Comparative Study of Conversational Search Engine Retrieval Pipelines

Group ID: #5

Til Mohr
tv.mohr@stud.uis.no

Ishaac Ourahou
i.ourahou@stud.uis.no

ABSTRACT

In our master's degree program, we delved into the realms of information retrieval and text mining. Our group project focuses on developing a system capable of searching for specific passages within a conversation. This entails considering the historical context of previous exchanges when interpreting the current user's query. The system reformulates the user's query to better align with the conversation's overarching topic and then seeks the most relevant passages from a vast collection of 8.8 million documents, specifically from the MS MARCO document collection. We initially create a baseline information retrieval pipeline, which serves as our foundational reference. We then explore advanced techniques and methods in the field of conversational search engines. Our report concludes with a comparative analysis between the baseline system and our advanced retrieval mechanisms, outlining the strengths and shortcomings of our approaches, and offering conclusions.

KEYWORDS

information retrieval, conversational search engine, conversational search system

1 INTRODUCTION

With the rise of conversational agents like Amazon Alexa and Apple Siri, the ability to accurately and relevantly respond to user requests has become increasingly important. These agents are designed to interact fluidly with users, and the essence of their functionality rests on providing precise information in their responses.

Our project centers on building a conversational search engine, a system that not only addresses the user's immediate question but also takes into account previous queries and the ongoing conversation's topic. This provides a more holistic approach to the user's information needs. Initially, we will construct a basic architecture, establishing our baseline system. This baseline will serve as a comparison point against more sophisticated techniques. It's required, however, that even this foundational system meets or exceeds the performance of the reference system set by our project guides.

Our research will utilize the MS MARCO dataset, a comprehensive collection of 8.8 million documents. Efficiently retrieving relevant documents from this vast pool necessitates an indexing process, which we will execute using the `pyterrier` API.

Section 2 will articulate the specific problem we aim to address. We will then elaborate on the methodologies behind our retrieval pipelines. Broadly, these pipelines can be broken down into stages: Query revision, single- or multi-pass retrieval, and reranking. Sections 4 and 8 will delve into our baseline and advanced techniques for each of these stages, respectively. Our report will conclude in section 9, where we evaluate and compare the effectiveness of our systems, highlighting the strengths and potential areas of enhancement in our advanced approaches.

The codebase for our project can be found on GitHub[1].

2 PROBLEM STATEMENT

This project focuses on developing a Conversational Search Engine (CSE). What distinguishes a CSE from a traditional search engine is its inherent context-aware nature. In a CSE, queries asked by a user within a session are semantically interconnected. Rather than treating each query in isolation, a CSE leverages information from both the previous queries and the system's responses to those queries to retrieve relevant passages. It must cater to the continuous thread of a dialogue, while also being agile enough to accommodate sudden shifts in conversation topics. For every user query, the goal is to generate a ranked listing of the top 1000 passages out of an expansive 8.8 million document collection.

A vital component that sets a CSE apart is its query rewriting mechanism. This feature reformulates the presented query to better reflect the ongoing conversation topic. Figure 1 offers a schematic representation of how the various stages of a CSE retrieval pipeline interact. Analogous to traditional search engines, a CSE also requires prior indexing of the document collection, enabling computationally efficient retrieval and ranking processes.

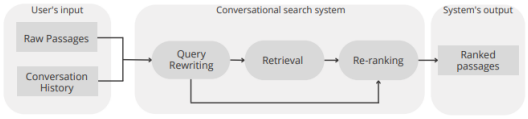

Figure 1: Overall system's pipeline architecture

3 RELATED WORK

In this section, we delve into pertinent research encompassing the realms of conversational search engines and the broader area of information retrieval. While certain highlighted studies do not directly cater to conversational search engines or explicit information retrieval, their techniques remain invaluable in various stages of the conversational retrieval process.

Pseudo-Relevance Feedback by Query Expansion

Text-to-Text Transfer Transformer

The vast domain of natural language processing (NLP) revolves around the understanding of natural language, whether presented

[1]URL: https://github.com/CodingTil/2023_24---IRTM---Group-Project

in text or speech form. NLP aspires to equip computers with the capability to grasp the depth of human language and harness this understanding to execute a range of tasks, such as text summarization, machine translation, and question answering. Given the diverse nature of these tasks in terms of their input, output, and underlying challenges, developing a unified model proficient across the entire spectrum poses a significant challenge.

Enter the Text-to-Text Transfer Transformer (T5) [8]. This work by Raffel et al. introduces transfer learning in NLP, aiming to craft a versatile model that can be used for any NLP problem. In essence, T5 models first learn the basics of language. Then, they're sharpened for particular tasks using targeted data. It's common to find models that have been trained in this manner for any specific NLP problem.

doc2query Document Expansion

Traditional retrieval techniques, such as BM25, rely primarily on term occurrences in both queries and documents. However, they often overlook the semantics of the content. As a result, documents that may be semantically relevant to a query might be scored as non-relevant due to differences in syntax or terminology. Dense retrieval methods, which emphasize semantic similarities between texts, can address this problem but are computationally taxing during retrieval.

A notable solution to this is the doc2query method proposed by Nogueira et al. [6]. It employs a text-to-text transformer to convert documents into queries. By generating and appending a few of these transformed queries to the original document, classical retrieval methods show significantly improved performance. This is because these additional queries often capture semantic nuances similar to those in the actual query [4, 6, 7]. Importantly, doc2query shifts the computational load to the indexing phase, ensuring minimal performance lag during retrieval. By leveraging the T5 model, the authors further enhanced the query generation quality, leading to the variation known as docTTTTTquery, doc-T5query, or doc-2query-T5 [4].

SPARTA Sparse Retrieval

SPARTA, introduced by Zhao et al. [10], represents a nuanced take on sparse retrieval. At its core, it works by encoding documents into sparse representations during the indexing phase. These representations not only capture the document's actual content but also incorporate terms that are semantically resonant, even if they're not present in the document. This underlying principle echoes the rationale of approaches like doc2query and dense retrieval models.

Yet, where SPARTA differentiates itself is in its retrieval phase. Unlike dense retrieval models, it retrieves pertinent documents using straightforward index lookups, mirroring lexical retrieval strategies like BM25 [10].

However, in real-world applications, SPARTA faces challenges. Several other models, including BM25 and doc2query-T5, surpass it in ranking efficacy. Additionally, its indexing footprint is substantially larger compared to alternatives like doc2query-T5 [9].

monoT5 & duoT5 Rerankers

monoT5 and duoT5 are neural re-rankers, also developed by Nogueira et al., which attempt to inject semantic understanding into the retrieval process [3, 5]. Using the T5 model, they re-rank a list of documents based on their semantic relevance to a given query. Specifically, monoT5 processes a query and a single document, outputting a relevance score. In contrast, duoT5 considers a query and two documents, determining which document is more relevant. Although duoT5 offers a more nuanced ranking, its pairwise comparison method makes it computationally heavier. Hence, a staged re-ranking approach is proposed: first using monoT5 for the top k documents and subsequently applying duoT5 to a smaller subset, the top l, where $l \ll k$ [5, 7].

Expando-Mono-Duo Design Pattern

The same research team introduced a strategic pattern for integrating the above tools into retrieval pipelines, termed the Expando-Mono-Duo design pattern [7]. Here's how it works: During indexing, doc2query-T5 is employed to enhance document representation and better the initial retrieval results from methods like BM25. The retrieved results are then re-ranked with monoT5. A selected top tier from this list undergoes another re-ranking using duoT5. Trials show that this composite approach leads to marked improvements in result quality across multiple evaluation metrics [7].

Conversational Query Rewriting

Conversational search engines distinguish themselves from standard search engines by determining document relevance through the entirety of a conversation, not just the immediate query. In conversational contexts, subsequent questions often lean on prior interactions, implying that previous questions and answers must be factored in when fetching relevant documents. However, there's also a need to cater to conversation shifts where the immediate query doesn't relate to preceding exchanges. Blindly considering the entire conversational history in such cases could detriment retrieval accuracy.

Elgohary et al. address this challenge with an innovative approach [2]. They suggest reshaping the current query based on the overarching conversation. This reformulated query is designed to function autonomously within conventional retrieval pipelines. In essence, this technique extends the utility of standard search engines to conversational question-answering scenarios by introducing a preceding conversational query modification stage.

Employing text-to-text transformers, like T5, can be instrumental in achieving this rewrite. These models are nurtured to revamp the immediate query, factoring in the conversational backdrop. Studies validate the efficacy of this approach, highlighting its capacity to enhance the retrieval accuracy of traditional search engines in conversational contexts [1, 2, 11].

4 BASELINE METHOD

Our baseline method is inspired by the baseline method presented by Łajewska et al. [11]. It is structured in the following sequence:

(1) T5 Query Rewriting
(2) BM25 Retrieval
(3) Re-ranking

(a) Re-ranking using monoT5
(b) Top-document re-ranking using duoT5

4.1 T5 Query Rewriting

In conversation search engines, query rewriting is the crucial component to include the semantics of the conversation history into the currently asked query, which results into a singular rewritten query that can be fed into the retrieval pipeline.

For this purpose, we include all the previously rewritten queries $q'_0 \dots q'_{n-1}$ of our conversation, as well as the response r_{n-1} of the CSE to the previous rewritten query q'_{n-1} into the current query q_n. This is done by concatenating the previous rewritten queries and the response into a single string:

$$q'_n \coloneqq q'_0\text{<sep>} \dots \text{<sep>}q'_{n-1}\text{<sep>}r_{n-1}\text{<sep>}q_n$$

This approach resembles the approach taken by Łajewska et al. [11].

We have found through experimentation that this mere concatenation is insufficient to produce satisfiable results: The concatenated string is too long, and too much focus during the later retrieval is being put on r_{n-1}, which make responses sudden topic changes impossible.

Driven by these insights, we have turned our attention to other query rewriting techniques. Pyterrier provides the SequentialDependence query rewriting method[2]. We have found, however, that this rewriter also does not produce the desired results.

Subsequent exploration led us to the T5 neural query rewriter trained for conversational question rewriting, see Section 3. With this method, q'_n closely mirrored q_n, subtly infusing it with the conversation's context, particularly when no drastic topic alterations were identified. A valuable by-product was the concise nature of the rewritten query, a departure from the growing length observed previously. Since retrieval latency is a critical factor, we utilized a smaller T5 model: castorini/t5-base-canard[3]

4.2 BM25 Retrieval

We settled on the BM25 retrieval method, a commonly used formula in the realm of information retrieval, for its simplicity and its deployment in the reference system, allowing for direct comparisons.

4.3 Re-ranking

The re-ranking stage of our baseline system consists of two stages: First, the top 1000 documents retrieved by the BM25 retrieval method are re-ranked using the monoT5 reranker. Afterwards, the top 50 documents of the previous re-ranking stage are rearranged using the duoT5 reranker, see Section 3. The precise count of documents subject to reranking at each stage is a hyperparameter of our system, allowing to balance computational cost and result quality. These rerankers were implemented in the pyterrier_t5 library.[4] Again, since a low latency of our retrieval pipeline is crucial to us, we utilized smaller T5 models: castorini/monot5-base-msmarco[5] for monoT5 and castorini/duot5-base-msmarco[6] for duoT5.

[2]URL: https://pyterrier.readthedocs.io/en/latest/rewrite.html#sequentialdependence
[3]URL: https://huggingface.co/castorini/t5-base-canard
[4]URL: https://github.com/terrierteam/pyterrier_t5
[5]URL: https://huggingface.co/castorini/monot5-base-msmarco
[6]URL: https://huggingface.co/castorini/duot5-base-msmarco

5 INCORPORATING PSEUDO-RELEVANCE FEEDBACK INTO OUR BASELINE

Recognizing the substantial performance enhancements associated with pseudo-relevance feedback, we felt compelled to integrate a query expansion mechanism into our baseline retrieval method, see Section 4. Our choice fell upon the RM3 query expansion technique, well-established for its robustness and acceptance within the information retrieval community. For a deeper dive into its mechanics and principles, readers are directed to Section 3.

In the Pyterrier framework, the setup requires that any query expansion follows an initial retrieval phase. This initial retrieval fetches the top p documents, forming the foundation for subsequent query expansion using RM3. With the query expanded, it's then passed into a secondary retrieval phase to retrieve the final document set for the end-user. And, to fine-tune the output, we again apply re-ranking using both monoT5 and duoT5.

Henceforth, we'll label this integrated retrieval approach as "baseline + RM3", which is structured as follows:

(1) T5 Query Rewriting
(2) BM25 Retrieval
(3) RM3 Pseudo-Relevance Feedback Query Expansion
(4) BM25 Retrieval
(5) Re-ranking
  (a) Re-ranking using monoT5
  (b) Top-document re-ranking using duoT5

6 DOCUMENT EXPANSION METHOD

JUST IDEA

(0) doc2query-T5 Document Expansion
(1) T5 Query Rewriting
(2) BM25 Retrieval
(3) Re-ranking
  (a) Re-ranking using monoT5
  (b) Top-document re-ranking using duoT5

7 EXTENDING THE DOCUMENT EXPANSION METHOD WITH PSEUDO-RELEVANCE FEEDBACK

JUST IDEA

(0) doc2query-T5 Document Expansion
(1) T5 Query Rewriting
(2) BM25 Retrieval
(3) RM3 Pseudo-Relevance Feedback Query Expansion
(4) BM25 Retrieval
(5) Re-ranking
  (a) Re-ranking using monoT5
  (b) Top-document re-ranking using duoT5

8 ADVANCED METHOD

Explain what you are taking as your advanced method(s), as well as why this is a promising attempt to outperform the baseline method, and why you are making specific implementation choices.

9 RESULTS

The individual methods were evaluated on the MS MARCO document collection and the following provided files: `queries_train.csv`, comprising of a list of queries grouped together into several conversation sessions, and `qrels_train.txt` that contains the relevance assessments for the training queries. Our evaluation focused on a suite of metrics:

- Recall at 1000 (R@1000)
- Mean Average Precision (MAP)
- Mean Reciprocal Rank (MRR)
- Normalized Discounted Cumulative Gain (NDCG_Cut@3)

These metrics were calculated using the `trec_eval` tool:

```
trec_eval -c -m recall.1000 -m map -m recip_rank
-m ndcg_cut.3 -l2 -M1000 qrels_train.txt
{GENERATED_TREC_RUNFILE}
```

As stated in section 4, the baseline method can be parameterized in a few different ways. For this evaluation, we utilized the following configurations: The document retrieval (`BM25`) used the default parameters from `pyterrier`[7] to retrieve the 1000 most-relevant documents for each query. All 1000 documents were then reranked using the `monoT5` reranker. Because of the high computational cost of the `duoT5` reranker, only of those 1000 documents the best 50 documents were then reordered using this reranker.

For the extension of the baseline, the baseline + `RM3` method, we utilized the same configuration for these components. The `RM3` query expansion component was parameterized to expand the query by 26 terms, using the top 17 documents retrieved by the initial `BM25` retrieval.

Table 1: Performance of the different methods on the MS MARCO document collection.

	MAP	MRR	R@1000	NDCG_Cut@3
Reference System	0.07			0.09
Baseline	0.1746	0.3230	0.5705	0.2461
Baseline + `RM3`	**0.2124**	**0.3600**	**0.5710**	**0.2673**

As can be seen in table 1, our baseline method is at least as efficient as the reference system provided by the project owners. Thus, our baseline method fulfills the hard requirement of the project.

Furthermore, comparing the results of the baseline with those of the baseline + `RM3` approach, one can see that simply by adding the query expansion component into the retrieval pipeline one can improve the quality of the ranked list of documents across all measured metrics.

10 DISCUSSION AND CONCLUSIONS

Summarize and discuss different challenges you faced and how you solved those. Include interpretations of the key facts and trends you observed and pointed out in the Results section. Which method performed best, and why? Speculate: What could you have done differently, and what consequences would that have had?

[7] URL: https://pyterrier.readthedocs.io/en/latest/terrier-retrieval.html

REFERENCES

[1] Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. 2020. Open-domain question answering goes conversational via question rewriting. *arXiv preprint arXiv:2010.04898* (2020).

[2] Ahmed Elgohary, Denis Peskov, and Jordan Boyd-Graber. 2019. Can you unpack that? learning to rewrite questions-in-context. *Can You Unpack That? Learning to Rewrite Questions-in-Context* (2019).

[3] Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. *arXiv preprint arXiv:2003.06713* (2020).

[4] Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to docTTTTTquery. *Online preprint* 6 (2019), 2.

[5] Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019. Multi-stage document ranking with BERT. *arXiv preprint arXiv:1910.14424* (2019).

[6] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375* (2019).

[7] Ronak Pradeep, Rodrigo Nogueira, and Jimmy Lin. 2021. The expando-mono-duo design pattern for text ranking with pretrained sequence-to-sequence models. *arXiv preprint arXiv:2101.05667* (2021).

[8] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research* 21, 1 (2020), 5485–5551.

[9] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663* (2021).

[10] Tiancheng Zhao, Xiaopeng Lu, and Kyusong Lee. 2020. SPARTA: Efficient open-domain question answering via sparse transformer matching retrieval. *arXiv preprint arXiv:2009.13013* (2020).

[11] Weronika Łajewska and Krisztian Balog. 2023. From Baseline to Top Performer: A Reproducibility Study of Approaches at the TREC 2021 Conversational Assistance Track. In *European Conference on Information Retrieval (ECIR '23)*. Springer, 177–191. https://doi.org/10.1007/978-3-031-28241-6_12

A DIVISION OF WORK DURING THE PROJECT