Personalized Information Retrieval

Search Engine capable of retrieving relevant answers to user queries from a community Question Answering dataset.

INFORMATION RETRIEVAL AND RECOMMENDER SYSTEMS PROJECT

16/02/2025

Author: Andrea Scalora 516954

Personalized Information Retrieval

1 Introduction

This project focuses on exploring the practical and theoretical aspects of Personalized Information Retrieval (PIR). PIR aims to enhance traditional search systems by incorporating user-specific signals, enabling more targeted and contextually relevant search results. For this project, we will focus on a subset of the SE-PQA dataset.

Our approach leverages both statistical retrieval methods, such as BM25, and modern neural re-ranking techniques using lightweight models like MiniLM and DistilBERT QA. Additionally, we integrate query expansion strategies, where models like T5 are used to enrich user queries with additional context, and personalization signals—such as tag-based similarity—to further refine the ranking process. By combining these methods, we strive to build a robust retrieval pipeline that effectively fuses multiple signals, ultimately improving the overall relevance and quality of search results.

The project is implemented using PyTerrier, which provides a flexible framework for constructing retrieval pipelines, and is designed to work efficiently in environments with limited hardware resources (e.g., CPU-only setups). Our experiments involve optimizing key parameters for BM25 and query expansion, as well as evaluating the performance of various fusion strategies using standard retrieval metrics.

2 Intuition of my Idea

The fundamental intuition behind this project is to enhance traditional retrieval methods by integrating multiple complementary signals that capture both lexical matching and semantic understanding. BM25 has long been a robust baseline in information retrieval due to its robust performance with term frequency and document length normalization. However, BM25 relies purely on statistical term matching, which may not fully capture the nuanced meaning of queries and documents.

To address this, I incorporated neural re-ranking models such as MiniLM and DistilBERT QA. These models generate dense embeddings that capture semantic relationships between queries and documents, thereby helping to re-rank the initial BM25 results. This re-ranking step aims to promote documents that are not only lexically similar but also semantically relevant.

In addition, I experimented with query expansion using T5. The idea is to enrich the original user query with additional, contextually relevant terms. This can help overcome vocabulary mismatch issues and improve retrieval performance by introducing diverse expressions related to the original query.

Finally, a fusion strategy was employed to combine the strengths of different retrieval signals. Techniques such as Reciprocal Rank Fusion (RRF) and weighted sum fusion were explored to blend the BM25 score, neural re-ranking scores and T5 query expansion score into a single final score. The intuition here is that while BM25 provides a strong baseline, neural models and additional signals can complement it by capturing semantic nuances, and fusion helps leverage the consensus across these different approaches.

This multi-faceted approach is designed to deliver more personalized and relevant search results, particularly in environments with limited computational resources, where lightweight neural models are essential.

3 Methodology

In this project, the overall methodology is structured into several key components: data preparation, utility files, modeling and retrieval pipeline construction, query expansion, and fusion retrieval.

3.1 Data Preparation

Dataset Loading and Preprocessing:
The raw data (train, validation, and test sets) are loaded from JSONL files. A custom text processing pipeline is applied to the "text" fields to clean and normalize the data—this includes case folding, punctuation removal, tokenization, and lemmatization. The processed data are then saved as pickle files for efficient reuse (using Pickle library).

Merging and Standardization:
The text of the answers, which are missing from the subset provided, are merged with the original dataset using a custom function, i.e. *merge_answers_in_chunks*, to enrich the primary dataset. The merged data are then standardized by renaming columns (we obtain qid, query, docno and text) and removing rows with missing or empty values.

Index Construction:
A unified corpus is built by concatenating the train, validation, and test datasets (with duplicate document identifiers removed) and is indexed using PyTerrier's *IterDictIndexer* and *IndexFactory*. This index serves as the foundation for subsequent retrieval experiments.

3.2 Utility files

These four files, *lib_to_use*, *general_functions*, *paths*, *text_processing_functions*, were created to centralize configurations and reusable functions, which helped to streamline the development process and allowed for consistent data processing and parameter tuning across different experimental notebooks.

3.3 Modeling and Retrieval Pipelines

BM25 Baseline and Parameter Optimization:
The BM25 retrieval model is implemented as a baseline using PyTerrier. A grid search is performed over a range of parameters (k1 and b) to optimize the BM25 settings on the validation set.

Neural Re-Ranking:
Two lightweight neural models—MiniLM and DistilBERT QA—are employed for re-ranking the BM25 results. Custom re-ranker classes are defined (using PyTerrier's transformer framework) to compute dense embeddings for queries and document texts in batches, and cosine similarity is used to assign new scores. These neural re-rankers are integrated into the retrieval pipelines using the query-level application operator, to effectively re-order the documents based on the full context of the query rather than processing each document in isolation.

Query Expansion:
A T5-based query expander (t5-small) is implemented to enrich the original queries. The expander constructs prompts that instruct the model to add new, relevant terms without repeating those already present. A grid search is also conducted on the parameter *max_new_tokens* to determine the optimal amount of expansion that improves retrieval performance.

3.4 Fusion Retrieval

Reciprocal Rank Fusion (RRF):
For each query, individual results from different retrieval systems are ranked, and an RRF score is computed as

1 / (k + rank). The RRF scores from various systems are then summed to obtain a final score, allowing the consensus across systems to boost consistently high-ranking documents.

Weighted Sum Fusion:
In a parallel approach, the individual scores from the retrieval systems are first normalized (using SKLearn StandardScaler()) and then combined using a weighted linear sum. The weights are chosen based on empirical evaluation, giving higher importance to the baseline model and the one using t5-small while still leveraging the semantic signals from neural models.

3.5 Evaluation Metrics

The performance of each system and fusion strategy is evaluated using a set of standard retrieval metrics, including:

- **Precision at rank 1 (P_1):** Indicates the accuracy of the top-ranked result (most important metric).
- **Recall at 100 (recall_100):** Measures the fraction of relevant documents retrieved among the top 100.
- **Mean Average Precision (MAP_cut_100):** Provides a single-figure measure that accounts for both precision and recall over the top 100 results.
- **Normalized Discounted Cumulative Gain (ndcg_cut_3):** Focuses on the ranking quality of the top 3 results by considering the positions of the relevant documents.

These metrics and their cutoffs are chosen to balance between assessing early precision (critical in retrieval tasks) and overall ranking performance, ensuring that the system not only retrieves relevant documents but also orders them effectively.

4 Experiment and Results

Experiment 1: BM25

The results of the grid search shows that the optimal parameters for the BM25 model are k1= 1 and b=1. This model configuration achieved this performance:

$$P@1 \approx 0.71,\ recall@100 \approx 0.93,\ MAP@100 \approx 0.77,\ NDCG@3 \approx 0.77.$$

Experiment 2: Neural Re-ranking

BM25+MiniLM achieved metrics:

$$P@1 \approx 0.63,\ recall@100 \approx 0.85,\ MAP@100 \approx 0.69,\ NDCG@3 \approx 0.69.$$

BM25+DistilBERT QA produced similar results:

$$P@1 \approx 0.64,\ recall@100 \approx 0.82,\ MAP@100 \approx 0.70,\ NDCG@3 \approx 0.70.$$

Experiment 3: Query Expansion with T5

I implemented a T5-based (small) query expander and conducted a grid search on the parameter *max_new_tokens*, testing values from 3 to 30.
The expanded queries were integrated into the BM25 pipeline, and while the approach aimed to enrich the query context, the overall retrieval improvements were marginal. From the grid search I noticed that after the threshold of 12 tokens, the value of the P@1 metric starts to decrease. The best value for *max_new_tokens* (equal to 10) is where P@1 reached its peak:

P@1 ≈ 0.72, recall@100 ≈ 0.93, MAP@100 ≈ 0.78, NDCG@3 ≈ 0.77.

Experiment 4: Fusion Retrieval

We explored two fusion strategies:

- **Reciprocal Rank Fusion (RRF):** RRF aggregates rank-based scores from each system by summing the reciprocal of (k + rank) for each document. It achieved metrics:

  P@1 ≈ 0.66, recall@100 ≈ 0.93, MAP@100 ≈ 0.72, NDCG@3 ≈ 0.71.

- **Weighted Sum Fusion:** For this approach I tried different combinations of weights, considering to assign greater weights to the better models. With weights BM25: 0.2, MiniLM: 0.15, DistilBERT QA: 0.15, T5: 0.5 it achieved:

  P@1 ≈ 0.71, recall@100 ≈ 0.93, MAP@100 ≈ 0.78, NDCG@3 ≈ 0.78.

Models	P@1	recall@100	MAP@100	NDCG@3
BM25	0.71	0.93	0.77	0.77
MiniLM	0.63	0.85	0.69	0.69
DistilBert-QA	0.64	0.84	0.70	0.70
T5-small	0.72	0.93	0.78	0.77
RRF Fusion	0.66	0.93	0.72	0.71
Weighted Fusion	0.71	0.93	0.78	0.78

Table to show the results of each model.

Results

- **BM25 Baseline:** Provides a strong starting point for retrieval.
- **Neural Re-ranking:** Although promising in theory, the neural re-rankers did not outperform BM25 on our dataset, possibly due to domain-specific factors.
- **Query Expansion:** Adding context via T5 can help address vocabulary mismatch, but it requires careful tuning to avoid introducing noise in the original query.
- **Fusion Strategies:** Combining multiple signals (lexical, semantic, and contextual) through fusion methods shows potential for modest improvements, indicating that integrating diverse retrieval signals is a viable approach.

5 Possible Improvements and Conclusions

This project demonstrates that while traditional retrieval methods such as BM25 provide a strong baseline, there is potential for improvement by integrating additional signals. In the future, it would be beneficial to implement a more advanced personalization system by incorporating user-specific signals, such as computing the similarity between user tags and document tags. For instance, integrating a mechanism that adjusts the query based on a user's profile and historical interactions could further refine retrieval performance. Additionally, exploring learning-to-rank techniques that combine both traditional and neural signals, with domain-specific fine-tuning, may lead to significant improvements in personalized information retrieval.