Indian Institute of Technology Delhi

Department of Computer Science and Engineering

COL 764: Assignment 4

Document Reranking using LLMs

Submitted by

Student Name: Rohan Chaturvedi
Registration Number: 2022MT11262

Course Instructor

Srikanta Bedathur

Submission Date: 25/10/2024

Contents

1 Introduction

This report presents the implementation and analysis of document reranking using Large Language Models (LLMs) for COL 764 Assignment 4. The objective of this assignment is to evaluate the performance of query expansion using LLMs for improving retrieval quality.

The task involves:

- Generating pseudo-documents using zero-/few-shot prompting with LLMs.
- Expanding queries with terms from these pseudo-documents.
- Reranking a pre-retrieved set of top-100 documents for each query using BM25.

Key metrics used for evaluation are nDCG@5, 10, and 50.

2 Implementation

The implementation of the system consists of several key components, including data preprocessing, ranking using the BM25 algorithm, query expansion, and evaluation. Each of these components is explained in detail below.

2.1 Data Preprocessing

The system begins by standardizing the input data, which includes queries, document collections, and relevance judgments. Preprocessing involves the following steps:

- **Tokenization:** Each document and query is split into individual tokens (words).
- **Stop-word Removal:** Commonly occurring, non-informative words (e.g., "and", "the") are removed.
- **Vocabulary Construction:** A vocabulary V is built, consisting of unique terms from the corpus, with their corresponding frequencies $f(t)$ for $t \in V$.

The vocabulary serves as the basis for computing term statistics, such as the document frequency $df(t)$, which is the number of documents containing term t.

2.2 Ranking using BM25

The BM25 algorithm computes a relevance score for each query-document pair. The score is calculated using the following formula:

$$\text{BM25}(q,d) = \sum_{t \in q} \text{IDF}(t) \cdot \frac{f(t,d) \cdot (k_1 + 1)}{f(t,d) + k_1 \cdot \left(1 - b + b \cdot \frac{|d|}{\text{avgdl}}\right)}$$

where:

- $f(t,d)$: Term frequency of term t in document d.
- $|d|$: Length of document d in terms of number of tokens.
- avgdl: Average document length in the corpus.
- k_1: Parameter controlling the term frequency saturation effect.
- b: Parameter for length normalization.

- IDF(t): Inverse Document Frequency of term t, given by:

$$\text{IDF}(t) = \log\left(\frac{N - df(t) + 0.5}{df(t) + 0.5} + 1\right)$$

  where N is the total number of documents in the corpus.

Documents are ranked in descending order of their BM25 scores for each query.

2.2.1 Query Expansion

To enhance retrieval performance, the system employs a query expansion technique that combines the original query with additional terms generated from the retrieval context. However, due to the brevity of the initial query, its influence on retrieval is reinforced by repetition. The process involves the following steps:

- The original query q is repeated five times to form an extended base query:

$$Q_{\text{base}} = q \oplus q \oplus q \oplus q \oplus q$$

  where $\oplus$ represents the concatenation operator. For example, if the original query is:

$$q = \text{"who killed nicholas ii of russia"}$$

  the extended query becomes:

  Q_{base} = "who killed nicholas ii of russia who killed nicholas ii of russia who killed nicholas ii of russia wh

- Expansion terms T_{exp} are generated using a relevance model or other techniques (e.g., embedding-based similarity). These terms are selected to maximize query coverage over relevant documents. For instance, if $T_{\text{exp}} = \{t_1, t_2, \ldots, t_k\}$, the expanded query is:

$$Q = Q_{\text{base}} \oplus T_{\text{exp}}$$

- The final expanded query Q is processed by the retrieval model, which incorporates the repeated terms appropriately. The repetition ensures that the model retains the emphasis on the original query terms while leveraging the additional semantic context from T_{exp}.

The use of repeated query terms ensures that the original intent remains prominent, even when additional terms are appended. The retrieval model must adjust term weighting to balance the influence of repeated terms and expansion terms. For instance, term frequency $f(t, Q)$ in the scoring function is adjusted to account for repeated terms, maintaining effective query representation.

2.3 Evaluation Metrics

The ranking quality is evaluated using the Normalized Discounted Cumulative Gain (nDCG) metric. For a given query, the nDCG at position p is computed as:

$$\text{nDCG}_p = \frac{\text{DCG}_p}{\text{IDCG}_p}$$

where:

- $\text{DCG}_p = \sum_{i=1}^{p} \frac{2^{\text{rel}_i} - 1}{\log_2(i+1)}$, with rel_i being the relevance score of the document at position i.
- IDCG_p: The maximum possible DCG value for the top p documents, obtained by ranking the documents in descending order of relevance.

The nDCG values are averaged across all queries to compute the overall performance.

2.4 Dataset Sampling for k-shot Learning

The system supports dataset sampling for k-shot learning experiments. Let Q represent the set of all queries, and $D(q)$ the set of documents associated with query q. For k-shot sampling:

$$S_k = \{(q, D'(q)) \mid q \in Q, |D'(q)| = k, D'(q) \subseteq D(q)\}$$

where S_k is the sampled dataset containing k documents per query. The sampling strategy can be random or based on disjoint subsets, depending on experimental requirements.

2.5 System Workflow

The overall workflow integrates these components as follows:

1. Preprocess the input data and construct the vocabulary.
2. Construct a set of samples queries
3. Perform k-shot sampling.
4. Rank documents based on scores on new query and evaluate using nDCG.

This modular design ensures flexibility and extensibility for various retrieval tasks.

3 Hyperparameters

The performance of the system is influenced by several key hyperparameters:

- **BM25 Parameters:** The BM25 algorithm utilizes two critical parameters:
  - k_1:(1.9044) Controls the term frequency saturation effect, determining the extent to which term frequency contributes to the score.
  - b:(0.3299) A length normalization parameter that adjusts the importance of document length in scoring.
- **Query Expansion:** Optional augmentation of queries incorporates additional context, which can be generated using diverse or random sampling. This extension enhances retrieval by expanding the scope of document matching.
- **Sampling Parameters:** During k-shot learning, the number of samples (k) is adjustable, allowing for experimentation with different dataset sizes. Additionally, the sampling strategy can be either random or disjoint, depending on the desired degree of overlap with existing datasets.

The system's modularity and tunable hyperparameters ensure flexibility and adaptability for various retrieval tasks, enabling robust evaluation and optimization of ranking performance.

3.1 Training Data Preparation

The training queries and the corresponding pseudo-relevance set of top-100 documents were used to generate (query, passage) pairs. Two strategies were employed for passage selection:

1. **Random Passage Selection:** Passages were randomly selected from documents marked as relevant. Each passage was limited to 1-4 sentences.
2. **Diverse Passage Selection:** Diverse passages were selected based on cosine similarity between term-vector representations. The first passage was chosen as the initial 3 sentences of the top-ranked document.

3.2 Generated Collections

- **Random Collection:** A total of n_1 (query, passage) pairs were generated using random selection.
- **Diverse Collection:** A total of n_2 (query, passage) pairs were generated using diversity-based selection.

4 Reranking with BM25

The BM25 retrieval model was used for reranking the expanded queries. Key considerations included:

- Concatenating the original query 5 times with the expansion terms.
- Hyperparameter tuning for BM25, including values of k_1 and b.

4.1 Evaluation Metrics

The table below reports the nDCG scores for different strategies (Random, Diverse) and disjoint setups, with k-values ranging from 0 to 4. The metrics include nDCG@5, nDCG@10, and nDCG@50.

Strategy Type 1	Strategy Type 2	k	nDCG@5	nDCG@10	nDCG@50
Random	**Random**	0	0.50	0.53	0.70
		1	0.50	0.51	0.68
		2	0.49	0.52	0.69
		3	0.53	0.54	0.71
		4	0.46	0.50	0.68
	Disjoint	0	0.50	0.51	0.68
		1	0.44	0.47	0.67
		2	0.49	0.52	0.69
		3	0.53	0.55	0.70
		4	0.48	0.52	0.69
Diverse	**Random**	0	0.50	0.51	0.69
		1	0.48	0.50	0.69
		2	0.51	0.51	0.69
		3	0.49	0.52	0.68
		4	0.47	0.51	0.68
	Disjoint	0	0.50	0.52	0.70
		1	0.48	0.48	0.67
		2	0.53	0.53	0.71
		3	0.46	0.50	0.68
		4	0.46	0.50	0.67

4.2 Best Strategy

The best-performing combination was:

- Optimal Strategy: *Diverse & Disjoint*
- Number of Examples: $k = 2$
- nDCG@5: **0.53**

- nDCG@10: **0.53**
- nDCG@10: **0.71**

5 Conclusion

In this assignment, we explored the effectiveness of document reranking using Large Language Models (LLMs) in conjunction with different preprocessing and prompt-selection strategies. By augmenting the initial query with pseudo-relevant passages generated via few-shot prompting, we aimed to enhance retrieval performance on a benchmark dataset.

Through systematic experimentation, we evaluated the impact of two passage selection strategies—*Random* and *Diverse*—along with prompt-selection methods such as *Random* and *Disjoint*. The evaluation metrics, specifically nDCG@5, nDCG@10, and nDCG@50, were used to assess the quality of reranked results across varying k-values (0 to 4).

Our results indicate that the *Diverse & Disjoint* strategy combination consistently outperformed other configurations, achieving the best performance at $k = 2$. This demonstrates the importance of diversity in passage selection and the effectiveness of disjoint prompts in leveraging the contextual richness of LLM-generated expansions. Additionally, the findings reaffirm the utility of BM-25 as a robust baseline for reranking tasks, especially when paired with high-quality query expansions.

In conclusion, this study highlights the potential of integrating LLMs into classical retrieval systems to improve ranking effectiveness. Future work could involve scaling the experiments to larger datasets, exploring alternative LLMs, and incorporating additional metrics to provide a more comprehensive evaluation of retrieval quality.