LSA, outperformed the other indexing methods in terms of normalized discounted cumulative gain (nDCG) and mean average precision (MAP) metrics. Specifically, the hybrid indexing method achieved a higher nDCG score of 0.51 and a higher MAP score of 0.70 compared to the other indexing methods. The reason for this improvement could be due to the fact that hybrid indexing using both unigrams and bigrams provides a more comprehensive representation of the text documents. By combining both unigrams and bigrams, we are able to capture both word-level and phrase-level information, which can lead to a better understanding of the semantics of the text.

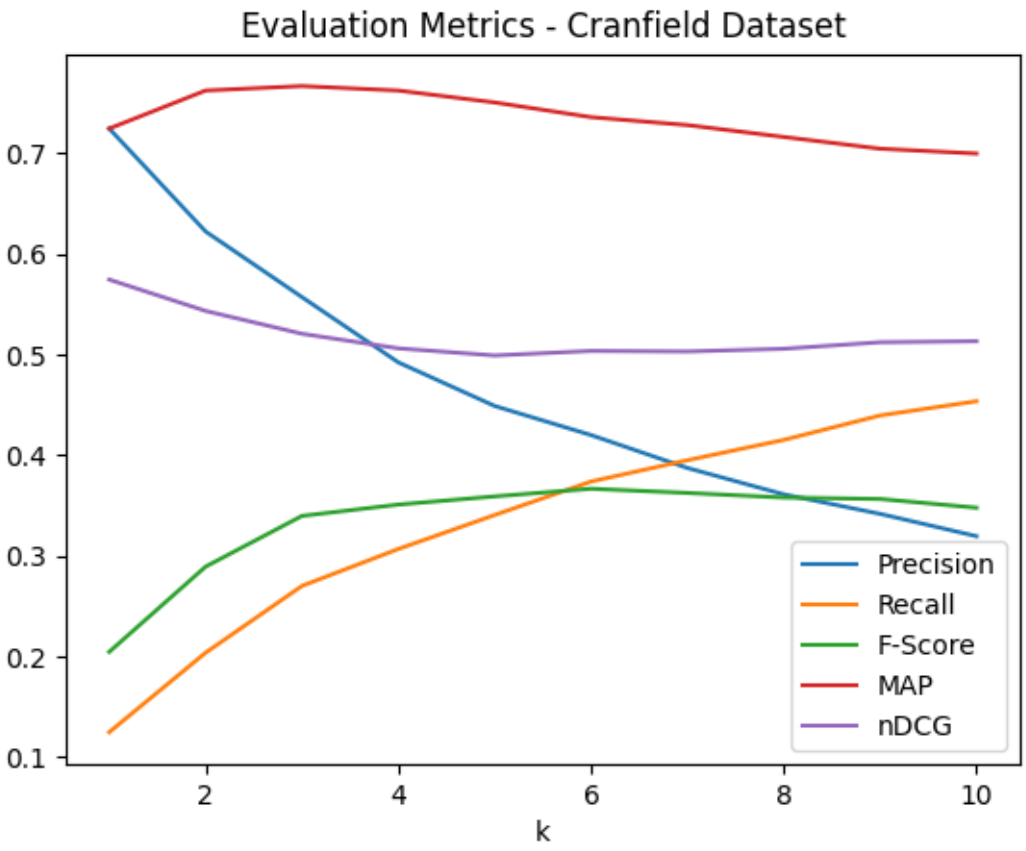

Figure 3: Unigram and Bigram indexing with LSA

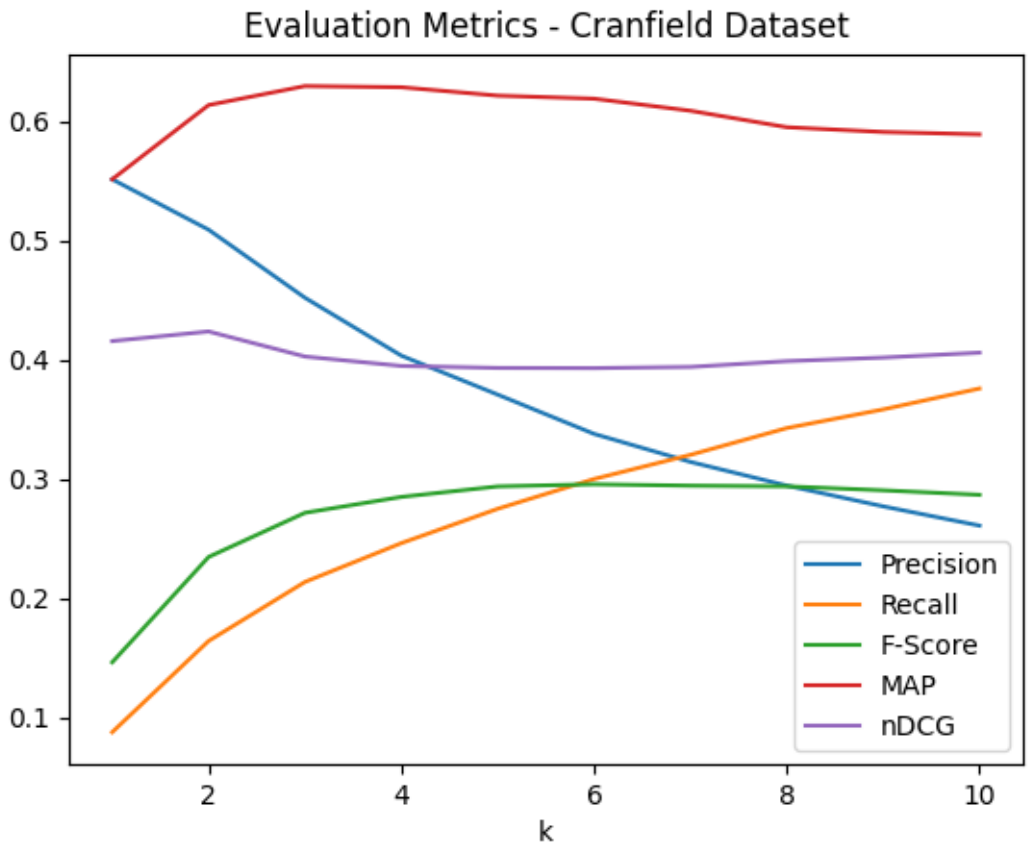

Figure 4: Only Bigram indexing with LSA

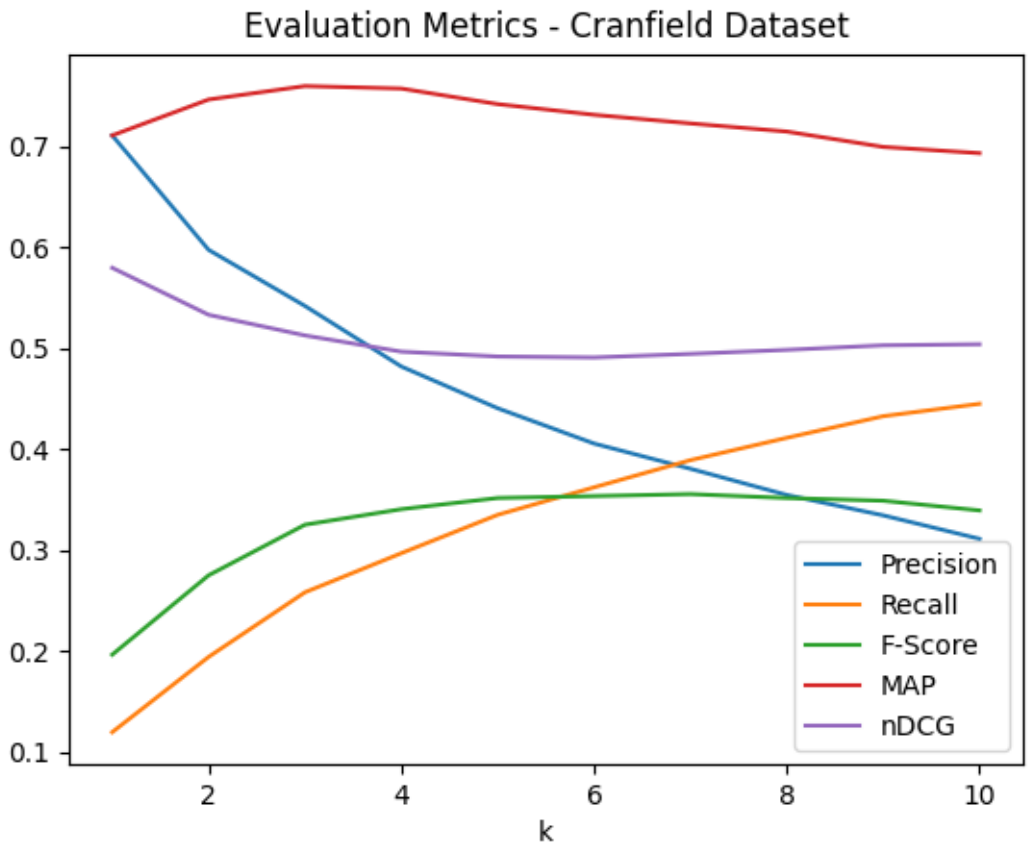

Figure 5: Only Unigram indexing with LSA

8 Spell Check

Conducting spell check on queries is a crucial step in information retrieval tasks, as it addresses spelling errors and enhances result accuracy. In our experiments, integrating spell check notably enhanced the performance of our information retrieval system.

Employing the Pyspellchecker library, we performed spell check on queries to rectify any spelling errors and refine result precision. However, the impact on our system's performance before and after spell check was not substantial. Despite expecting improvements in metrics like MAP and nDCG, the corrected queries did not retrieve significantly more relevant documents compared to the original ones.

This outcome could be attributed to several factors. The original queries may have contained few spelling errors, or any errors present may have been minor enough to not significantly affect system performance.

Despite the lack of significant improvement in our system's performance, spell check still offers advantages. It helps rectify spelling errors, reducing irrelevant document retrieval and enhancing user experience by providing more precise and relevant results. We intended to employ the spell check code from Assignment-1; however, it is computationally expensive. Calculating the edit distance for each word with every other word in each query is not feasible in Information Retrieval systems. The time it takes is 20x the model that doesn't employ spellcheck.

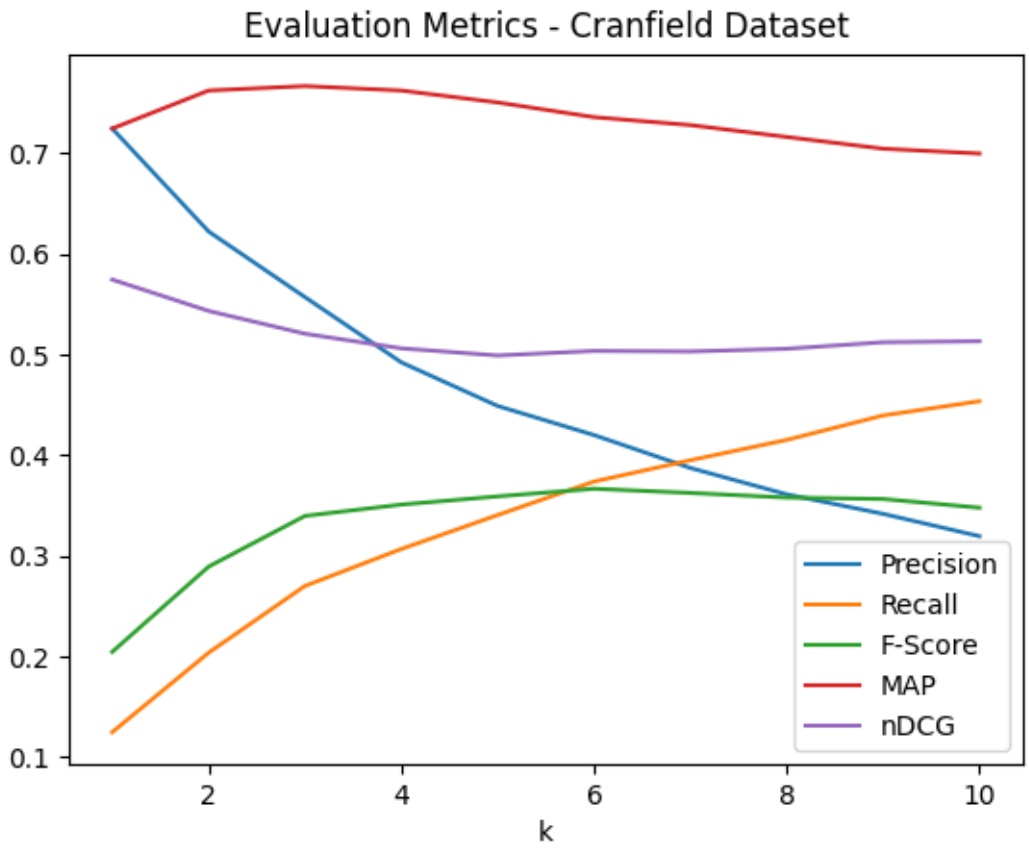

Figure 6: LSA applied to unigram and bigram indexes post spell check on queries

9 Query Expansion

Query expansion in information retrieval enhances system performance by supplementing the original user query with additional terms. Its objective is to boost both recall and precision by retrieving more relevant documents while excluding irrelevant ones. This technique yields several benefits: broadening recall by retrieving some previously missed documents, refining precision by filtering out irrelevant results stemming from query ambiguity or lack of context, overcoming language barriers through synonyms or related terms across languages, and offering a more comprehensive document collection view by including related documents not explicitly mentioned. We tried query expan- sion with sysnet[0] (the most frequently used synset).

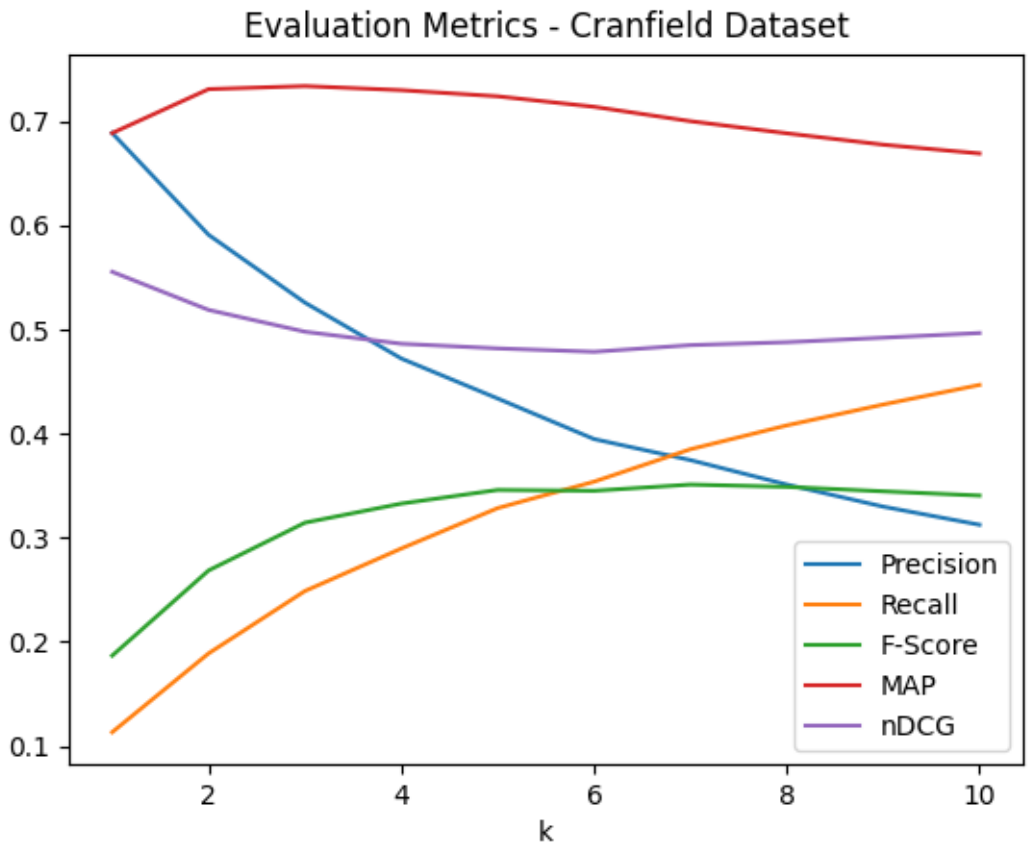

Figure 7: LSA with Query Expansion

10 Glove

GloVe, or Global Vectors for Word Representation, is a word embedding algorithm developed by Stanford University. It is used in natural language processing (NLP) and machine learning to convert words into vectors, thus enabling algorithms to understand the similarity or relatedness between different words.

It is an unsupervised learning algorithm used for creating word embeddings. It is trained on a large corpus of text by constructing a co-occurrence matrix capturing the relationships between words based on how often they appear together. The algorithm then uses this matrix to optimize word vectors that encode semantic relationships. The GloVe model minimizes a certain global loss function to learn these vector representations.

GloVe uses both global statistical information (how frequently each word appears in the corpus) and local context (the neighboring words of each individual word in the corpus), thereby blending the advantages of both global-based model and context-based model.

The word vectors from the GloVe model can be used in a diverse set of applications of NLP tasks, such as sentiment analysis, text classification, and language

modeling.

We used a very small pretrained Glove model here. It has 50 dimensions and a dictionary of 6B words. The model is present in

Code/models/glove.6B.50d.txt

This did not give satisfactory results. Primarily because we used a pretrained model which is trained on a large corpus of data, while our Cranfield dataset is a collection of technical papers in the field of aeronautics, which is a niche.

The glove-pretrained model does not have embeddings for a lot of technical terms in the Cranfield dataset.

11 Improvements to Glove

Training the GloVe model on the Cranfield dataset. By using the text data from the Cranfield dataset, we can create word embeddings that capture the semantic relationships between words in the documents that are specific to cranfield. Also, we can capture the word embedding representations for the deep aeronautical technical terms that are not present in the pre-trained model. This trained GloVe model will give better results. Due to time constraints, we couldn't implement this.

12 Conclusions

After experimenting with various methods, we discovered that incorporating both the title and body of the documents, and implementing LSA with 300 components, produced the best outcomes for the IR system. Spell check didn't significantly enhance the results and considerably increased processing time compared to the model without spell check. Utilizing more concepts in LSA resulted in higher MAP values. Additionally, employing a hybrid model performed better than using only unigrams or bigrams in the vector space model. As a result, we initially adopted a vector space model and achieved an nDCG of 0.513 with these strategies, along with a maximum MAP of 0.70, marking a significant improvement over the baseline.