Reproducing and Extending: A Prompting Framework for Contextual Conversational Search with Large Language Models

Anesa Ibrahimi
University of Amsterdam
Amsterdam, Netherlands
anesa.ibrahimi@student.uva.nl

Amalia Stuger
University of Amsterdam
Amsterdam, Netherlands
amalia.stuger@student.uva.nl

Angel Bujalance Gomez
University of Amsterdam
Amsterdam, Netherlands
angel.bujalance.gomez@student.uva.nl

Abstract

In this work, we study the reproducibility of the paper *Large Language Models Know Your Contextual Search Intent: A Prompting Framework for Conversational Search* to verify the original results by Mao et al. [24]. *LLM4CS* leverages LLMs as a Search Intent Interpreter to understand the user contextual search intent in a multi-turn conversation context. We could verify most of the results of the original work. Our dense results using GPT3.5 are comparable to the original results. However, we faced some obstacles in reproducing some aspects of the paper. Additionally, we extended their original results by using an alternative LLM as the Search Intent Interpreter and replacing a dense with a sparse encoder. Our code is available at https://github.com/angelbujalance/LLM4CS.

Keywords

LLMs, Contextual Search, Information Retrieval, Indexing, Prompting Strategies

1 Introduction

Large Language Models (LLMs) have demonstrated enormous language understanding capabilities in generating text that mimics human-generated text [30]. Moreover, LLMs have been proven to be successful at getting contextual information from a multi-turn conversation [24]. This capability of LLMs is especially suitable for information retrieval as it potentially makes the retrieval of information more similar to a conversation with a human expert, where the user can ask some questions in natural language and start a multi-turn conversation.

Traditional IR methods faced problems that made them incapable of performing conversational searches. Classical search engines retrieve information based on keyword terms provided in the query. Moreover, these models could not retrieve references from previous turns in the current query. However, a method known as *Conversational Query Rewriting (CQR)* employs LLMs to rewrite the user's query into a de-contextualized form. After processing the query, ad-hoc search methods can be used to retrieve the information. Alternatively, *Conversational Dense Retrieval (CDR)*, encodes the user's real search intent and the passages data into dense representations, and performs dense retrieval Yu et al. [40].

LLM4CS framework makes use of LLMs to interpret the user query in a multi-turn conversation. The LLM model receives both the current query and the conversational context $C^t = (q^1, r^1, ..., q^{t-1}, r^{t-1})$, where q^i and r^i are the user queries and the search system responses at the i-th turn. The LLM is prompted with the query in turn t and the conversational context C^t to obtain contextual queries. The authors developed three prompting methods:

Rewriting Prompt (REW). The LLM is used as a conversational query rewrite tool and is prompted to generate the rewrite of the query q^t to contain the conversation context.

Rewrite-Then-Response (RTR). Previous research has shown that generating a hypothetical response to a query can improve retrieval performance. REW takes as input the rewrite to generate a hypothetical response.

Rewriting-And-Response (RAR). The RTR method generates query rewrites and responses in a two-stage manner. RAR generates the rewrite and the response in one stage.

Once the rewriting is finished, the information of the full conversation is aggregated to represent the user's search intent. The authors explored three different aggregation methods.

- **MaxProb.** The rewrite and the response with the highest probability are selected.
- **Self-Consistency (SC).** Selects the intent vector that is more similar to the cluster center of all intent vectors to be the final search intent vectors, it represents the most accepted search intent.
- **Mean.** The rewrite vectors and the generated response vectors are averaged. It considers more diverse information than selecting a single rewrite or response.

In this work, we aim to reproduce the findings of Mao et al. [24] and perform ablations to their original framework to establish the robustness of LLM4CS. Our work focuses on the following aspects:

- We reproduce the experiments conducted by the authors to verify the consistency of their results and the aspects where their framework was difficult to reproduce.
- We extend their experiments by replacing the dense encoding with a sparse vector encoding. Sparsity reduces inference time [22] and sparse representations are more interpretable with performance comparable to dense representations [6, 7].
- We test the method's robustness by replacing OpenAI's gpt3.5-turbo-16k as the Search Intent Interpreter for Llama3.1-8B-Instruct. We believe that many use cases cannot afford to access a closed model, and developers will focus on using open-source models for small LLM. For that reason, we wanted to test the robustness of LLM4CS in a budget-constrained scenario.

2 Related Work

Conversational Search. Involves the retrieval of information in multi-turn conversations between a user and a search system. Two main methods have been developed to perform this task: conversational dense retrieval and conversational query rewriting. On the one hand, conversational dense retrieval, developed by Yu et al. [40], encodes the full conversation search to produce dense retrieval. On

the other hand, query rewriting uses a generative model to rewrite session queries to retrieve the relevant contextual information of the full session [10, 26].

Moreover, conversation search can be considered as a two-step framework according to Gao et al. [9]. On the one hand, **Conversational Query Understanding** involves interpreting queries within the context of prior interactions in a conversation history, distinguishing it from traditional query understanding. There are two main approaches: methods based on traditional IR and natural language generation approaches, such as query rewriting. On the other hand, **Conversational Document Ranking** focuses on recent developments in ranking documents for conversational search. This task is defined as distinct from ad hoc retrieval ranking, with its corresponding benchmark and evaluation metrics.

Large Language Models General Capabilities. State-of-the-art LLMs are very successful at complex everyday tasks as they are capable to adhere to prior instructions. For instance, GPT-4 can outperform most humans on some academic or professional tasks [30]. In recent years, LLMs have been proven successful in natural language processing tasks, such as question answering, reading comprehension, summarization, and machine translation [1, 30]. Specifically, Llama 3 performs on par with leading language models like GPT-4 [30], and it is very close to match the state-of-the-art. Moreover, Llama-3 presents a series of small models that outperform models with a similar number of parameters [1]. Llama 3 models are open-released to stimulate innovation in the research community [1].

Use of LLMs in IR. As we have already pointed out, LLMs have shown impressive capabilities in different domain and can perform at a human level on some case. For that reason, the generation and understanding abilities of LLMs have been applied previously to IR tasks. In particular, LLMs have previously been used in IR to improve the retrieval process by query generation, document prediction, and query expansion. LLMs can enhance the generation quality of their responses by first retrieving and then using the generated responses to increase the original search query and improve the retrieval process. LLMs have also been used to improve the passage retrieval task, outperforming traditional IR methods [29] as deep learning methods allow us to better understand the complex patterns that exist in natural language and acquire contextual knowledge within the text that cannot be acquired with bag-of-words retrieval models [4, 20, 23].

In this work, LLMs are used to generate rewrites of the original user queries to contain the user's contextual history. For instance, if a user references a term in the current query, the LLM will be capable of retrieving this term in the current query rewrite. Furthermore, the authors wanted to explore whether additionally using the LLM to generate a hypothetical response improves the retrieval process. Previous research has shown that using a LLM to generate a hypothetical response to a user query improves the retrieval performance. For instance, Mao et al. [26] generated relevant context using LLM without external supervision improved the query information and improved the retrieval accuracy. Similarly, Gao et al. [10] fed the user's query to a generative model to write a document that answers the question, generating a hypothetical document. The generated document may not be factual and could contain errors; it served as a proxy for a relevant document. In the subsequent step, an unsupervised contrastive encoder was employed to encode the generated document into an embedding vector.

Dense Encoders. Dense retrieval has demonstrated remarkable empirical performance across a range of ad hoc search tasks and open-domain question-answering (QA) benchmarks, as evidenced in prior studies [2, 19]. Unlike traditional sparse retrieval methods that rely on sparse word representations, dense retrieval utilizes pre-trained neural models, such as BERT [18], to encode both queries and documents into dense embeddings. These embeddings enable retrieval to be conducted entirely within a dense representation space [17], which allows for more nuanced and context-aware matching. By employing advanced fine-tuning techniques, dense retrieval approaches like ANCE [37] achieve substantial improvements in performance compared to their sparse counterparts. This superior performance underscores the advantages of dense retrieval in capturing semantic relationships that may be overlooked by sparse methods. However, the process of learning an effective dense representation space typically requires an abundance of high-quality relevance labels. This dependence poses challenges as the development of conversational dense retrieval is affected by the lack of data [25]. Mostly because conversational search systems have not been deployed in practice, and real conversational search sessions are hard to obtain to train these types of models [40].

Sparse encoders. Most information retrieval (IR) systems commonly employ sparse ranking methods, such as BM25, for first-stage retrieval. Sparse retrieval relies on matching query terms with document terms to retrieve relevant documents. Specifically, it builds an inverted index composed of all documents in a corpus, to encode aspects such as document length, term position, or term frequencies [44]. Traditional methods like TF-IDF and BM25 [32] perform by employing term frequency and exact word matching [44]. However, such systems often struggle with the vocabulary mismatch problem, where queries and documents use different terms to convey the same meaning, thereby limiting their effectiveness in capturing semantic equivalence [35]. To mitigate these problems, neural networks approaches such as Word2Vec enhances the semantic understanding by improving the semantic similarity measurement. Alternatively, deep learning models can be used to expand possible query terms. For instance, DeepCT utilizes BERT to extract context-based term features and leverages them to predict the importance of each term through a supervised per-token regression task [3]. In few-shot contexts, sparse methods may retain certain advantages, highlighting a critical area for further investigation and methodological refinement [35].

Reproducibility in Artificial Intelligence. Reproducibility in Artificial Intelligence consists of the ability of an independent research team to obtain the same results using documentation or code provided by the original research team [13]. However, AI researchers experience problems when they want to reproduce the results of others [14]. Some of the reasons for this discrepancy are: lack of access to the original training data, errors or lack of availability of the original code to perform the experiments, or a selective report of results [31]. Gundersen and Kjensmo [13] identifies three different degrees of reproducibility in artificial intelligence. **Experiment reproducible.** When the implementation of an AI method generates the same results in the same data. **Data reproducible.** The results are data reproducible if an alternative implementation of

the AI method generates the same results in the same data. **Method reproducible.** The method is reproducible when an alternative implementation of the AI method generates the same results on a different data set. Method reproducibility only requires the documented method and guarantees that the experimental results can be replicated without being influenced by hardware, data noise, or implementation details. This establishes that the results are solely attributed to the AI method. By using a new implementation and a dataset, the AI method demonstrates generability beyond the conditions of the original experiment. In terms of generalizability, Method reproducibility is the highest degree of reproducibility followed by Data reproducibility which means that Experiment-reproducible results are less general than data and method-reproducible results.

3 Methodology

3.1 Reproduction Phase

As mentioned above, the LLM4CS framework contains three prompting methods (Rewriting Prompt (REW), Rewriting-Then-Response Prompt (RTR), and Rewriting-And-Response Prompt (RAR)). Given that the authors of this work provided the preprocessed cast19, cast20, and cast21 datasets, the prompting phase was the next step to generate the rewrites and/or hypothetical responses. This step involved a fairly direct process where the already-included prompting scripts were executed alongside our personal OpenAI API key. Each of the scripts produced rewritten queries, stored in JSON format, which reformulated the conversational queries into more contextually-appropriate queries. Additionally, the generated rewrites were accompanied by an additional layer of generated responses if RTR or RAR strategy was used. Following the creation of the prompts, the evaluation step was carried out. This step itself acquired several design choices and adjustments to the original guidelines provided by the authors and their documentation. In more detail, the evaluation phase involved the usage of three aggregation methods (MaxProb, Mean, SC) to obtain the final search intent vector to perform dense retrieval with ANCE [37]. In the author's documentation, the first potential issue that interfered with reproducibility was the outdated link for downloading the ANCE checkpoints that have been used throughout their evaluation procedure. Access to these pre-trained checkpoints is critical for verifying the results of the repository and storing pre-trained dense representations for passage retrieval. Thus, its absence causes the user to be unable to replicate the experiments in order to verify the claims made in the documentation. The outdated ANCE checkpoint link revealed critical reproducibility and maintenance challenges within the LLM4CS repository.

Moreover, the authors instead re-directed the user to three public repositories to build a dense index and therefore did not provide their own framework for Dense retrieval. These re-directions and out-of-date links turned into inconvenience, unfortunately, due to the large differences and varying approaches that were found in the public repositories themselves. More specifically, two (ConversationQueryRewriter [39], ConvTrans [25]) out of the three repositories (ConversationQueryRewriter [39], ConvTrans [25], ConvDR [41]) contained specific training pipelines and objectives that did not serve as a foundational and adequate framework to build the dense indexes for the LLM4CS framework. Firstly, the methodological differences between the two above-mentioned repositories posed as a barrier to successfully being able to integrate it into the existing framework. The reason being, that the work laid in [39] focused on generative conversational query rewriting. Implying that the main aspect of the work was to rephrase queries rather than optimize retrieval tasks. This in return, placed the emphasis on the pre-processing stage rather than the retrieval (dense representation) task. The other work, [25], aimed at transforming web search sessions into conversational search sessions in order to then train its encoder to be able to perform dense retrieval. Therefore, focusing on session-level encoders for dense retrieval tasks. Even though the ConvTrans pipeline ([25]) operates at a session level, it still synthetically generates conversations on the basis of fixed patterns rather than the open-ended and dynamic prompts which are used in LLM4CS. Consequently, neither of the repositories aligns with LLM4CS's need for real-time, generative, and open-ended contextual understanding. Both of the repositories introduced rather rigid pre-processing and re-engineering steps that conflicted with LLM4C's pipeline. The third mentioned work, ConvDR [41], on the other hand, served to a certain extent as a helpful framework to build dense indexes & passage embeddings before running the evaluation step. However, when following their provided series of steps, the data preparation phase remained as faulty as the LLM4CS pipeline. The pre-trained ANCE checkpoints of this repository directed the user to an outdated link. In contrary to the LLM4CS repository, the authors did provide a work-around solution which involved a re-direction to an existing model provided by Huggingface. This solution though came with a great cost, as it was emphasized by the authors that several adjustments in their existing codebase was required accordingly without explicit steps or pointers. In return, after making the necessary adjustments (ourselves) in their tokenizing and embedding phase to accommodate the Huggingface model version of ANCE, we were able to generate the dense passage embeddings. When 'injecting' these newly-constructed embeddings to the evaluation script and phase of the LLM4CS codebase, a series of errors were occurring during the generation of the final evaluation metric values (MRR, NDCG@3, R@100). As highlighted in their repository, the dense embedding generation phase requires a large and inconvenient amount of memory (200GB) which was unattainable given our resources. A workaround to this issue, was to instead use a subset of the 38M CAsT passages which contained a series of documents from TREC CAR, MSMARCO, and the Washington Post. We initially extracted a 10k subset via the first 10k entries which we later deduced it unfortunately contains only passages from TREC CAR and therefore causing mismatches with the qrel dataset of LLM4CS which contained only MSMARCO passages. After re-creating the cleaned subset that contained only MSMARCO passages instead, the final generated metric values remained zero-valued implying there is a mismatch between the generated embeddings from our sub-collection with the pre-processed CAsT qrel dataset. Alternatively, we resorted to another public repository and work, PCIR [28]. As it included a ready-to-use framework to generate dense passage embeddings for the purpose of evaluating reformulated queries (as is the case in LLM4CS), we integrated the following algorithm (1) to effectively generate a sub-collection from python package

Algorithm 1 Create Sub-Collection and Map Document IDs to Row IDs

Input: `trec-cast/v1/2019` dataset from `ir_datasets`
Output: `trec_cast_2019_subcollection.tsv` and `2019_rowid_qrel.tsv`
Load the dataset using `ir_datasets.load("trec-cast/v1/2019")`
Initialize `judged_docs` as a set of document IDs from `qrels_iter()`
Open `trec_cast_2019_subcollection.tsv` for writing
Initialize `row_id` to 1
for all documents `doc` in `docs_iter()` **do**
    if `doc.doc_id` is in `judged_docs` **then**
        Replace tabs and newlines in `doc.text` with spaces
        Write `row_id` and cleaned text to the file
        Map `doc.doc_id` to `row_id` in `originalid2rowid`
        Increment `row_id`
    end if
end for
Close the file
Open `2019_rowid_qrel.tsv` for writing
for all qrel entries `qrel` in `qrels_iter()` **do**
    if `qrel.doc_id` is in `originalid2rowid` **then**
        Map `qrel.doc_id` to `row_id` using `originalid2rowid`
        Write `qrel.query_id`, `qrel.iteration`, `mapped_row_id`, and `qrel.relevance` to the file
    end if
end for

ir_datasets. This approach seamlessly shuffles the documents of various sources with each-other creating an enhanced collection that can effectively test the retrieval capabilities of the LLM4CS framework. Despite resorting to the work introduced in PCIR, the mismatch between the generated embeddings and the qrel documents of CAsT dataset was still persistent. Ultimately, resorting to a famous widely-used python framework and tool-kit *Pyserini* [21] changed the trajectory of our reproducibility path. As emphasized by the authors of this tool-kit, the design of Pyserini enables an ease of use and various reproducibility opportunities. As this tool-kit provides ready-to-use pre-trained indexes and encoders it successfully was able to convert queries into dense representations. Moreover, it also provided pre-configured evaluation scripts for widely used test collections including MS MARCO which aligned with the pre-processed dataset of the LLM4CS work. Therefore, when utilizing all of these attributes and modular parts of the Pyserini framework, it became the backbone of the evaluation phase in the LLM4CS framework, seamlessly supporting its dense retrieval tasks and ensuring reproducible and reliable results.

3.2 Extension Phase

Building upon the original work, sparse retrieval and an alternative (smaller) open-source LLM served as an extension and expansion of the LLM4CS's applicability. Before delving into the specific and necessary changes that we undertook to facilitate these extensions it is important to initially highlight some key adjustments made in the aggregation methods of the original work. During the evaluation phase, we implemented both the Mean and MaxProb aggregation methods as described in the original paper. Due to resources and time constraints, however, we were unable to include the third aggregation method (Self-Consistency) in our experiment and evaluation phase.

There were several adjustments needed to be carried out to ensure the compatibility of the two aggregation methods with the evaluation phase. To implement MaxProb for the REW method, we selected the first generated rewrite. For the RTR and RAR methods, we instead concatenated the selected rewrite and hypothetical response. For Mean aggregation method, we concatenated all responses for REW, while for RTR and RAR, we concatenated all rewrites and hypothetical responses. These decisions were guided by the input requirements of the sparse encoding method we employed, and we extended the same approach to dense encoding to ensure comparability of results across both methods. A key limitation of this approach is that we could not use the original evaluation pipeline provided by the authors due to compatibility issues with the dense encoding process. Instead, we developed our own evaluation pipeline, which incorporated the custom aggregation methods described above and the Pyserini framework. While our aggregation methods performed sufficiently for our experiments, they may lack the fine-tuned optimization of the original pipeline, potentially introducing deviations in performance metrics. Furthermore, the concatenation approach may have introduced redundancy or diluted the focus on the most relevant query components, affecting the interpretability and precision of retrieval outcomes.

To incorporate and substitute the original dense encoder of the original framework, sparse encoder was introduced and implemented through Pyserini toolkit. This adjustment therefore was effectively enabled by Pyserini's flexible support for sparse retrieval using BM25 scoring while leveraging Lucene [8] as its backbone. Pyserini's ability to handle pre-built sparse indexes and provide access to document vectors and term statistics further allowed the incorporation of the sparse encoder, ensuring it aligned with the original framework.

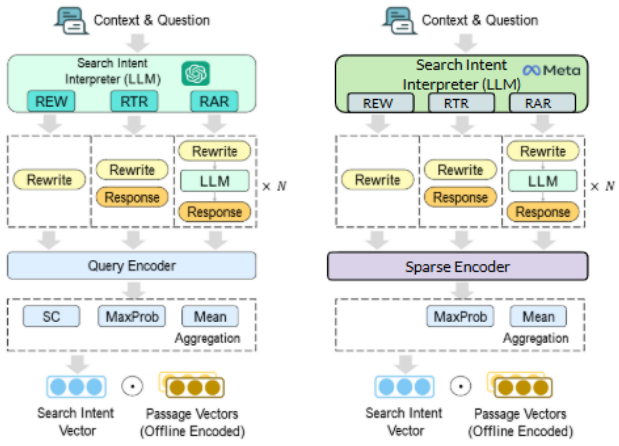

Figure 1: Overview of Original Work (Left) & Extension (Right)

Encoder - LLM	Aggregation	CAsT-19			CAsT-20		
		REW	RTR	RAR	REW	RTR	RAR
Dense-GPT3.5 (Original)	MaxProb	0.441	0.459	0.464	0.356	0.415	0.430
	Mean	0.447	0.464	**0.488**	0.380	0.425	**0.442**
Dense-GPT3.5 (Pyserini)	MaxProb	0.469	0.555	0.558	0.406	0.457	0.442
	Mean	0.458	0.555	0.558	0.386	0.457	0.442
Sparse-GPT3.5	MaxProb	0.278	0.399	0.405	0.200	0.384	0.351
	Mean	0.289	0.405	**0.439**	0.229	0.417	**0.402**
Sparse-Llama	MaxProb	0.259	0.395	0.070	-	-	-
	Mean	0.326	**0.396**	0.069	-	-	-

Table 1: Performance comparisons with respect to NDCG@3 using different prompting and aggregation methods. The best combination on each dataset is bold.

In addition to introducing an alternative to the original dense encoder, a smaller open-source LLM was integrated into the framework to inspect and test its impact on computational efficiency and retrieval performance. More precisely, the open-source LLM was integrated into the framework to inspect and test its impact on computational efficiency and retrieval performance. The model , *Llama3.1-8B-Instruct*, originates from a larger set of foundation models for languages (Llama 3 [11]) which is fundamentally designed for reasoning, multilinguality and many other tasks. The smaller variant which we made use of inherits the dynamic pre-training and pos-training strategies of the LLama 3 family. Essentially, the choice of this smaller LLM model was driven by its open-source nature, computational efficiency as well as its ability to handle specialized requirements such as long-context windows and reasoning-heavy tasks.

4 Experimental Setup & Results

When referring to table 1 , an extensive evaluation of the performance of various encoders and LLM models on the CAsT-19 and CAsT-20 datasets has been provided. In order to grasp the values presented in the table, it is important to understand the experimental setup and specific configurations that were attached to each of the models. As mentioned thoroughly in the Methodology section, after generating the appropriate rewrites per prompting strategy the evaluation of dense retrieval was the subsequent step. Instead of using the evaluation framework provided in the existing codebase of the original work, we have resorted to using Pyserini's built-in evaluation and interpretation tools for both sparse and dense retrieval methods. When using this framework we were able to compare in a consistent matter the different encoding methods and were able to closely match the values obtained in the original work.

The first row of the table, showcases the performance of original work with Dense retrieval with the GPT3.5 model on the two aggregation methods (MaxProb and Mean). These results were obtained by the authors through the usage of the full collection of TREC-CAst which contained a total of 38 million passages. A key observation to be made when inspecting the first row is that the NDCG@3 metric value is the highest and best performing for both of datasets when using the RAR prompting strategy alongside the mean aggregation method. The second row on the other hand are the results we aimed at obtaining when reproducing the original work. It is important to emphasize certain configurations that were made to obtain the values. More specifically, we tested and generated the dense embeddings and indexes on a sub-collection instead. This design choice , as mentioned and argued in the Methodology section, was constructed via algorithm 1. In comparison to the 38 million passages , a total of 21726 passages were generated for CAsT-19 and a total 29264 passages for CAsT-20. Furthermore, the reproducibility and extension results were generated via single-node GPU cluster. Additionally the evaluation and embedding generation steps were all conducted via the pyserini framework itself which was independent from the LLM4CS framework. When further inspecting second row, relatively higher results were obtained for all of the prompting and aggregation methods on both datasets in comparison to the original work (first row). Contrary to the original results, our reproducibility study does not exhibit the same domination of one prompting strategy or configuration in comparison to the others. In other words there are no apparent differences between RTR and RAR as opposed to the original work which showed best results via the RAR prompting method. Moreover it can observed that both RAR and RTR outperform REW relatively more than in the original work. Lastly, the most effective combination during our reproducibility study is obtained via the RAR prompting strategy alongside *both* Mean and MaxProb aggregation methods.

The last two rows of the table showcase our extension efforts. It is important to highlight that the values obtained during our extension stage with sparse retrieval were conducted and reliant on the full collection set (38 million) as opposed to the dense retrieval (reproducibility) values which solely relied on the sub-collection. The entries found under the *Sparse-GPT3.5* section were generated using the same LLM model (GPT3.5) as the original work alongside the sparse encoder of pyserini framework instead. The values on both datasets are relatively lower than the original and reproducibility study when using this specific encoder-llm configuration. However, a similar behavior to the original framework can be observed when viewing the best performing NDCG@3 value. The best combination on each dataset is the RAR and mean aggregation method which is identical to the original work. When viewing the last entry which aimed to effectively combine our two proposed extensions simultaneously the subsequent results were obtained. Similar to the row above it, *Sparse-llama* contains lower metric values than the original and reproducibility study. Upon further comparisons, it can be seen that this configuration in fact contains the lowest performing values out of all the entries in the table. When isolating this model from

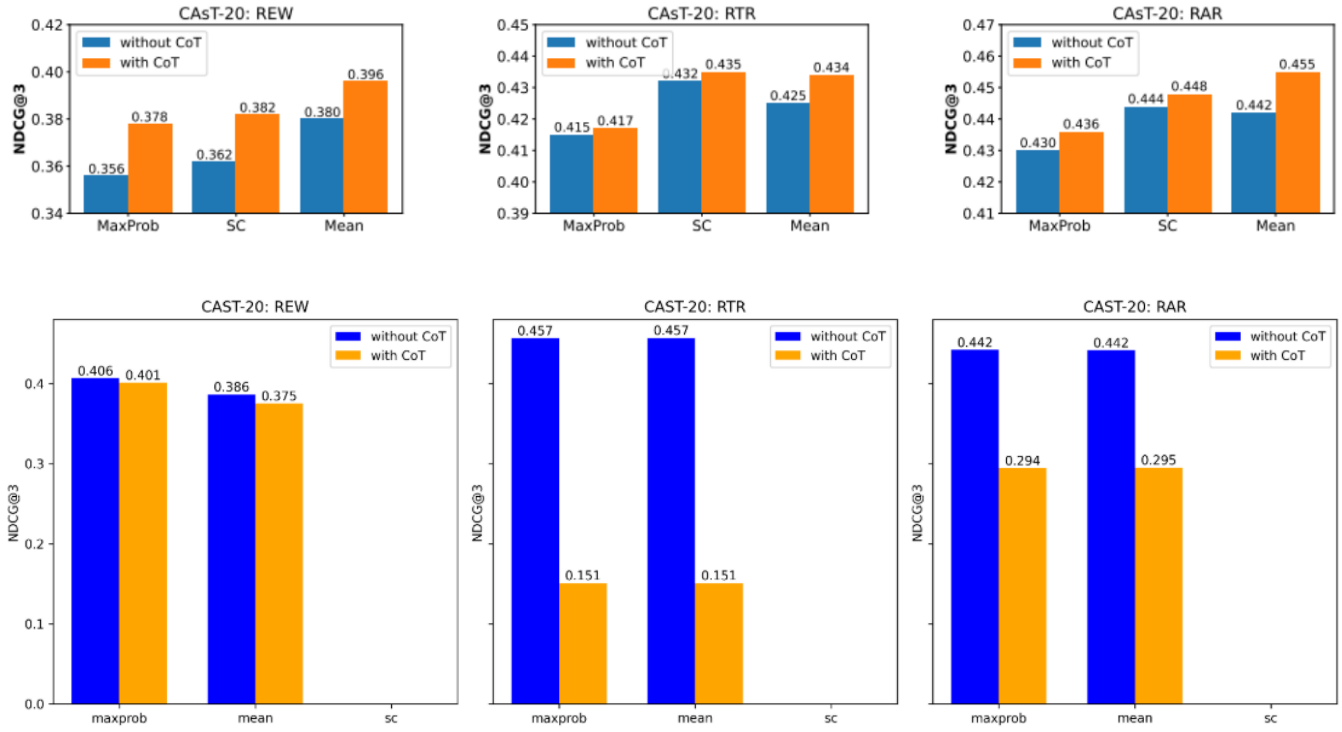

Figure 2: NDCG@3 comparisons between incorporating LLM4CS's tailored CoT or not across different prompting and aggregation methods on CAsT-20 dataset.

others, the best combination of CAsT-19 was when utilizing RTR and mean aggregation method. The non-available results for CAsT-20 were directly caused by the noisy prompting caused during the usage of the llama model. Moreover, several implementation-wise errors and faulty behavior were present when trying to use this model configuration to generate the adequate prompting rewrites. Consequently, preventing us from evaluating and comparing these prompting strategies with the generated sparse embeddings.

Aside from the table, a set of bar plots above (Fig.2) was generated. As mentioned in the original LLM4CS work, CoT prompting was introduced before generating the rewrites and/or responses to ultimately investigate its impact on the reasoning of the contextual search intent. The three bar plots at the top (blue and orange color) are directly extracted from the original research , whereas the plots beneath (blue and yellow) are the ones obtained via our reproducibility study. As can be seen in the top row , the incorporation of the custom-made CoT prompting method of the authors positively impacted the search performance for all prompting and aggregation methods. The improvements can be particularly notable in the REW prompting method compared to the other methods. As stated by the authors themselves [24], the introduction of several hypothetical responses enhance the capabilities of the search intent vector and ultimately search performance. Conversely, the plots that were generated during our reproducibility efforts via the Dense encoder alongside GPT3.5 model using the sub-collection show other behavior. As mentioned earlier, the last column/entry of the bar plot which accommodates the self-consistency aggregation method has not been able to be reproduced when generating embeddings due to faulty formatting issues and adjustments needed to be made to be compatible with the Pyserini framework [21]. Therefore, when focusing only on the two remaining aggregation methods, the performance of all prompting & aggregation methods with CoT were lower than that without CoT. In fact, as opposed to the results of the original work, there is a great discrepancy and difference between the performances of RTR and RAR methods when inspecting the impact of CoT prompting. Even though the CoT prompting effect is higher on RAR prompting than RTR, its removal proved to be more optimal. One common pattern that can be drawn from our reproducibility of the CoT impact, is the increased performance found in REW prompting when integrating the CoT generated responses. Despite the slight difference between performances between non-CoT and CoT prompting, the introduction and enhancement of hypothetical responses from chain-of-thought does not go unnoticed. While these results demonstrate the strengths of different models and prompting methods, the underlying factors contributing to these outcomes and their broader implications will be discussed in the upcoming sections below.

5 Discussion

5.1 Challenges in Reproducibility

Our study reveals several reproducibility challenges with the LLM4CS framework [24]. One significant issue was the reliance on outdated resources, such as non-functional ANCE checkpoints [36], which hindered our ability to replicate the dense retrieval process as described in the original paper. Additionally, the lack of a dedicated framework for dense retrieval forced us to explore external repositories, which often deviated significantly from the requirements of the LLM4CS pipeline. These challenges highlight a broader issue in reproducibility within the AI domain—the importance of maintaining up-to-date resources and providing modular, self-contained codebases.

The incompatibility of the provided preprocessed datasets with the generated embeddings also underscores the need for clear documentation and seamless integration of all pipeline components. Initially, we attempted to use the ConvDR repository along with another repository referenced by the original authors as solutions for dense retrieval encoding. However, both repositories posed significant challenges. The ANCE checkpoint link provided by the authors was non-functional, which further impeded our efforts. Additionally, the frameworks themselves proved difficult to implement due to mismatches in the mapping of the datasets and embeddings, rendering them unusable in our experimental setup.

To resolve these issues, we explored the PCIR framework[28], which was developed by the original authors and thus seemed a logical alternative. Despite its potential, PCIR also encountered mapping problems within our pipeline and failed to produce usable results. This persistent issue highlighted the limitations of available frameworks for dense retrieval tasks, even when authored by the same team.

As a final approach, we turned to Pyserini [21], leveraging its modular capabilities and pre-built indexes for dense retrieval. Pyserini provided partial relief by enabling some progress; however, its dense encoding process posed significant memory and disk usage challenges. These constraints required adjustments to our computational setup, including attempts to increase the available RAM and implement variations in the encoding process, such as adding checkpoints, in consultation with our TA. Despite these efforts, time constraints prevented us from fully addressing these challenges.

The large size of the CAST-19 and CAST-20 datasets for dense encoding was a significant contributing factor to the computational issues encountered. Processing these datasets required substantial disk space, as the embeddings generated during dense retrieval consumed significant storage resources. Additionally, memory constraints posed repeated bottlenecks during experimentation. Efforts to resolve these limitations included upgrading RAM capacity and segmenting the encoding process into smaller batches, but these measures proved insufficient given the scale of the datasets and the limited computational resources available. Checkpoints were introduced to save intermediate states and reduce redundant computation in cases of interruptions; however, this only partially mitigated the challenges and required extensive manual intervention to align with the pipeline.

Ultimately, due to time constraints and the persistent computational bottlenecks, we opted to proceed by reproducing results on a smaller subset of the datasets. This decision was made as a practical trade-off to ensure the timely completion of our reproducibility study. The subset approach allowed us to avoid the memory and storage issues associated with processing the entire dataset. However, while this approach yielded numerically better results than those reported in the original paper, these improvements can largely be attributed to the reduced dataset size and the absence of matched negatives, which artificially inflated performance metrics. The lack of matched negatives, in particular, simplified the retrieval task and introduced a bias toward higher scores, highlighting a critical limitation of our subset-based approach.

This outcome emphasizes the need for future work to address the limitations of dense retrieval in large datasets systematically. For example, testing on multiple random subcollections could provide more robust insights by addressing variability concerns and ensuring that results are representative across different configurations of the dataset. Additionally, adopting scalable computational strategies, such as distributed processing or cloud-based solutions, could enable the efficient handling of full-scale datasets. Incorporating frameworks specifically designed for dense retrieval tasks, with detailed documentation and modular architectures, would also help mitigate the challenges encountered in our study. Such approaches would not only enhance reproducibility but also improve the scalability and reliability of information retrieval research pipelines.

5.2 Performance Analysis

Our performance analysis revealed notable differences compared to the results reported in the original paper. Due to computational constraints, we conducted dense retrieval experiments on a smaller subset of the datasets, which likely influenced the results. While we achieved higher performance metrics than those reported in the original paper, these improvements can be attributed to the smaller subset size and the absence of matched negatives, which simplified the retrieval process. Despite these methodological differences, we observed that both RAR and RTR prompting strategies outperformed REW, consistent with the original findings. However, unlike the original paper, where RAR significantly outperformed RTR, we found no performance difference between RAR and RTR. This discrepancy is likely due to the reduced dataset size, which may have limited the distinctions between these two methods, given that smaller datasets are less representative and may fail to capture nuanced differences.

To complement the dense retrieval experiments, we employed sparse retrieval methods on the full CAST-19 and CAST-20 datasets. Sparse retrieval was computationally more efficient, easier to implement, faster to run, and more interpretable compared to dense retrieval. However, sparse retrieval exhibited lower performance metrics across all datasets, aligning with expectations. This decrease in performance can be attributed to inherent limitations of sparse indexes, including a lack of semantic understanding [27] and poor context handling, which are critical for conversational search tasks. These characteristics likely contributed to the poorer results observed for sparse retrieval methods, particularly when compared to dense retrieval.

The performance of the open-source Llama3.1-8B-Instruct [1] model further underscored the challenges of using smaller models

for complex information retrieval tasks. Llama exhibited noisy outputs, especially when combined with the RAR prompting strategy on CAST-20, where the results were so unreliable that they could not be meaningfully evaluated. For CAST-19, the sparse retrieval results with Llama demonstrated significantly lower performance, as seen in Table 1. These findings highlight the limitations of smaller models like Llama, particularly in handling complex contextual information and maintaining consistency across multi-turn conversational tasks.

In contrast, the original paper reported that GPT-3.5 [30] demonstrated superior performance across all methods, including RAR, RTR, and REW, with RAR consistently achieving the highest scores. The lower performance of Llama, combined with its tendency to generate noisy outputs, is unsurprising given its significantly smaller model size and reduced capacity for contextual understanding.

The trade-offs observed between dense and sparse retrieval methods, as well as between smaller and larger models, emphasize the importance of evaluating retrieval strategies and model performance across diverse datasets and configurations. While sparse retrieval may offer computational advantages, its limitations make it less suitable for tasks requiring deep semantic understanding and robust contextual reasoning. Similarly, smaller models like Llama provide a cost-effective alternative but require additional fine-tuning or architectural adjustments to handle the complexities of conversational search effectively.

5.3 Chain-of-Thought Prompting

The original paper's claims regarding the positive impact of chain-of-thought (CoT) prompting were only partially supported by our findings. In our experiments, we tested CoT prompting exclusively on dense retrieval methods, as other configurations, including sparse retrieval and the Llama3.1-8B-Instruct model, were either outside the scope of our study or unable to produce the required prompts. Specifically, Llama was unable to generate the structured prompts necessary for CoT due to its limitations in handling complex prompt engineering tasks.

For dense retrieval, our results indicated a decrease in performance when using CoT prompting, contrary to the improvements reported in the original paper. This discrepancy may stem from the experimental setup differences, particularly our use of a smaller subset of the datasets. CoT prompting relies on sufficient data coverage to establish nuanced reasoning chains across conversational turns. With a reduced dataset, the structured reasoning CoT provides may have been underutilized or even detrimental, as it introduces additional complexity without sufficiently improving contextual understanding.

Another possible explanation lies in the alignment between CoT prompting and the evaluation framework. CoT may require a more robust dataset and retrieval environment to fully leverage its potential, which our subset-based approach may have failed to provide. The smaller dataset may have resulted in incomplete conversational contexts, leading to reasoning chains that lacked coherence or relevance, thereby undermining performance.

Our findings suggest that while CoT prompting has the potential to enhance performance, its effectiveness is highly sensitive to dataset size and configuration.

5.4 Dataset Selection and Limitations

Following our TA's recommendation, we decided to exclude CAST-21 [5] from our experiments and focus solely on CAST-19 and CAST-20. While the original authors of LLM4CS conducted experiments on CAST-21, our decision was guided by practical considerations aimed at ensuring the reproducibility and reliability of our study within the constraints of our resources and timeline. CAST-19 and CAST-20 offered well-established baselines with thoroughly vetted annotations and evaluation protocols, making them better suited to the scope of our project and ensuring alignment with the reproducibility objectives.

The recommendation to prioritize CAST-19 and CAST-20 stemmed from their established use in prior studies and their suitability for reproducing the core results of the LLM4CS framework. While CAST-21 was included in the original study, integrating it into our experiments would have introduced additional complexities due to its more recent nature and the associated learning curve. These factors could have added unnecessary variability and time investment, potentially detracting from our ability to fully address the reproducibility challenges inherent to the LLM4CS framework.

Time constraints and practical limitations were also significant factors in this decision. Given the finite duration available for conducting our experiments and documenting findings, focusing on datasets with more established workflows allowed us to allocate our efforts effectively. CAST-19 and CAST-20 provided the necessary foundation to reproduce key results, while excluding CAST-21 streamlined our study and ensured that our findings were robust and reliable within the context of the reproducibility objectives.

While CAST-21 offers valuable opportunities for extending the scope of conversational search research, it was not strictly necessary for replicating the core results reported in the original paper. Our approach ensured a more targeted and efficient study, aligning with our objectives and resource constraints. Future work could incorporate CAST-21 to further validate and extend the applicability of the LLM4CS framework, bridging the gap between its results on earlier datasets and its potential performance on newer ones.

6 Conclusion

Our reproducibility study of the LLM4CS framework [24] validated several aspects of the original work while uncovering significant challenges and opportunities for improvement. Despite encountering issues such as outdated resources, non-functional ANCE checkpoints, and gaps in documentation, we were able to reproduce key results using alternative tools like Pyserini. This underscores the importance of widely supported and standardized toolkits in addressing reproducibility barriers within information retrieval research.

Our performance analysis highlighted the strengths and limitations of the framework under various configurations. Dense retrieval methods, while computationally expensive, remain the superior choice for multi-turn conversational search tasks due to their ability to capture semantic nuances and context. However, sparse

retrieval demonstrated its value as a more interpretable and computationally efficient alternative, particularly in resource-constrained scenarios, albeit at the cost of reduced performance. The observed trade-offs emphasize the need for diverse retrieval strategies tailored to specific resource and performance requirements.

Experiments with the open-source Llama3.1-8B-Instruct model revealed its limitations in handling complex IR tasks, particularly when compared to proprietary models like GPT-3.5. While Llama offers a cost-effective and accessible alternative, its smaller size and reduced capacity for contextual understanding significantly impacted performance, especially in scenarios requiring chain-of-thought reasoning or robust query rewriting. These findings highlight the challenges faced by budget-constrained setups and emphasize the importance of further optimizing smaller models for IR tasks.

Our study also shed light on the impact of dataset size and configuration on retrieval performance. The use of a smaller subset in dense retrieval experiments likely contributed to differences from the original paper, including the inability to replicate distinctions between RTR and RAR prompting strategies. This highlights the sensitivity of conversational search frameworks to dataset characteristics and suggests that future work should explore testing across multiple random subcollections to enhance robustness and generalizability.

Ultimately, our work demonstrates the importance of reproducibility as a cornerstone of IR research. By identifying and addressing the limitations of existing frameworks, adopting scalable computational strategies, and extending methodologies with sparse encoders and open-source LLMs, researchers can advance the field toward more robust and accessible solutions.

7 Future Work

Future research should aim to address several challenges and limitations identified in this study while expanding on promising directions suggested by our findings.

A key area for exploration is the application of chain-of-thought (CoT) prompting across larger datasets and alternative retrieval setups. CoT prompting relies on sufficient data to establish nuanced reasoning chains, and our findings suggest its performance may be sensitive to dataset size and context availability. Systematic evaluations are needed to better understand the conditions under which CoT enhances dense retrieval tasks. Building on works such as Wei et al. [34], future studies could refine CoT strategies by integrating retrieval-specific optimizations or hybrid models that combine dense and sparse techniques for conversational search.

Another promising direction is the refinement of aggregation methods. Our experiments demonstrated the utility of mean aggregation but also highlighted its limitations when used in conjunction with reduced datasets. Inspired by advancements in weighted query modeling, such as Term Weighting BERT (TW-BERT) [33], future work could assign greater weight to the initial rewrite, emphasizing the most contextually relevant components. By replicating the first rewrite multiple times before concatenating it with subsequent rewrites, systems could better capture critical intent signals. Rigorous testing across diverse query types and datasets would be necessary to validate these strategies.

The optimization of smaller, open-source LLMs, such as Llama3.1-8B-Instruct, also presents a significant research opportunity. While these models offer computational efficiency and accessibility, our findings revealed their limitations in handling complex multi-turn IR tasks. Future work should focus on task-specific pretraining, instruction tuning, and fine-tuning techniques tailored for conversational search. Leveraging approaches like LoRA (Low-Rank Adaptation of LLMs) [15], which efficiently adapts large models to specific tasks, could enhance the utility of smaller LLMs in resource-constrained environments.

Reproducibility remains an integral part of information retrieval research. Developing modular, scalable frameworks with well documented pipelines and prebuilt indexes could significantly reduce barriers for reproducing experimental results. Toolkits like Pyserini [21] and Hugging Face's ecosystem provide a strong foundation but could benefit from integration with tools that offer better support for large-scale dense retrieval tasks, such as FAISS [16]. Future frameworks should include more flexible options for handling large datasets and detailed instructions to mitigate reproducibility challenges, as seen in frameworks like PCIR [28].

Cross-dataset generalization is another critical area for investigation. While our study focused on CAST-19 and CAST-20, future work should explore the adaptability of the LLM4CS framework to domain-specific corpora, such as biomedical [12], legal [43], or financial datasets [38]. Understanding how conversational search frameworks perform under varying retrieval contexts and dataset characteristics will be key to extending their applicability.

Finally, incorporating explainability mechanisms into conversational search frameworks could foster transparency and user trust. Generating human-readable explanations for search intent representations and retrieval results, inspired by work on explainable IR systems [42], could enhance interpretability and usability. Methods such as generating natural language justifications for retrieval decisions or visualizing the semantic relationships captured by dense and sparse embeddings could help bridge the gap between model performance and user understanding.

By addressing these areas, future work can advance the robustness, generalizability, and accessibility of conversational search frameworks, paving the way for their broader adoption in real-world applications. A deeper understanding of model behavior, combined with scalable and interpretable retrieval systems, will be essential for fostering innovation and ensuring the reliability of conversational search tools.

Acknowledgments

Special appreciation and gratitude goes to our TA Simon Lupart, who consistently provided guidance, practical support, and innovative workarounds to help us tackle the challenges encountered throughout this project. His availability and responsiveness ensured that we had timely advice whenever needed, demonstrating his active engagement and dedication to our success. Simon's eagerness to assist and his commitment to our progress played a crucial role in enabling us to complete this project successfully. We sincerely thank him for his support and mentorship.

References

[1] AIMeta. 2024. The Llama 3 Herd of Models. arXiv:2407.21783 [cs.AI] https://arxiv.org/abs/2407.21783

[2] 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. 2018. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. arXiv:1611.09268 [cs.CL] https://arxiv.org/abs/1611.09268

[3] Zhuyun Dai and Jamie Callan. 2019. Context-Aware Sentence/Passage Term Importance Estimation For First Stage Retrieval. arXiv:1910.10687 [cs.IR] https://arxiv.org/abs/1910.10687

[4] Zhuyun Dai and Jamie Callan. 2019. Deeper Text Understanding for IR with Contextual Neural Language Modeling. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19)*. ACM. https://doi.org/10.1145/3331184.3331303

[5] Jeffrey Dalton, Chenyan Xiong, Ian Soboroff, Andrew Trotman, and Nick Craswell. 2021. TREC 2021 Conversational Assistance Track (CAsT-21). In *Proceedings of the 30th Text REtrieval Conference (TREC 2021)*. National Institute of Standards and Technology (NIST), Gaithersburg, Maryland, USA. https://trec.nist.gov/pubs/trec30/trec2021.html Accessed on [Insert Date].

[6] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE v2: Sparse Lexical and Expansion Model for Information Retrieval. arXiv:2109.10086 [cs.IR] https://arxiv.org/abs/2109.10086

[7] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 2288–2292. https://doi.org/10.1145/3404835.3463098

[8] Apache Software Foundation. [n. d.]. Apache Lucene - A high-performance, full-featured text search engine library. https://lucene.apache.org/.

[9] Jianfeng Gao, Chenyan Xiong, and Paul Bennett. 2020. Recent Advances in Conversational Information Retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 2421–2424. https://doi.org/10.1145/3397271.3401418

[10] Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2022. Precise Zero-Shot Dense Retrieval without Relevance Labels. arXiv:2212.10496 [cs.IR] https://arxiv.org/abs/2212.10496

[11] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, and et al. 2024. The Llama 3 Herd of Models. arXiv:2407.21783 [cs.AI] https://arxiv.org/abs/2407.21783

[12] Yu Gu, Robert Tinn, Hao Cheng, et al. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computational Biology and Bioinformatics (TCBB)* 18, 3 (2021), 620–629. https://doi.org/10.1109/TCBB.2021.3053059

[13] Odd Erik Gundersen and Sigbjørn Kjensmo. 2018. State of the Art: Reproducibility in Artificial Intelligence. *Proceedings of the AAAI Conference on Artificial Intelligence* 32, 1 (Apr. 2018). https://doi.org/10.1609/aaai.v32i1.11503

[14] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. 2018. Deep reinforcement learning that matters. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence* (New Orleans, Louisiana, USA) *(AAAI'18/IAAI'18/EAAI'18)*. AAAI Press, Article 392, 8 pages.

[15] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint arXiv:2106.09685* (2021).

[16] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. In *IEEE Transactions on Big Data*, Vol. 7. 535–547. https://doi.org/10.1109/TBDATA.2019.2921572

[17] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-Scale Similarity Search with GPUs. *IEEE Transactions on Big Data* 7, 3 (2021), 535–547. https://doi.org/10.1109/TBDATA.2019.2921572

[18] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, Vol. 1. Minneapolis, Minnesota, 2.

[19] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics* 7 (2019), 452–466. https://doi.org/10.1162/tacl_a_00276

[20] Canjia Li, Andrew Yates, Sean MacAvaney, Ben He, and Yingfei Sun. 2021. PARADE: Passage Representation Aggregation for Document Reranking. arXiv:2008.09093 [cs.IR] https://arxiv.org/abs/2008.09093

[21] Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python Toolkit for Reproducible Information Retrieval Research with Sparse and Dense Representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*. 2356–2362.

[22] Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Ré, and Beidi Chen. 2023. Deja Vu: contextual sparsity for efficient LLMs at inference time. In *Proceedings of the 40th International Conference on Machine Learning* (Honolulu, Hawaii, USA) *(ICML'23)*. JMLR.org, Article 919, 40 pages.

[23] Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. CEDR: Contextualized Embeddings for Document Ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19)*. ACM. https://doi.org/10.1145/3331184.3331317

[24] Kelong Mao, Zhicheng Dou, Fengran Mo, Jiewen Hou, Haonan Chen, and Hongjin Qian. 2023. Large Language Models Know Your Contextual Search Intent: A Prompting Framework for Conversational Search. arXiv:2303.06573 [cs.IR] https://arxiv.org/abs/2303.06573

[25] Kelong Mao, Zhicheng Dou, Hongjin Qian, Fengran Mo, Xiaohua Cheng, and Zhao Cao. 2022. ConvTrans: Transforming Web Search Sessions for Conversational Dense Retrieval. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (Eds.). Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2935–2946. https://doi.org/10.18653/v1/2022.emnlp-main.190

[26] Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021. Generation-Augmented Retrieval for Open-domain Question Answering. arXiv:2009.08553 [cs.CL] https://arxiv.org/abs/2009.08553

[27] Bhaskar Mitra and Nick Craswell. 2018. An Introduction to Neural Information Retrieval. *Foundations and Trends® in Information Retrieval* 13, 1 (2018), 1–126. https://doi.org/10.1561/1500000061

[28] Fengran Mo, Longxiang Zhao, Kaiyu Huang, Yue Dong, Degen Huang, and Jian-Yun Nie. 2024. How to Leverage Personal Textual Knowledge for Personalized Conversational Information Retrieval. arXiv:2407.16192 [cs.IR] https://arxiv.org/abs/2407.16192

[29] Rodrigo Nogueira and Kyunghyun Cho. 2020. Passage Re-ranking with BERT. arXiv:1901.04085 [cs.IR] https://arxiv.org/abs/1901.04085

[30] OpenAI. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[31] Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Lariviere, Alina Beygelzimer, Florence d'Alche Buc, Emily Fox, and Hugo Larochelle. 2021. Improving Reproducibility in Machine Learning Research(A Report from the NeurIPS 2019 Reproducibility Program). *Journal of Machine Learning Research* 22, 164 (2021), 1–20. http://jmlr.org/papers/v22/20-303.html

[32] Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (April 2009), 333–389. https://doi.org/10.1561/1500000019

[33] Karan Samel, Cheng Li, Weize Kong, Tao Chen, Mingyang Zhang, Shaleen Gupta, Swaraj Khadanga, Wensong Xu, Xingyu Wang, Kashyap Kolipaka, et al. 2023. End-to-End Query Term Weighting. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4778–4786.

[34] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv preprint arXiv:2201.11903* (2022).

[35] Chenyan Xiong, Zhenghao Liu, Si Sun, Zhuyun Dai, Kaitao Zhang, Shi Yu, Zhiyuan Liu, Hoifung Poon, Jianfeng Gao, and Paul Bennett. 2020. CMT in TREC-COVID Round 2: Mitigating the Generalization Gaps from Web to Special Domain Search. arXiv:2011.01580 [cs.IR] https://arxiv.org/abs/2011.01580

[36] Lee Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russel Power. 2020. ANCE: Approximate Nearest Neighbor Negative Contrastive Estimation Checkpoints. https://github.com/microsoft/ANCE. https://github.com/microsoft/ANCE Checkpoints for the ANCE model, a dense retriever for information retrieval tasks..

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

[38] Wanxiang Yang, Wei Sun, et al. 2020. FinBERT: A Pretrained Language Model for Financial Communications. *arXiv preprint arXiv:2006.08097* (2020).

[39] 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] https://arxiv.org/abs/2006.05009

[40] Shi Yu, Zhenghao Liu, Chenyan Xiong, Tao Feng, and Zhiyuan Liu. 2021. Few-Shot Conversational Dense Retrieval. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 829–838. https://doi.org/10.1145/3404835.3462856

[41] Shi Yu, Zhenghao Liu, Chenyan Xiong, Tao Feng, and Zhiyuan Liu. 2021. Few-Shot Conversational Dense Retrieval. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR*

'21). ACM, 829–838. https://doi.org/10.1145/3404835.3462856

[42] Zhao Zhang et al. 2020. Explainable Information Retrieval: A Survey. *arXiv preprint arXiv:2011.04467* (2020).

[43] Lewis Zheng, Neeraj Guha, et al. 2021. LegalBERT: The Muppets straight out of law school. *Proceedings of the 18th International Conference on Artificial Intelligence and Law* (2021), 121–130. https://doi.org/10.1145/3462757.3466097

[44] Yujia Zhou, Jing Yao, Zhicheng Dou, Ledell Wu, and Ji-Rong Wen. 2022. DynamicRetriever: A Pre-training Model-based IR System with Neither Sparse nor Dense Index. arXiv:2203.00537 [cs.IR] https://arxiv.org/abs/2203.00537