IR Project Report – Part 3

Paragraph Retrieval using BM25 and flan-t5 base

Group Members

Hari Hara Prasad Goud 2022A7PS0011H Rohit Srinivas Bollineni 2022A7PS1294H Abhinav Ganta 2022A7PS0151H Course: Information Retrieval

April 2025

1 Introduction

The aim of this project is to develop an Information Retrieval (IR) system capable of retrieving relevant paragraphs from a document corpus based on a user query. The system leverages the BM25 ranking function to retrieve candidate paragraphs and uses BLEU score for evaluation of the retrieval quality. This report focuses on the final phase of the project, including the system's complete design, evaluation metrics, result analysis, challenges faced, and a demonstration of the working system.

2 Retrieval Process

We implemented BM25, a state-of-the-art probabilistic retrieval algorithm, using the rank_bm25 Python library. The process involves:

- **Preprocessing:** Tokenization, lowercasing, optional removal of punctuation and stopwords.
- **Indexing:** Each paragraph from the corpus is tokenized and indexed in the BM25 model.
- Query Processing: User queries are cleaned and tokenized similarly.
- Retrieval: BM25 computes a relevance score between the query and each paragraph, returning the top-k results.

This method is efficient and well-suited for our use case with medium-sized document corpora.

3 Ranking Algorithm

BM25 scores a document D with respect to a query Q as:

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

Where:

- $f(q_i, D)$ is the term frequency of query term q_i in document D
- |D| is the length of document D
- avgdl is the average document length in the corpus
- $k_1 = 1.5, b = 0.75$ are empirically tuned constants

This scoring balances term relevance and document length normalization.

4 Analysis

High BLEU scores and precision indicate that the retrieved paragraphs closely match reference ones. Moderate recall suggests some relevant results might be missed, possibly due to paraphrasing. The BM25 model outperforms Boolean retrieval by ranking partially matching paragraphs more effectively and offering finer-grained relevance scoring.

5 Challenges Faced

- Sparse Queries: Some queries lacked descriptive terms.
 - Solution: Used query expansion with synonyms.
- Long Paragraphs: BM25 favored longer texts.
 - Solution: Tuned the b parameter to penalize verbosity.
- BLEU Limitations: BLEU was not designed for IR.
 - Solution: Supplemented BLEU with MAP, nDCG, Precision/Recall.

6 System Demonstration

We developed a Python-based CLI/GUI tool that:

- Accepts user queries
- Retrieves and ranks paragraphs using BM25
- Displays results with relevance scores and accuracies
- Outputs evaluation metrics for test queries

The system is responsive and scalable, aligning with the described methodology and demonstrating all key functionalities effectively.

7 Dataset

We have curated our dataset from the publicly available resources on the Indian Kanoon website, a widely used legal information portal. The dataset comprises 20 landmark Supreme Court cases from India, spanning the period from 2015 to 2019. These cases were carefully selected to ensure a diverse representation across various months and legal domains, including constitutional law, civil disputes, criminal appeals, and administrative judgments. Each case was parsed to extract relevant paragraph-level text data, which forms the basis of our document corpus for retrieval tasks.

```
Enter your corry; the compliciting series one establish on evidence that the splitcation has in fact security of discretion even in spite of the sparred sericitory. So the appoilant camer contend, at this rase, that he is emitted to discharge on the sponsed that the imputations in the attested publication were not per as defeatory.

**Series of 1,5006**

**The only affects*

**Series of 1,5006**

**The only affects*

**Series of 1,5006**

**The only affects*

**The only affects*

**Series of 1,5006**

**The only affects*

**The
```

Figure 1: Observed Output