Conversational Passage Retrieval

Group ID: 11

Riccardo Lunelli
University of Stavanger
Università degli studi di Trento
r.lunelli@stud.uis.no

Riccardo Parola
University of Stavanger
Università degli studi di Trento
r.parola@stud.uis.no

Michal Mareš
University of Stavanger
Czech Technical University in Prague
m.mare@stud.uis.no

ABSTRACT

This report delves into the domain of Conversational Passage Retrieval, a critical component in the functioning of conversational assistants. Our study focuses on developing a system capable of understanding conversational context and effectively retrieving relevant passages. Utilizing the MS MARCO Passages dataset from the Text Retrieval Conference (TREC) Conversational Assistance Track, we explore innovative methods to enhance retrieval performance. Our approach employed GPT-4 for query rewriting (QR), hypothesizing that its capabilities substitute the need for traditional finetuning of models based on BERT or T5. Additionally, we experimented with first-stage ranking methods, including BM25 and SPLADE, and utilized BERT for re-ranking (RR) passages. The study presents a comprehensive analysis of the results, highlighting the performance of our advanced methods, particularly the integration of QR with SPLADE. Our best pipeline, consisting of GPT-4 QR and SPLADE achieves the highest performance of all explored methods with Recall@1000=0.71 and MRR=0.36. To our surprise, adding an RR does not improve the performance. Overall, QR using GPT-4 has the most impact on the performance and is the main contribution of this work. The entire codebase of the project is available on GitHub[1].

KEYWORDS

information retrieval, machine learning, query rewriting, passage retrieval, first stage retrieval, document re-ranking, LLMs

1 INTRODUCTION

Conversational assistants are becoming part of our daily lives as well described in CAsT[5]. With Conversational Passage Retrieval (CPR) being one of the main building boxes of such systems, it has become more important than ever. In contrast to traditional IR where the system aims to return a list of documents based on a single query, in conversation retrieval we are able to use the conversation context, the user's questions, and possibly even the user's answers to generate follow-up queries to synthesize the final query. The final query is then used to select and rank relevant passages. This paper is going to explore the problem of CPR, set up a baseline method, and compare it with an implementation of an advanced method.

2 PROBLEM STATEMENT

Our main objective was to develop a system that understands the context of the conversation and effectively retrieves relevant passages from the collection. We are using MS MARCO Passages [3] document collection from the Conversational Assistance Track (CAsT) [5] of Text Retrieval Conference (TREC) as our dataset. In contrast to documents, passages are short texts at most 3 sentences long. Let us define a conversation history as the sequence of user queries and system responses, sometimes also called utterings. The input of our system consists of two parts: the raw passages $P \subseteq \mathcal{P}$ to be ranked and the conversation history $C \subseteq \mathcal{C}$, where $\mathcal{P}$ is the space of all possible passages and $\mathcal{C}$ is the space of all the possible conversations. The output of the system is a ranking of passages in P. Conversation Passage Retrieval system f can be considered a mapping

$$f : C \mapsto \mathbf{p} \tag{1}$$

where $\mathbf{p}$ is an ordered vector of passages in P.

For performance evaluation, we are using the following metrics: Recall@1000, Normalized Discounted Cumulative Gain at 3 (NDCG@3), Mean Average Precision (MAP), and Mean Reciprocal Rank (MRR).

3 RELATED WORK

The MS MARCO dataset, used in this paper as well as many others, presented by Bajaj et al. [3] could be considered the standard for CPR tasks. It contains queries, answers, and passages. Compared to other datasets, MS MARCO derives questions from real users' search engine queries representing human behavior more closely as it also contains questions with typos, complicated formulations, and even unanswerable questions. The downside is that passages connected to questions may not be exhaustive, meaning there could be another relevant passage for any of the questions on top of the ones provided.

The previously mentioned issue of assembling the query from a conversation is a crucial part of CPR and most of the works we analyzed use query rewriting (QR) to deal with it. This task can be easy enough for humans but poses a great challenge for computers. Anantha et al. [1] introduced the Question Rewriting in Conversational Context (QReCC) dataset, which enabled the community to train new models for the QR task. Another dataset, CANARD introduced by Elgohary et al. [6] contains human rewritten queries that are standalone and not dependent on the context anymore. As analyzed by Vakulenko et al. [16], numerous methodologies for QR have been proposed, which can broadly be categorized into two approaches: sequence generation and rule-based methods. The former often involves large language models (LLMs) like GPT-2, as demonstrated by Yu et al. [18] in their work. In contrast, the latter approach typically involves appending terms from the conversation history to the current question based on predefined rules. Lin et al. [11] conducted a comprehensive analysis comparing these two types of approaches, shedding light on the strengths and weaknesses of

[1] https://github.com/Pappol/conversational_passage_retrieval

each. Pronoun substitution and Seq2Seq LSMT-based model is used to asses this problem by the authors. Vakulenko et al. [15] decomposed the conversational question answering (QA) challenge into question rewriting and answering, with their architecture setting a benchmark on the TREC CAsT 2019 passage retrieval dataset. Their QR model also improved performance on the QuAC dataset, nearing human-level proficiency.

Once the queries are preprocessed by QR, relevant passages need to be retrieved or ranked. As it would be resource-intensive to rank the entire collection, initial retrieval is usually considered to lower the number of passages to be ranked. While most of the works use BM25 [13] as their first stage ranker for its speed, other methods can also be found, such as in the work presented by Formal et al. [9]. The proposed method SPLADE uses a sparse vector representation to solve the problem of synonyms and ambiguity in words. This allows to combine the advantages of bag-of-word (BOW) representation and dense representation.

After the initial retrieval, re-ranking (RR) of the subset of passages is needed. Kumar and Callan [10] presents a two-stage pipeline for both initial retrieval and reranking. The re-ranking stage, called Multi-View Reranking by the authors, constructs multiple "views" (interpretations) based on the conversation history and ranks the retrieved documents for each of them. Multiple rankings are then merged to form the final one. In other work, Nogueira and Cho [12] leverages the [CLS] vector of the BERT model to train a single-layer neural network (NN) to predict the relevance of the document to the query.

Other approaches such as ConvDR [19] explore the field of neural IR. Their primary focus is on improving the initial retriever within ranking pipelines. While dense embeddings have traditionally been effective in retrieval using approximate nearest-neighbor methods [17], there's a growing interest in sparse representations for both documents and queries.

4 BASELINE METHOD

For the baseline method, we are considering each conversational utterance as an individual query. Simple preprocessing combined with BM25 [13] score is used. Our text preprocessing pipeline was composed of the following steps:

- Lowercasing the text;
- Word tokenization with the `nltk.word_tokenize` function;
- Stopword removal, based on `nltk.corpus.stopwords`;
- Stemming using `nltk.stem.WordNetLemmatizer()`.

To rank the passages we have chosen Okapi BM25. It is simple to calculate and does not use any advanced mechanisms which will be used in the advanced methods.

We use the Okapi BM25 implementation provided by the rank-BM25 [4] python package which is based on the BM25 formula from [14] as follows:

$$rsv_q = \sum_{t \in q} log(\frac{N}{df_t}) \cdot \frac{tf_{td} \cdot (k_1 + 1)}{tf_{td} + k_1 \cdot (1 - b + b \cdot \frac{L_d}{L_{avg}})} \qquad (2)$$

The resulting value rsv_q, where q is the specified query, sums over terms $t \in q$. N is the size of the collections, df_t is the number of documents containing term t, tf_{td} is the occurrence count of term t in document d, L_d is the term-wise length of document d and L_{avg} is the average term-wise length over all documents in the collection. The remaining variables k_1 and b are parameters of the function, k_1 affects the term frequency scaling and b serves as the document length normalization. Those parameters were left to their default values in the used library, $k_1 = 1.5$ and $b = 0.75$.

The results are presented in Table 2.

5 ADVANCED METHOD

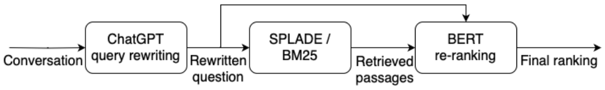

Figure 1: Digram describing our pipeline.

Here we present our advanced method: it consists of a pipeline of multiple building blocks, the diagram is presented in Figure 1. Each of the elements is presented in the following sections: QR using ChatGPT-4 in Section 5.1, first-stage ranking in Section 5.2, and finally, passage reranking using BERT in Section 5.3.

5.1 Query rewriting with ChatGPT

Inspired by the work presented in "Few-Shot Generative Conversational Query Rewriting" [18], our research aims to further advance the field by incorporating recent advancements. While the aforementioned paper used a GPT-2 model to enrich query context, we think that this approach has become somewhat outdated. The introduction of GPT-3.5 and GPT-4 models marked a significant leap in capabilities within this domain. While the initial work employed a few-shot learning approach with finetuning, our hypothesis is that with GPT-3.5 and above we do not need finetuning anymore. Instead, a well-crafted prompt can guide the model effectively, especially in tasks such as QR. Supporting our hypothesis, Zhou et al. [20] demonstrated that InstructGPT (all the GPT models from OpenAI are based on that) surpasses human performance in prompt engineering.

We decided to use ChatGPT-4 and craft our custom GPT instance for QR, which is available for ChatGPT Plus users[2]. It is interesting to note that the reformulated queries transition from interrogative forms to declarative statements.

```
Input message:
17_1,What are the different forms of energy?,17,1
17_2,How can it be stored?,17,2
17_3,What type of energy is used in motion?,17,3
17_4,Tell me about mechanical energy.,17,4

Query rewriting GPT response:
17_1,Different forms of energy,17,1
17_2,Methods for storing different forms of energy,17,2
17_3,Type of energy used in motion,17,3
17_4,Overview of mechanical energy,17,4
```

These statements encapsulate the essence of the original question, distilling the core intent and vital information required to address the query. This transformation inherently eliminates superfluous details, trims most stopwords, and introduces keywords potentially derived from these omitted stopwords. For instance, a query beginning with "Tell me about" is transformed to "Overview."

[2] https://chat.openai.com/g/g-R920EZnY6-query-rewriting-gpt

It's worth noting that documents are more likely to contain the latter expression, enhancing retrieval efficacy.

5.2 First-stage ranking

We compare two first-stage ranking methods. As the baseline, we consider BM25 once again as described in Section 4 with the difference being that this time, BM25 will receive rewritten questions as described in Section 5.1.

The second method, SParse Lexical AnD Expansion (SPLADE) [9] is a first-stage ranker building upon SparTerm [2]. SparTerm outputs the importance weight for each element in the vocabulary. The SPLADE paper proposes two main changes: logarithmic activation function and sparse regularization. This results in large performance improvements compared to the original paper. Here, the representation of a token sequence (e.g. document) is calculated by summing over the importance weights of the tokens after a ReLU function. The final (query, document) ranking score is obtained using the dot product of their representations. For more details, please refer to the original publication [9].

Both the code and model weights trained on the MS MARCO dataset are available online[3], which enables us to use this model without further training. For our model, we decided to employ the `naver/splade-cocondenser-ensembledistil` weights from the repository [7], which is based on a newer version of the SPLADE algorithm developed in [8].

5.3 Re-ranking using BERT

For passage re-ranking (RR), we decided to use the method proposed by Nogueira and Cho [12]. This approach leverages the [CLS] vector used in BERT LLM, which summarizes the input sequence. [CLS] vector of each document retrieved in the first stage and the [CLS] vector of the query are then sent to a single layer NN, which is trained using the cross-entropy loss to output the probability that the document is relevant to the query. The model is trained on the MS MARCO dataset and can therefore be used "as is", code and model weights were made available by the authors[4]. However, we decided to use the weights found on Hugging Face[5] from another author implementing the same paper for ease of use.

6 RESULTS

This section presents the results of our experiments, focusing on evaluating various retrieval methods. The results are summarized in Table 2, which compares the MAP, MRR, Recall@1000, and NDCG@3 across different methods. A bar plot is also presented in Figure 2 for better visualization. Note, that we are using shortcuts for describing the different pipeline blocks, QR stands for query rewriting and RR stands for re-ranking phase. Let us first focus on the BM25 methods. Both QR + BM25 and QR + BM25 + RR have the same recall, as the pipeline is shared up until after the first-stage ranking. However, the version with RR significantly improves all other performance metrics, for example, MRR improves by almost 50%.

Although using SPLADE on its own is better than the BM25 baseline as expected, it reaches competitive performance even compared to the combination of QR + BM25. This indicates just how powerful the SPLADE method is out of the box. This is further solidified by the fact that adding a RR phase after SPLADE first-stage retrieval doesn't improve any of the performance metrics, it does quite the contrary in fact, even though it had a significant impact in the case of the QR + BM25 pipeline. As a result, our best-performing model is the combination of QR using GPT4 and SPLADE. It is worth noting that while the QR + BM25 + RR has lower Recall@1000 by about 20% compared to QR + SPLADE, other performance metrics are comparable. This is visible in Figure 3.

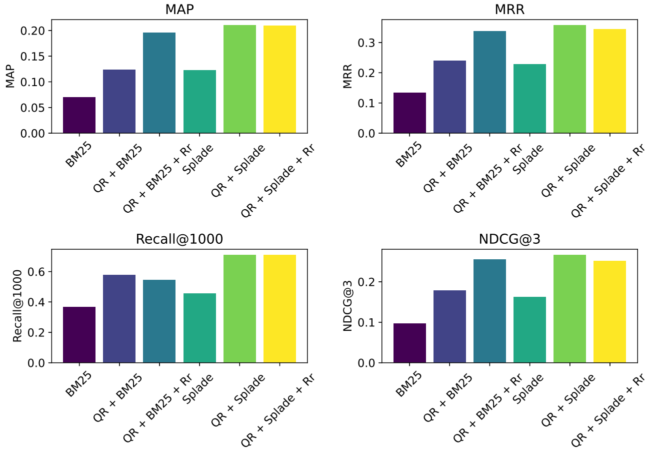

Figure 2: Scores of the different models.

Method	1^{st} docId	2^{nd} docId	3^{rd} docId
QR + BM25 + RR	7231103	3961766	5326236
QR + SPLADE	2253184	2253186	7731285
QR + SPLADE + RR	7231103	3961766	5326236

Table 1: Results of our proposed methods for the query 4_1: "What was the neolithic revolution?"

In Table 1 we can see the document IDs retrieved by the query "What was the neolithic revolution?". The first observation we made was that both the models that are using RR have the same results. This means that even the BM25 is, in this case, able to find in the first 1k elements a good set of documents. The second observation we made was that while the first two documents retrieved in all our models were all responding well to the query, the third response the model QR + SPLADE was better than the third of the other two variants (that has the same result). More in particular, the third document of the model using RR was responding more to a query like "What are the key stages of the Neolithic Revolution?". The text of the document is: *"The Neolithic Revolution. Chronology of the Neolithic Revolution: The Neolithic revolution took place in several stages. First, people settled down in permanent communities (sedentism), and*

[3] https://github.com/naver/splade
[4] https://github.com/nyu-dl/dl4marco-bert
[5] https://huggingface.co/amberoad/bert-multilingual-passage-reranking-msmarco

Method	**MAP** (train)	**MRR** (train)	**Rec@1000** (train)	**NDCG@3** (train)	**NDCG@3** (test)
BM25 (baseline)	0.0693	0.1341	0.3676	0.0973	0.077
QR + BM25	0.1238	0.2400	0.5796	0.1789	0.238
QR + BM25 + RR	0.1993	0.3498	0.5796	0.2571	0.392
SPLADE	0.1226	0.2272	0.4560	0.1630	0.247
QR + SPLADE	**0.2106**	**0.3579**	**0.7115**	**0.2670**	**0.419**
QR + SPLADE + RR	0.2092	0.3445	**0.7115**	0.2517	0.396

Table 2: Results of our proposed methods.

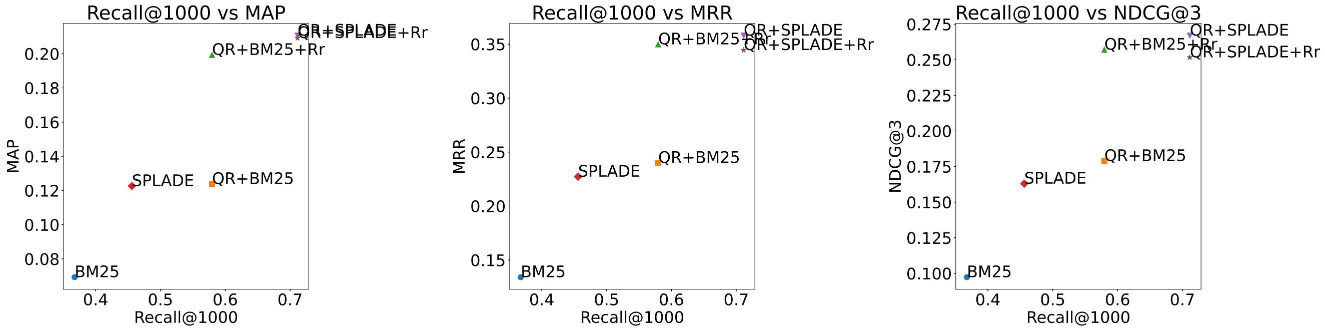

Figure 3: Scatter graph of Recall against all other metrics.

afterwards, they developed food production. Paleolithic before 10,000 BCE nomadic hunter-gathers of the Pleistocene (Ice Age)".

For the model QR + SPLADE the result on the third document was still directly relevant: "The Neolithic Revolution is the term given to the development of agricultural societies. This revolution in economic, political, and social organization began in the Middle East as early as 10,000 B.C.E. and gradually spread to other centers, including parts of India, North Africa, and Europe."

7 DISCUSSION AND CONCLUSIONS

In this study, we explored an approach combining our own version of query rewriting (QR) with two other methods, SPLADE and BERT re-ranking (RR). To the best of our knowledge, this combination is a novel approach in the field. Our experiments led to some interesting insights and important conclusions.

The most remarkable outcome of our experiments is the performance of the QR + SPLADE model. It's particularly noteworthy that the addition of RR to this combination did not yield the expected improvements. This suggests that SPLADE, when combined with our QR method, is already optimized to a level where further RR does not contribute any additional value, it may even lead to worse results. This finding was somewhat unexpected, as RR generally enhances retrieval results, as presented in our own QR + BM25 pipeline.

A key observation is the consistency in the performance of retrieval methods across both training and test datasets. This consistency is vital for ensuring the robustness and reliability of the retrieval models. Remarkably, the same retrieval systems that excel on the training set also demonstrate superior performance on the test set, maintaining their respective order in terms of effectiveness. This alignment between training and test results reinforces confidence in the generalizability of these models beyond controlled experimental conditions.

Upon closer inspection of the results, we noticed that some retrieved results are not present in the ground truth given by the MS MARCO dataset. The authors mention in the original publication [3], that the passages connected to questions may not be exhaustive. An example of such behavior is passage ID 2253184, which gives a concise description of the neolithic revolution, given as the 1st result of the QR + SPLADE pipeline. That effectively means that our reported performance is only a lower bound, and in reality, our model might perform even better.

Our results also highlighted the important role of the QR process in the setting of conversational information retrieval. The improvement in metrics such as MAP, MRR, and NDCG@3 with the integration of QR is proof of its effectiveness.

In conclusion, our work presents a consistent and well-working pipeline in information retrieval, showcasing the power of combining advanced query rewriting with efficient retrieval methods like SPLADE.

A DIVISION OF WORK DURING THE PROJECT

Riccardo Lunelli mainly worked on query rewriting and re-ranking. Riccardo Parola mainly worked on re-ranking and baseline. Michal Mareš mainly worked on baseline and first-stage ranking. Contribution to the final report was equally distributed among the group members.

REFERENCES

[1] Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. 2021. Open-Domain Question Answering Goes Conversational via Question Rewriting. arXiv:2010.04898 [cs.IR]

[2] Yang Bai, Xiaoguang Li, Gang Wang, Chaoliang Zhang, Lifeng Shang, Jun Xu, Zhaowei Wang, Fangshan Wang, and Qun Liu. 2020. SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval. arXiv:2010.00768 [cs.IR]

[3] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. In *InCoCo@NIPS*.

[4] Dorian Brown, Sarthak Jain, Vít Novotný, and nlp4whp. 2022. *dorian-brown/rank_bm25:*. https://doi.org/10.5281/zenodo.6106156

[5] Jeffrey Dalton, Chenyan Xiong, and Jamie Callan. 2020. TREC CAsT 2019: The Conversational Assistance Track Overview. *CoRR* abs/2003.13624 (2020). arXiv:2003.13624 https://arxiv.org/abs/2003.13624

[6] Ahmed Elgohary, Denis Peskov, and Jordan Boyd-Graber. 2019. Can You Unpack That? Learning to Rewrite Questions-in-Context. In *Empirical Methods in Natural Language Processing* (Hong Kong, China). http://umiacs.umd.edu/~jbg//docs/2019_emnlp_sequentialqa.pdf

[7] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2022. From Distillation to Hard Negative Sampling: Making Sparse Neural IR Models More Effective. https://doi.org/10.48550/ARXIV.2205.04733

[8] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2022. From Distillation to Hard Negative Sampling: Making Sparse Neural IR Models More Effective. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Madrid, Spain) *(SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 2353–2359. https://doi.org/10.1145/3477495.3531857

[9] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking. arXiv:2107.05720 [cs.IR]

[10] Vaibhav Kumar and Jamie Callan. 2020. Making information seeking easier: An improved pipeline for conversational search. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. 3971–3980.

[11] Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy Lin. 2021. Multi-Stage Conversational Passage Retrieval: An Approach to Fusing Term Importance Estimation and Neural Query Rewriting. arXiv:2005.02230 [cs.CL]

[12] Rodrigo Nogueira and Kyunghyun Cho. 2020. Passage Re-ranking with BERT. arXiv:1901.04085 [cs.IR]

[13] K. Sparck Jones, S. Walker, and S.E. Robertson. 2000. A probabilistic model of information retrieval: development and comparative experiments: Part 1. *Information Processing & Management* 36, 6 (2000), 779–808. https://doi.org/10.1016/S0306-4573(00)00015-7

[14] Andrew Trotman, Antti Puurula, and Blake Burgess. 2014. Improvements to BM25 and language models examined. In *Proceedings of the 19th Australasian Document Computing Symposium*. 58–65.

[15] Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2020. Question Rewriting for Conversational Question Answering. arXiv:2004.14652 [cs.IR]

[16] Svitlana Vakulenko, Nikos Voskarides, Zhucheng Tu, and Shayne Longpre. 2021. A Comparison of Question Rewriting Methods for Conversational Passage Retrieval. *CoRR* abs/2101.07382 (2021). arXiv:2101.07382 https://arxiv.org/abs/2101.07382

[17] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. *CoRR* abs/2007.00808 (2020). arXiv:2007.00808 https://arxiv.org/abs/2007.00808

[18] Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-Shot Generative Conversational Query Rewriting. arXiv:2006.05009 [cs.IR]

[19] Shi Yu, Zhenghao Liu, Chenyan Xiong, Tao Feng, and Zhiyuan Liu. 2021. Few-Shot Conversational Dense Retrieval. *CoRR* abs/2105.04166 (2021). arXiv:2105.04166 https://arxiv.org/abs/2105.04166

[20] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large Language Models Are Human-Level Prompt Engineers. arXiv:2211.01910 [cs.LG]