Generating Multi-Aspect Queries for Conversational Search

Zahra Abbasiantaeb **Simon Lupart** **Mohammad Aliannejadi**
University of Amsterdam
Amsterdam, The Netherlands
{z.abbasiantaeb,s.c.lupart,m.aliannejadi}@uva.nl

Abstract

Conversational information seeking (CIS) systems aim to model the user's information need within the conversational context and retrieve the relevant information. One major approach to modeling the conversational context aims to rewrite the user utterance in the conversation to represent the information need independently. In this work, we hypothesize that breaking down the information of an utterance into multiple queries covering different aspects of the information need can lead to more effective retrieval performance. This is more evident in more complex utterances that require gathering evidence from various information sources, where a single query rewrite or query representation cannot capture the complexity of the utterance. We propose MQ4CS, a multi-aspect query generation and retrieval framework, which uses large language models (LLMs) to break the user utterance into multiple queries. This approach improves retrieval performance, as most utterances benefit from more than one rewritten query. We evaluate MQ4CS on six widely used CIS datasets, showing it outperforms state-of-the-art query rewriting methods. Using MQ4CS, we also construct MASQ dataset, which includes multiple-aspect queries for the six datasets. Fine-tuning the Llama model on MASQ yields significant improvements. We make our code and dataset publicly available at `https://github.com/ZahraAbbasiantaeb/MQ4CS-MASQ.git`.

1 Introduction

Conversational information seeking (CIS) is a well-established topic in information retrieval (IR) (Zamani et al., 2023; Aliannejadi et al., 2024b), where a knowledge assistant interacts with the user to fulfill their information needs. While conversations can be complex (Radlinski and Craswell, 2017), involving various types of interactions such as revealment and clarification, one of the main goals of the system is to provide an answer to the user's

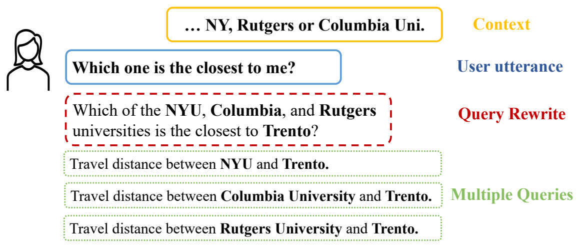

... NY, Rutgers or Columbia Uni.
Context
Which one is the closest to me?
User utterance
Which of the NYU, Columbia, and Rutgers universities is the closest to Trento?
Query Rewrite
Travel distance between NYU and Trento.
Travel distance between Columbia University and Trento.
Travel distance between Rutgers University and Trento.
Multiple Queries

Figure 1: An example conversation with a complex user utterance. The system needs to generate three distinct queries with different aspects and search for every query.

request. As a conversation is prolonged, several challenges and complexities arise, such as language dependency (e.g., anaphora, ellipsis), long conversation context modeling, and more complex information needs (Dalton et al., 2020; Owoicho et al., 2022; Aliannejadi et al., 2024a). Much research aims to address these issues by utterance rewriting, where the written query is aimed to resolve language dependencies and encapsulate the context and information complexities (Voskarides et al., 2020; Yu et al., 2020b).

Encapsulating the complex nature of the conversational information need in one single rewritten utterance can lead to several limitations, especially in cases where the query cannot be answered using a single passage and requires complex reasoning over multiple facts from different sources in a chain-of-thought scenario (Aliannejadi et al., 2024a; Lyu et al., 2023). Take the user query of Figure 1 as an example. It is unlikely to find a passage that has distance information about all these universities compared to the user's address. Therefore, the system would need to gather relevant information from different sources (e.g., distance of each city) and reason over the gathered evidence to generate the final response. Existing ranking methods rely solely on semantic similarity between a query and a passage, without high-level reasoning or control over the set of retrieved passages. For example, they do not ensure that the top results contain address information about all three universities the user is interested in. Therefore, there is no guarantee that the passages containing different pieces of relevant information would be ranked high.

Recent research suggests that retrieving passages based on multiple queries can lead to improved retrieval performance (Mao et al., 2023a; Kostric and Balog, 2024). Existing methods like LLM4CS, however, focus on prompting the LLM to generate multiple query rewrites. This would lead to generating different versions of the same query rewrite, taking advantage of the language diversity, rather than capturing different aspects of the user's information need in the different queries. To the best of our knowledge, no research has systematically studied and analyzed the impact of breaking information need into multiple distinct queries with different aspects. Throughout this paper, we use the term multi-aspect to refer to multiple distinct queries that capture different aspects of the information need. We hypothesize that summarizing the user's information need into a single query rewrite is not effective for retrieval across many conversational turns. As the conversations progress, the user's information needs evolve and grow more complex, making a single query rewrite insufficient for retrieval. To test our hypothesis, in a *single* LLM call, we prompt the LLM to generate 1–5 queries with different aspects for each utterance for six major conversational search (CS) datasets. Our goal is to break the information need of the user into several queries with each query covering one aspect of the information need. Then, we use the generated queries in the same retrieval pipeline used for a single query rewrite. Assuming that the system knows the optimum number of generated queries for each user utterance based on an Oracle setting, we observe that at least more than 50% of the utterances (see Figure 3) from TREC Interactive Knowledge Assistance Track (iKAT) and Conversational Assistance Track (CAsT) datasets exhibit better performance using the multi-aspect generated queries. This verifies our hypothesis, showing that the majority of the utterances benefit from multi-aspect query generation. Based on this, we build a new dataset, called MASQ, consisting of multi-aspect queries for each user utterance, together with the optimum number of queries to represent the user utterance.

Inspired by our findings, we propose a simple yet effective conversational retrieval framework based on generating multi-aspect queries, called MQ4CS (**M**ulti-aspect **Q**uery Generation and Retrieval **for** **C**onversational **S**earch). MQ4CS takes a conversational utterance as input and generates the maximum of given number of queries to address the information need from multiple aspects. It then retrieves passages for each query and in the final step does rank list fusion to output a single ranking. MQ4CS relies on the LLM internal knowledge to model the user's information need and generate multi-aspect queries. We implement the MQ4CS in both zero-shot and fine-tuning settings, using our MASQ dataset for fine-tuning.

We conduct experiments on six widely used CS datasets, namely, Text REtrieval Conference (TREC) CAsT 19, 20, & 22, TREC iKAT 23 & 24, and TopiOCQA (Adlakha et al., 2022). We show that MQ4CS outperforms the SoTA query rewriting approaches on all the datasets in terms of various metrics. Furthermore, we study the effect of multi-aspect query generation in various experimental setups, such as different levels of complexity and topic switches. We observe a higher performance gap between MQ4CS and SoTA models as the complexity of user utterance increases, demonstrating the effectiveness of multi-aspect query generation for complex user queries. We summarize our contributions as follows:

- We propose a conversational passage retrieval framework by leveraging the LLM's internal knowledge to rewrite user utterances to multi-aspect queries and fuse their rankings.
- We show generating multi-aspect queries improves retrieval for the majority of queries. To facilitate research in this area and establish it as a new task, we build and release a multi-aspect query dataset, called MASQ, focusing on six major CS datasets.
- We fine-tune the Llama model on our MASQ dataset. We conduct extensive experiments, showcasing the effectiveness of both fine-tuned and zero-shot MQ4CS for conversational passage retrieval on six major CS datasets using commercial and open-source LLMs.

2 Related Work

Recently, CIS has gained significant popularity in both IR and natural language processing (NLP) communities (Anand et al., 2020). Similar to knowledge-intensive dialogues (Dinan et al., 2019; Feng et al., 2021; Li et al., 2022), a key challenge in CIS is to model the dialogue context to better understand the user information need and perform effective retrieval (Zamani et al., 2023). TREC CAsT 19–22 (Dalton et al., 2020) and iKAT 23 (Aliannejadi et al., 2024a) aim to address these challenges through a common evaluation framework in which complex and knowledge-intensive dialogues were provided to participants, as well as several passage collections. The goal was to retrieve relevant passages for each turn in a dialogue and generate a response synthesizing several passages. TREC CAsT 22 (Owoicho et al., 2022) advanced this track by introducing mixed-initiative (clarifying questions (Rao and III, 2018; Aliannejadi et al., 2019)) and user feedback (Owoicho et al., 2023) turns. TREC iKAT 23 focuses on the long-term personal conversational memory of the model via introducing a personal knowledge graph.

A line of research aims at learning to represent the dialogue context directly for passage retrieval (Yu et al., 2020b; Hai et al., 2023), where a distillation loss learns to map the representation of the whole dialogue context to the gold resolved query, hence improving the dense retrieval performance. The INSTRUCTOR (Jin et al., 2023) model trains the document encoder model by using the relevance score predicted by an LLM.

Most existing methods tackle the context modeling problem by query rewriting where the goal is to address the ambiguity and dependence of a user utterance by resolving its dependencies and making it self-contained (Voskarides et al., 2019, 2020; Lin et al., 2021c). The rewritten query is supposed to be a self-contained and context-independent query that represents the user's information need per turn. CRDR (Qian and Dou, 2022) forms the rewritten query with modifying the query by disambiguation of the anaphora and ellipsis. The existing work trains GPT2 (Yu et al., 2020a; Vakulenko et al., 2021) and T5 (Dalton et al., 2020) models on the CANARD dataset (Elgohary et al., 2019) to generate the rewritten query. LeCoRE (Mao et al., 2023b) and DiSCo (Lupart et al., 2025) are extensions of the SPLADE model (Formal et al., 2021) obtained by denoising the representation of the context. The denoising model works by distilling knowledge from query rewrite. ConvGQR (Mo et al., 2023) model expands the query rewrite with potential answers. They train two separate models for the query rewrite and answer generation. CONQRR (Wu et al., 2022) trains the T5 model using reinforcement learning to generate query rewrite based on the retrieval performance and achieves a better performance compared to the T5QR (Raffel et al., 2020) model. Ye et al. (2023) propose using LLMs as zero- and few-shot learners in two steps including query rewriting and rewrite editing to form the query rewrite. LLM4CS (Mao et al., 2023a) employs different prompting strategies and creates multiple query rewrites and answers. The embedding of query rewrites and answers are combined using various methods and the aggregated representation is used for retrieval. LLM4CS is the most similar work to ours. Our work distinguishes itself from LLM4CS in various aspects: (i) LLM4CS does not prompt the LLM to generate multi-aspect queries. Instead, it prompts for a query rewrite and repeats this prompt five times to get different generation variations, with no guarantee that the generated queries address different aspects of the original query. MQ4CS, instead prompts the LLM to break the user information need into multiple queries by generating various multi-aspect queries, focusing on different perspectives. (ii) LLM4CS either selects one of the five generated queries or computes the average of representations of multiple queries for retrieval. Therefore, the retrieval task is only done based on one query/representation. MQ4CS, on the other hand, aims not to miss any documents that can be retrieved by each single query. Therefore, it does passage retrieval for all five queries independently and fuses their rankings.

3 Methodology

3.1 Task Definition

Each conversation has several turns, where a turn starts with a user utterance u_i, followed by a system response r_i. The context of the conversation at turn i is shown as $c_i = \{(u_1, r_1), ..., (u_{i-1}, r_{i-1})\}$. Different from other datasets, the TREC iKAT 23 & 24 datasets also include the persona of the user. The persona is a knowledge base, consisting of a set of statements shown as $PTKB = \{s_1, ..., s_l\}$. Each statement is a natural language sentence explaining the user preferences, interests, and personal information (e.g. "I am vegan", "I plan to move to China", "I am a photographer", "I am allergic to lactose"). The task of passage retrieval for conversational assistants is to retrieve relevant passages to the current user utterance (u_i) from the collection D where $D = \{d_1, ..., d_{|D|}\}$. The ordered list of retrieved passages for user utterance u_i is shown as D'_i which is a subset of D.

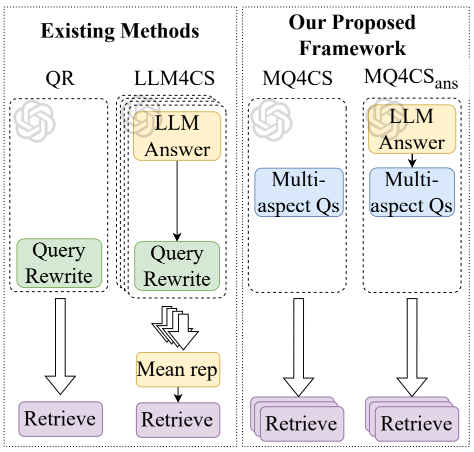


Figure 2: A high-level overview of the proposed framework, compared with existing models. In QR a single query is generated by LLM and in LLM4CS, multiple LLM calls are made to generate different query rewrites. In MQ4CS and MQ4CS$_{\text{ans}}$, we generate multi-aspect queries in a single prompt. We perform retrieval on each query independently to avoid information loss.

3.2 MQ4CS Framework

In this section, we introduce our proposed CS framework, called MQ4CS (**M**ulti-aspect **Q**uery Generation and Retrieval **for** **C**onversational **S**earch). In Figure 2, we show the overview of the existing query rewriting techniques (on the left), compared to our proposed framework (on the right). In general, a CS framework consists of a query rewriting module that aims to resolve the current utterance's dependencies and generate a stand-alone query. The newly generated query is used for retrieval and reranking.

Methods like LLM4CS (Mao et al., 2023a) prompt the LLM multiple times to generate the query rewrite, and then take the average representation of them to do the retrieval. We take a different approach in the query rewriting phase, that is, we prompt the LLM only once to resolve and generate multi-aspect queries (a maximum of ϕ) that represent the information need for the current utterance from different aspects (see the example in Figure 1). We leverage the internal knowledge and reasoning capabilities of LLMs to understand complex information need and break it into multiple queries. We then pass each query to retrieval and reranking and fuse the final ranking list of each query to output one single ranking for the utterance. We describe our framework's components below.

LLM answer. Inspired by existing work that shows asking LLMs for explanation further improves their performance (Wei et al., 2022), as well as the work that shows that CS can be improved by asking the LLM to generate an answer (Mao et al., 2023a), we ask the LLM to first give a response to the user's utterance. Therefore, the LLM response generator module instructs the LLM to generate the response r'_i to the user utterance given the conversation context. The generated response in this step could be used by the query generator as shown in Equation 2. Our response generation module (called AG) takes u_i, c_i, and PTKB (if exists) as input and generates r'_i, as below:

$$r'_i = AG(u_i, c_i, \text{PTKB}) \ . \tag{1}$$

We use the first part of the prompt in Table 4 in Appendix A, for answer generation (i.e. AG function).

Multi-aspect query generation. 5This module takes the conversation context, user persona (if exists), and current utterance u_i as input, and prompts an LLM to generate a maximum of ϕ queries denoted as $Q^i = \{q^i_1, ..., q^i_\phi\}$ to retrieve passages for user utterance u_i. We designed two different prompts for this module for generating multiple queries in one single prompt. The first prompt includes the LLM response r'_i as input (second part of Table 4 in Appendix A). The other prompt directly instructs the LLM to generate queries without giving a response (Table 5 Appendix A). Therefore, our query generator module (QG) is as follows:

$$Q^i = QG(r'_i, u_i, c_i, \text{PTKB}, \phi) \ , \tag{2}$$

where ϕ is the number of queries to generate. We parse the output Q^i to get the list of generated queries denoted as $\{q^i_1, ..., q^i_\phi\}$.

Retrieval and reranking. Most existing methods follow a two-step approach for retrieval, i.e., first-stage retrieval and reranking (Lin et al., 2021b). We use two different setups (1) only first stage dense retrieval and (2) first stage sparse retrieval followed by reranking. For dense retrieval, we rely on dense embedding vectors from ANCE (Xiong et al., 2021) to encode rewrites, and then retrieve the closest documents from the collection for each rewrite. The ranked lists are then aggregated according to the strategy. Concerning sparse retrieval, we use BM25 from Pyserini (Lin et al., 2021a) for first-stage retrieval (Ret) and the pre-trained Cross-Encoder model `ms-marco-MiniLM-L-6-v2` from the `sentence_transformers` for reranking ($ReRank$). This module takes the query q^i_k as input and generates ranked list of documents ($D'_{k,i}$) as output for each query (q^i_k) as follows:

$$D'_{k,i} = ReRank(Ret(D, q^i_k), q^i_k) \ . \tag{3}$$

Note that for dense retrieval we eliminate the $ReRank$ function from above equation.

Ranked list fusion. This module takes the document ranking of each generated query as input (i.e., ϕ ranked lists), and produces one final ranking. We get inspiration from the existing studies on data fusion (Hsu and Taksa, 2005; Farah and Vanderpooten, 2007) and propose min–max normalization followed by round-robin rank list fusion. First, we apply min-max normalization on each list to normalize the relevance scores in the range of 0–1. Then, we select the documents with the same rank in each list and put them in the final ranking list based on their scores. We start from rank 1 and repeat the process with the second passage and so on, while removing the duplicates. The intuition behind repeating this process per each rank is to give an equal attention to the documents retrieved from various sources. We define the $Fuse$ function which aggregate the ϕ ranked lists $(D'_{1,i}, ..., D'_{\phi,i})$ into one list as follows.

$$D'_i = Fuse(D'_{1,i}, ..., D'_{\phi,i}) \ . \qquad (4)$$

Model variants. We propose three variants of our framework. They all follow the same paradigm, with slight variations.

- **MQ4CS**: This is our main model that only leverages the query generation (without LLM's answer) and performs ranked list fusion. This model utilizes LLMs as zero-shot learners. The prompt used for this model is shown in Table 5 in Appendix A.
- **MQ4CS** from **Ans**wer (**MQ4CS$_{\text{ans}}$**): This model leverages the LLM response to generate queries. Therefore it first generates r'_i and then passes it to QG module (Equation 2). This model also performs ranked list fusion. The prompt used for this model is shown in Table 4 in in Appendix A. Similar to the MQ4CS model, we leverage LLMs as zero-shot learners in this model.
- **MQ4CS** [FT]: This model is another version of the MQ4CS model which uses fine-tuned LLMs for query generation. We use the Oracle dataset (see section 4) to fine-tune the Llama model for the query generation task. Note that the number of queries is not provided as input, and the model learns to generate an appropriate number of queries during fine-tuning on the Oracle dataset.

Base LLM. We use GPT4 and GPT4o as zero-shot learners to implement QG and AG functions. Also, we fine-tune Llama 3.1 for QG function.

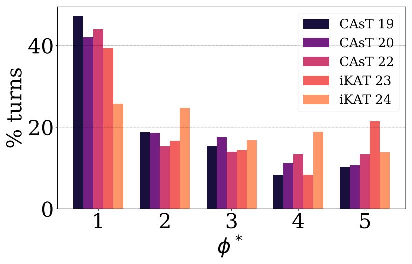


Figure 3: Distribution of the turns with the corresponding ϕ^*. The value of ϕ^* is selected based on nDCG@3.

4 Preliminary Experiments & Multi-Aspect Query Collection

In this section, we report the results of our preliminary experiments and describe how we collect our novel multi-aspect query datasets, called MASQ, that we use for LLM fine-tuning.

Motivation for Oracle. As argued earlier, despite the long-standing research on search result diversification (Santos et al., 2015), retrieval and reranking models, at their current design fail to plan and strategize their ranking. In other words, if we take the example of Figure 1, no retrieval system would ensure that the top-K results have distance information of the three cities mentioned in the query. Therefore, we hypothesize that we could leverage LLMs to break down complex queries into multi-aspect queries, using each we can retrieve documents that address those aspects independently. To test our hypothesis, we conduct a preliminary experiment based on Oracle query generation model.

In our preliminary experiments, we realized that LLMs are not effective in determining the best ϕ value and always tend to generate the maximum number of queries instructed in the prompt. While, each user utterance needs a different number of queries depending on the complexity of the information need and the retrieval model. Consequently, while MQ4CS can generate multi-aspect queries in a zero-shot setting, it still needs to have the best ϕ for each query as input. Therefore, we determine the best ϕ using the performance of the retrieval model with different values of ϕ for every single query. Formally, we define ϕ^*_i as follows:

$$\phi^*_i = \underset{\phi \in \{1,...,5\}}{\arg\max}\ eval(MQ4CS(u_i, c_i, PTKB, \phi))$$

where ϕ^*_i denotes the best value of ϕ for turn i and MQ4CS represents our end-to-end framework which given the user utterance, context, and per-

Table 1: Performance of the Oracle experiments compared to Human-resolved utterances (HumanQR) using Sparse retrieval (BM25) with re-ranking.

Method	**CAsT 20**			**CAsT 22**			**CAsT 19**		
	nDCG@3	**R@100**	**MRR**	**nDCG@3**	**R@100**	**MRR**	**nDCG@3**	**R@100**	**MRR**
HumanQR	50.5	61.8	66.8	41.3	37.6	60.4	-	-	-
Oracle MQ4CS$_{\text{ans}}$	63.6	71.9	79.3	55.6	43.7	79.9	68.7	68.9	88.5
Oracle MQ4CS	66.0	67.8	83.7	53.7	44.9	75.7	67.8	70.8	84.4
Method	**iKAT 23**			**iKAT 24**			**TopiOCQA**		
	nDCG@3	**R@100**	**MRR**	**nDCG@3**	**R@100**	**MRR**	**MAP**	**R@100**	**MRR**
HumanQR	30.7	35.8	43.3	43.7	32.7	62.3	-	-	-
Oracle MQ4CS$_{\text{ans}}$	40.6	37.6	57.5	61.2	43.4	83.9	58.6	91.0	58.6
Oracle MQ4CS	37.7	36.0	56.4	59.6	43.5	81.3	59.4	91.9	59.4

sona, returns the ranked list of documents. The $eval$ shows the evaluation function which evaluates the retrieval performance. The performance of the Oracle model is considered as the upper bound because in this setting, for each user utterance, we issue a different prompt with the given ϕ.

Oracle Experiments. We report the results of our Oracle experiment in Table 1. Compared to the human (HumanQR baseline), we observe a massive improvement in the retrieval (up to 45%) in terms of all metrics, for all the datasets. These results show the necessity of using multiple queries with different aspects rather than using a single comprehensive query rewrite for improving the retrieval. To further understand the effect of multi-aspect query generation on retrieval, we compare the ϕ^* values of all the conversational turns and plot the results in Figure 3. In this figure, we can see what percentage of the utterances are too complex to be addressed with one query rewrite. Note that we exclude turns with $\phi^* = 1$ while the performance of the model is 0.0 for them. According to Figure 3, more than 55% of iKAT and CAsT 22 datasets' turns need more than one query to achieve higher nDCG@3.

Moreover, the average of ϕ^* of iKAT 23, CAsT 22, CAsT 20, and TopiOCQA are as follows: 2.37, 2.12, 1.96, and 1.29. This finding indicates that the TREC datasets include more complex user utterances compared to the TopiOCQA dataset. Although TopiOCQA features various topic shifts, the TREC datasets present more complexity as they also have various topic shifts in each dialog, as well as complex information needs (Dalton et al., 2020). This demonstrates that retrieval and ranking models do not learn to diversify the ranked results to generate a complete answer in a RAG setting.

Multi-Aspect Query Collection (MASQ). Our Oracle experiments prove that there is a large room for improvement both in the retrieval and query rewriting phases. Therefore, we use the data and labels we collect in this stage as training data to learn the task of multi-aspect query generation. We call our novel dataset MASQ. MASQ includes (i) the queries generated using values of 1–5 for ϕ using both MQ4CS and MQ4CS$_{\text{ans}}$ frameworks; (ii) the value of ϕ^* for each user utterance, as well as the retrieval performance for each value of ϕ; (iii) over the test set of TREC CAsT 19, 20, & 22, TREC iKAT 23 & 24, and TopiOCQA datasets. MASQ can be used to learn the optimum number of queries to be generated (i.e., ϕ^*), as well as to train smaller models for multi-aspect query generation. The statistics of MASQ is shown in Table 7 in Appendix B.

5 Results and Discussions

The detailed experimental setup, including descriptions of baselines, datasets, evaluation metrics, and hyperparameters, is provided in Appendix C.

Performance comparison. We report the performance of our proposed models based on zero-shot LLM (using a fixed value of $\phi = 3$), fine-tuning LLM, and the baselines in Table 2. We report the results using dense retrieval in Tables 9 and 10 in Appendix D. We observe that our zero-shot MQ4CS and its variants based on sparse retrieval + reranking outperform the SOTA model (i.e., LLM4CS) by a large margin using both dense retrieval and sparse retrieval + reranking. This observation demonstrates the effectiveness of multi-aspect query generation. Note that our zero-shot model is sub-optimal as uses a constant value of $\phi = 3$ for all turns and we know the LLMs cannot always determine the efficient number of queries for retrieval. The better performance of zero-shot MQ4CS compared to the LLM4CS baseline indicates (i) the importance of our proposed retrieval

Table 2: Performance of sparse retrieval (BM25) with re-ranking and dense retrieval. The best results are shown **bold** and the best result outperforming human is shown with underline. We use $\phi = 3$ for MQ4CS and MQ4CS$_{\text{ans}}$ models. Results that are significantly better than the GPT4QR baseline after Bonferroni correction ($p < 0.00417$ and $p < 0.00555$ for TREC and TopiOCQA datasets, respectively) are marked with †.

Method	LLM	CAsT 20			CAsT 22			CAsT 19		
		nDCG@3	R@100	MRR	nDCG@3	R@100	MRR	nDCG@3	R@100	MRR
T5QR	-	38.7	45.6	53.1	30.2	23.9	45.5	**56.5**	59.8	74.6
ConvGQR	-	35.7	47.7	49.4	25.0	22.1	41.0	50.2	53.6	67.5
LLM4CS	GPT4	38.7	51.1	54.5	27.5	30.0	45.3	52.2	55.8	72.7
LlamaQR	Llama 3.1	36.3	49.3	51.4	30.8	27.3	49.0	-	-	-
GPT4QR	GPT4	**46.8**	55.2	61.5	34.8	30.1	52.2	**56.5**	62.0	73.8
HumanQR		50.5	61.8	66.8	41.3	37.6	60.4	-	-	-
MQ4CS$_{\text{ans}}$	GPT4	43.6	59.5	**75.3**	**36.1**	31.3	**69.2**	45.3	59.9	71.4
MQ4CS	GPT4	44.8	**64.8**	64.3	35.9	**36.3**	59.8	52.6	**64.1**	**75.8**
MQ4CS [FT]	Llama 3.1	42.6	62.0	61.9	34.1	32.3	55.5	52.1	59.5	70.9

Method	LLM	iKAT 23			iKAT 24			TopiOCQA			
		nDCG@3	R@100	MRR	nDCG@3	R@100	MRR	MAP	R@10	R@100	MRR
T5QR	-	14.1	13.8	23.9	23.5	19.8	34.9	33.5	50.2	62.5	33.5
ConvGQR	-	14.7	13.9	21.9	24.1	19.3	35.6	31.1	49.0	63.2	31.1
LLM4CS	GPT4	10.5	14.5	16.5	14.9	17.8	25.8	36.8	57.1	75.9	36.8
LlamaQR	Llama 3.1	9.6	14.2	16.0	-	-	-	-	-	-	-
GPT4QR	GPT4	21.9	22.1	34.6	45.4	34.3	66.2	45.4	66.9	80.9	45.4
HumanQR		30.7	35.8	43.3	43.7	32.7	62.3	-	-	-	-
MQ4CS$_{\text{ans}}$	GPT4	**24.8†**	**29.1†**	**41.2†**	44.8	**35.9**	65.5	46.7	70.6	87.0	46.7
MQ4CS	GPT4	22.0	27.2	39.1	**46.6**	35.0	**69.2**	**47.5†**	**72.6†**	**87.8†**	**47.5†**
MQ4CS [FT]	Llama 3.1	14.6	14.8	24.1	30.0	31.1	52.5	41.4	67.2	79.6	41.4

framework i.e., doing retrieval using each query separately and fusing the retrieved passages, and (ii) the effectiveness of the generated multi-aspect queries to cover more diverse aspects from information need of the user.

It is worth noting that our proposed GPT-based query rewriting baseline, GPT4QR, outperforms the SoTA baseline (LLM4CS). Unlike LLM4CS, which requires multiple LLM calls and employs a more complex method to aggregate the generated queries, GPT4QR achieves superior performance with just a single LLM call. Comparing our zero-shot MQ4CS model with GPT4QR, we observe that using sparse retrieval + reranking, the MQ4CS outperforms GPT4QR over all metrics for iKAT 23 & 24, CAsT 22 and TopiOCQA datasets. Also, it outperforms the CAsT 19 & 20 datasets over R@100 and MRR metrics. Our zero-shot MQ4CS model achieves more improvement over Recall@100 compared to nDCG@3 metric. For example, it improves the Recall@100 compared to the best baseline (i.e., GPT4QR) 9.6%, 6.2%, 5.1%, and 7.8 % on CAsT 20 & 22, iKAT 23, and TopiOCQA datasets, respectively. Hence, the generated queries cover more aspects of the information need and MQ4CS retrieves passages from various sources of information.

Neither MQ4CS nor MQ4CS$_{\text{ans}}$ consistently outperforms the other. This indicates that generating multi-aspect queries using LLM responses is not inherently better than directly prompting the LLM to produce multi-aspect queries. However, MQ4CS is more efficient as it requires only a single LLM call, whereas MQ4CS$_{\text{ans}}$ involves two LLM calls.

According to Table 10 in Appendix D, our proposed zero-shot MQ4CS framework demonstrates significant improvements compared to LLM4CS over iKAT 23 & 24, CAsT 22, and TopiOCQA datasets (using either dense retrieval or sparse retrieval + reranking). These datasets have longer conversations compared to CAsT 19 & 20 datasets (see Table 8 in Appendix B). The iKAT datasets include more complex information needs and persona of the user and the TopiOCQA dataset includes conversations that exhibit topic shifts. The results show that zero-shot MQ4CS framework with a constant value of ϕ is more effective for such complex datasets which require reasoning over longer context and persona of the user. On the other hand, the zero-shot MQ4CS framework does not outperform the existing baselines on CAsT 19. The queries in CAsT 19 are much simpler compared to other CAsT datasets and do not include gold responses. As a result, the model only needs to reason over previous user queries (instead of previous user queries and system responses) to understand the current information need. This suggests that using a fixed value of ϕ is less effective for handling such simpler queries.

Fine-tuned MQ4CS. We fine-tune the Llama model to generate multi-aspect queries using our MASQ dataset (i.e., MQ4CS [FT]). According to

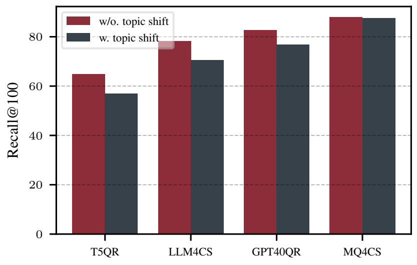


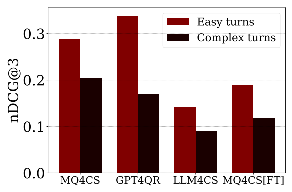
Figure 4: Performance of MQ4CS and baselines on TopiOCQA, over the turns with and without topic shift.

the Table 2, our MQ4CS [FT] model with sparse retrieval + reranking outperforms the LLM4CS model over CAsT 20 & 22, iKAT 23 & 24, and TopiOCQA datasets over all metrics while it uses Llama model with 8B parameters and LLM4CS uses much larger and stronger commercial LLM (i.e., GPT4).

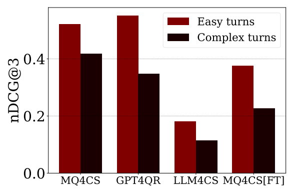
Effect of topic shift. In this experiment, we look at the TopiOCQA dialogues and study the performance of the models on the turns with and without a topic shift, as defined in the original dataset. Topic shift happens when the conversation changes focus from one topic to another. Figure 4 shows the performance in Recall@100 broken by the turn type (i.e., with or without topic shift). Looking at the three baselines (T5QR, LLM4CS, and GPT4QR), we see that they all perform better on the turns without a topic shift. These topics are presumably simpler; therefore, it is not surprising to see the higher performance. Surprisingly, MQ4CS demonstrates a different trend, where the performance of turns of the two groups is almost the same, demonstrating the power of multi-aspect query generation in tackling more complex dialog turns with topic shifts. MQ4CS has a more robust performance over turns with or without topic shift.

Effect of query complexity. We define the utterances with $\phi^* = 1$ as *easy*, and $\phi^* > 1$ as *complex* utterances. We report the performance of our proposed models and baselines over easy and complex turns in Figure 5. As shown in the figure, our proposed zero-shot MQ4CS model (with a constant value of $\phi = 3$) outperforms the baseline models on complex turns. However, it is not as good as the best baseline (i.e., GPT4QR) on easy turns. This indicates that even though selecting a fixed value of ϕ can improve the performance over complex turns, it is not optimal for easy turns. Interestingly, the fine-tuned version of our proposed model (i.e., MQ4CS [FT]), is performing better than LLM4CS over both complex and easy turns on all datasets (except complex turns of CAsT 19). This shows the effectiveness of MASQ dataset where a smaller open-source LLM can learn to generate more effective queries compared to LLM4CS, which leverages commercial LLM. Moreover, looking at the performance of the models on easy turns, we observe that our MQ4CS approach is outperforming the LLM4CS on all datasets except CAsT 19 and is outperforming the GPT4QR baseline on TopiOCQA dataset. This finding indicates that the zero-shot MQ4CS model (with a constant value of $\phi = 3$) is capable of generating more effective queries for simple turns compared to the LLM4CS baseline. Also, the queries generated by our model for easy turns are more effective than the single query generated by GPT4QR baseline for TopiOCQA. However, we know that the single query generated by zero-shot MQ4CS in the Oracle setting (i.e., $\phi^* = 1$), is even more effective than the zero-shot MQ4CS model with a constant value of $\phi = 3$ for the simple turns. Note that for the zero-shot MQ4CS model with a constant value of $\phi = 3$, the model can generate between 1–3 queries. In addition, according to Table 9 in Appendix D, our MQ4CS model outperforms the best baseline model (i.e., GPT4QR) over all metrics on all datasets except nDCG@3 for CAsT 19 & 20. This is in line with Figure 3, where we show 44% of the turns of CAsT 19 & 20 only require one query. So, considering a constant value of ϕ for datasets with less complex user utterances is sub-optimal. Nevertheless, our Oracle model significantly outperforms the baselines and even human-resolved utterances, reiterating the need for a flexible ϕ selection policy.

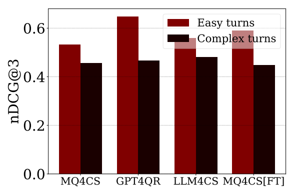
Aspect diversity. To assess whether the generated queries cover multiple aspects and how they compare to LLM4CS, we conduct both human evaluation study using Amazon Mechanical Turk (AMT) and automatic evaluations. In the study, we asked human assessors to do a pairwise comparison of LLM4CS- and MQ4CS- generated queries and compare them in terms of diversity (i.e., "Which set of queries cover more aspects of the original query?"). We show the five queries generated by zero-shot MQ4CS and the five queries generated by LLM4CS to the annotators. We asked four human annotators to assess each turn and decide the winner by the number of annotators who favored each system (e.g., 3 annotators chose MQ4CS vs. 1 annotator chose LLM4CS, making MQ4CS the



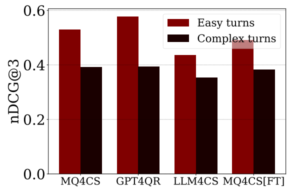
(a) iKAT 23



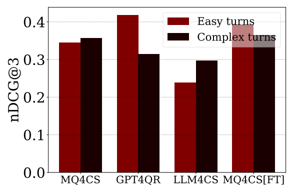
(b) iKAT 24



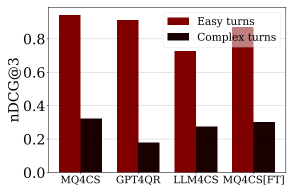
(c) CAsT 19



(d) CAsT 20



(e) CAsT 22



(f) TopiOCQA

Figure 5: A comparison between retrieval performance of MQ4CS and baselines over complex and easy turns.

winner). We ran the human annotation on all conversational turns of iKAT 23, and reported the results in Table 11 in Appendix E. As can be seen, in ∼45% of them MQ4CS wins the LLM4CS by generating more diverse queries, as well as ∼28% ties, confirming that our MQ4CS leads to more diverse, multi-aspect queries. We provide further details about our human evaluation study in Appendix E.

To further assess the diversity of the generated queries, we compute the pairwise cosine similarity between their embeddings, using the `all-MiniLM-L6-v2` model. This is done for each conversation turn, and the average similarity across all query pairs is then calculated. In addition, we assess retrieval diversity by calculating the percentage of unique passages among the top 10 retrieved by different queries for the same turn. We then average this percentage across all turns. We present these statistics in Table 3. As shown, the query similarity scores for the iKAT 23 & 24 and CAsT 22 datasets are notably lower than those for the other datasets, indicating higher diversity. In contrast, the diversity for CAsT 19 & 20 and TopiOCQA appears to be moderate. Furthermore, for iKAT 23 & 24 and CAsT 22 over 87% of the passages retrieved by different queries are unique, indicating a high degree of ranking diversity.

Rank list fusion. To assess the effectiveness of our fusion strategy, we evaluate min-max normalization followed by round robin against two es-

Table 3: Percentage of unique passages ($\mathcal{U}$) retrieved and semantic similarity ($\mathcal{S}$) between multi-aspect queries across all datasets. Higher $\mathcal{U}$ and lower $\mathcal{S}$ indicate greater diversity.

	iKAT 24	iKAT 23	CAsT 22	CAsT 20	CAsT 19	TopiOCQA
$\mathcal{U}(\uparrow)$	89.1	90.6	87.6	77.3	75.3	68.2
$\mathcal{S}(\downarrow)$	0.713	0.707	0.731	0.811	0.802	0.799

tablished rank fusion methods, namely Maximal Marginal Relevance (MMR) (Zheng and Fang, 2011) and Reciprocal Rank Fusion (RRF) (Cormack et al., 2009). The comparison is conducted using the MQ4CS model with identical experimental configurations. The results reported in Tables 12 and 13 in Appendix F, show that our method achieves superior performance in most cases, indicating that min–max normalization followed by round-robin selection provides a more effective mechanism for rank list fusion than the considered baselines.

Latency. We analyze the query generation and retrieval latency of our method and compare it with baseline approaches such as LLM4CS and GPT4QR. Our analysis shows that MQ4CS model achieve higher retrieval quality than LLM4CS while maintaining similar or lower retrieval latency and comparable generation cost. GPT4QR has slightly lower retrieval latency, but its generation cost is similar to MQ4CS. MQ4CS [FT] provides strong performance at zero additional cost, with lower retrieval latency than LLM4CS, demonstrating an efficient and effective trade-off between latency, cost, and quality. Full latency statistics, setup details, and hardware specifications are provided in Appendix G.

6 Conclusion

We study the effectiveness of using multi-aspect queries to enhance the retrieval for complex user queries in CS. We propose a retrieval framework that leverages an LLM to generate multi-aspect queries in the same LLM call and fuse their corresponding ranked lists. We show the effectiveness of our proposed models over complex user utterances. We release MASQ dataset, which includes the generated queries and the values of ϕ^* for each turn of the conversation to foster research in this area. Our proposed method, while being more effective than the SoTA LLM4CS, outperforms it significantly on all the datasets. We release MASQ dataset, which includes the generated queries and the values of ϕ^* per each turn of the conversation to foster research in this area.

7 Limitations

We propose to generate and use multi-aspect queries to enhance retrieval for complex user utterances. We rely on the intrinsic knowledge of LLMs to do reasoning and generate the multi-aspect queries. We observe that without fine-tuning, LLMs cannot determine the efficient number of multi-aspect queries for each user utterance based on its complexity. This is because of the gap between retrieval and query generation. We did not study other biases of different LLMs on query generation and response generation. When generating multi-aspect queries per utterance (up to $\phi = 3$), retrieval latency can increase. However, the retrieval pipeline can be done in parallel, without extra cost compared to one query rewrite.

8 Ethical considerations

Stressing the need to study and measure biases in Language Models (LLMs) when generating data, we think it could cause unexpected ethical issues. Consequently, we need to study the potential biases that exist in the data and formalize their impact on the final output of the model. While in this study, we propose to use the answers and queries generated by LLMs for retrieval models, we think these methods should be used carefully in real-world retrieval systems, and designers should consider these biases.

References

Vaibhav Adlakha, Shehzaad Dhuliawala, Kaheer Suleman, Harm de Vries, and Siva Reddy. 2022. Topiocqa: Open-domain conversational question answering with topic switching. *Trans. Assoc. Comput. Linguistics*, 10:468–483.

Mohammad Aliannejadi, Zahra Abbasiantaeb, Shubham Chatterjee, Jeffrey Dalton, and Leif Azzopardi. 2024a. Trec ikat 2023: A test collection for evaluating conversational and interactive knowledge assistants. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 819–829, New York, NY, USA. Association for Computing Machinery.

Mohammad Aliannejadi, Jacek Gwizdka, and Hamed Zamani. 2024b. Interactions with generative information retrieval systems. pages 47–71.

Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W. Bruce Croft. 2019. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 475–484. ACM.

Avishek Anand, Lawrence Cavedon, Matthias Hagen, Hideo Joho, Mark Sanderson, and Benno Stein. 2020. Dagstuhl seminar 19461 on conversational search: seminar goals and working group outcomes. volume 54, pages 3:1–3:11.

Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. 2021. Open-domain question answering goes conversational via question rewriting. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 520–534, Online. Association for Computational Linguistics.

Gordon V. Cormack, Charles L. A. Clarke, and Stefan Büttcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009*, pages 758–759. ACM.

Jeffrey Dalton, Chenyan Xiong, and Jamie Callan. 2020. Trec cast 2019: The conversational assistance track overview. *arXiv preprint arXiv:2003.13624*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library. *CoRR*, abs/2401.08281.

Ahmed Elgohary, Denis Peskov, and Jordan L. Boyd-Graber. 2019. Can you unpack that? learning to rewrite questions-in-context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5917–5923. Association for Computational Linguistics.

Mohamed Farah and Daniel Vanderpooten. 2007. An outranking approach for rank aggregation in information retrieval. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, page 591–598, New York, NY, USA. Association for Computing Machinery.

Song Feng, Siva Sankalp Patel, Hui Wan, and Sachindra Joshi. 2021. Multidoc2dial: Modeling dialogues grounded in multiple documents. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6162–6176. Association for Computational Linguistics.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE: sparse lexical and expansion model for first stage ranking. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 2288–2292. ACM.

Nam Le Hai, Thomas Gerald, Thibault Formal, Jian-Yun Nie, Benjamin Piwowarski, and Laure Soulier. 2023. Cosplade: Contextualizing SPLADE for conversational information retrieval. In *Advances in Information Retrieval - 45th European Conference on Information Retrieval, ECIR 2023, Dublin, Ireland, April 2-6, 2023, Proceedings, Part I*, volume 13980 of *Lecture Notes in Computer Science*, pages 537–552. Springer.

D. Frank Hsu and Isak Taksa. 2005. Comparing rank and score combination methods for data fusion in information retrieval. *Inf. Retr.*, 8(3):449–480.

Zhuoran Jin, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. 2023. Instructor: Instructing unsupervised conversational dense retrieval with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 6649–6675. Association for Computational Linguistics.

Ivica Kostric and Krisztian Balog. 2024. A surprisingly simple yet effective multi-query rewriting method for conversational passage retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, pages 2271–2275. ACM.

Yu Li, Baolin Peng, Yelong Shen, Yi Mao, Lars Liden, Zhou Yu, and Jianfeng Gao. 2022. Knowledge-grounded dialogue generation with a unified knowledge representation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 206–218. Association for Computational Linguistics.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021a. Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 2356–2362. ACM.

Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2021b. *Pretrained Transformers for Text Ranking: BERT and Beyond*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy Lin. 2020. Conversational question reformulation via sequence-to-sequence architectures and pretrained language models. *arXiv preprint arXiv:2004.01909*.

Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy Lin. 2021c. Multi-stage conversational passage retrieval: An approach to fusing term importance estimation and neural query rewriting. *ACM Trans. Inf. Syst.*, 39(4):48:1–48:29.

Simon Lupart, Mohammad Aliannejadi, and Evangelos Kanoulas. 2025. Disco: Llm knowledge distillation for efficient sparse retrieval in conversational search. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '25, page 9–19, New York, NY, USA. Association for Computing Machinery.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning. pages 305–329.

Kelong Mao, Zhicheng Dou, Fengran Mo, Jiewen Hou, Haonan Chen, and Hongjin Qian. 2023a. Large language models know your contextual search intent: A prompting framework for conversational search. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 1211–1225. Association for Computational Linguistics.

Kelong Mao, Hongjin Qian, Fengran Mo, Zhicheng Dou, Bang Liu, Xiaohua Cheng, and Zhao Cao. 2023b. Learning denoised and interpretable session representation for conversational search. In *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 3193–3202. ACM.

Fengran Mo, Kelong Mao, Yutao Zhu, Yihong Wu, Kaiyu Huang, and Jian-Yun Nie. 2023. Convgqr: Generative query reformulation for conversational search. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 4998–5012. Association for Computational Linguistics.

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. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, page 3954–3958, New York, NY, USA. Association for Computing Machinery.

Paul Owoicho, Jeff Dalton, Mohammad Aliannejadi, Leif Azzopardi, Johanne R. Trippas, and Svitlana Vakulenko. 2022. TREC cast 2022: Going beyond user ask and system retrieve with initiative and response generation. 500-338.

Paul Owoicho, Ivan Sekulic, Mohammad Aliannejadi, Jeffrey Dalton, and Fabio Crestani. 2023. Exploiting simulated user feedback for conversational search: Ranking, rewriting, and beyond. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, pages 632–642. ACM.

Hongjin Qian and Zhicheng Dou. 2022. Explicit query rewriting for conversational dense retrieval. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 4725–4737. Association for Computational Linguistics.

Filip Radlinski and Nick Craswell. 2017. A theoretical framework for conversational search. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval, CHIIR 2017, Oslo, Norway, March 7-11, 2017*, pages 117–126. ACM.

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. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Sudha Rao and Hal Daumé III. 2018. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2737–2746. Association for Computational Linguistics.

Rodrygo L. T. Santos, Craig MacDonald, and Iadh Ounis. 2015. Search result diversification. *Found. Trends Inf. Retr.*, 9(1):1–90.

Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2021. Question rewriting for conversational question answering. In *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual Event, Israel, March 8-12, 2021*, pages 355–363. ACM.

Nikos Voskarides, Dan Li, Andreas Panteli, and Pengjie Ren. 2019. ILPS at TREC 2019 conversational assistant track. In *Proceedings of the Twenty-Eighth Text REtrieval Conference, TREC 2019, Gaithersburg, Maryland, USA, November 13-15, 2019*, volume 1250 of *NIST Special Publication*. National Institute of Standards and Technology (NIST).

Nikos Voskarides, Dan Li, Pengjie Ren, Evangelos Kanoulas, and Maarten de Rijke. 2020. Query resolution for conversational search with limited supervision. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models.

Zeqiu Wu, Yi Luan, Hannah Rashkin, David Reitter, Hannaneh Hajishirzi, Mari Ostendorf, and Gaurav Singh Tomar. 2022. CONQRR: conversational query rewriting for retrieval with reinforcement learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 10000–10014. Association for Computational Linguistics.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval.

Fanghua Ye, Meng Fang, Shenghui Li, and Emine Yilmaz. 2023. Enhancing conversational search: Large language model-aided informative query rewriting. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 5985–6006. Association for Computational Linguistics.

Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020a. Few-shot generative conversational query rewriting. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 1933–1936.

Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul N. Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020b. Few-shot conversational dense retrieval. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1933–1936. ACM.

Hamed Zamani, Johanne R. Trippas, Jeff Dalton, and Filip Radlinski. 2023. Conversational information seeking. *Found. Trends Inf. Retr.*, 17(3-4):244–456.

Wei Zheng and Hui Fang. 2011. A comparative study of search result diversification methods. *Proc. of DDR*, pages 55–62.

A Prompts

We provide the prompts used for different modules in our proposed framework in this section. The sentences in purple in the prompts are only used for TREC iKAT datasets.

Table 4: The prompt designed for (1) Initial Answer Generation (AG) and (2) Query Generation (QG) in MQ4CS$_{\text{ans}}$ model variant. The sentences in purple are only used for iKAT datasets. The variables inside { } indicate the place holders for input. The r'_i and Q^i variables indicate the output generated by LLM.

(1) # Instruction: *I will give you a conversation between a user and a system. Also, I will give you some background information about the user. You should answer the last question of the user. Please remember that your answer to the last question of the user shouldn't be more than 200 words.*
# Background knowledge: $\{PTKB\}$
# Context: $\{c_i\}$
# User question: $\{u_i\}$
# Response: r'_i

(2) # *Imagine you want to retrieve your previous answer by searching Google. You should generate the unique search queries that you need to search in Google. Each query should cover one aspect of your answer. (Please write each query in one line and don't generate more than {ϕ} queries)*
# Generated queries: Q^i

Table 5: The prompt designed for QG function in MQ4CS model variant.

# Instruction: *I will give you a conversation between a user and a system. Also, I will give you some background information about the user. Imagine you want to find the answer to the last user question by searching Google. You should generate the unique search queries that you need to search in Google. Each query should cover one aspect of your answer. Please don't generate more than {ϕ} queries and write each query in one line.*
# Background knowledge: $\{PTKB\}$
# Context: $\{c_i\}$
# User question: $\{u_i\}$
# Generated queries: Q^i

B MASQ dataset

We provide the statistics of the MASQ dataset in Table 7. The total number of Oracle queries as well as the average value of ϕ^* for different datasets in shown. The statistics of the datasets that we use are provided in Table 8.

C Experimental Setup

We explain our baselines, the datasets, metrics, and hyper-parameters in the following. **Compared methods.** We compare our proposed models to six strong query rewriting (QR) baselines including

Table 6: The prompt designed for LLMQR using GPT4 and Llama models as a zero-shot learner.

# Instruction:*I will give you a conversation between a user and a system. Also, I will give you some background information about the user. You should rewrite the last question of the user into a self-contained query.*
# Background knowledge: $\{PTKB\}$
# Context: $\{c_i\}$
# Please rewrite the following user question: $\{u_i\}$
# Re-written query:

Table 7: Statistics of the MASQ dataset.

Dataset	# turns	# queries	Avg. ϕ^*
TREC CAsT 19	173	234	1.353
TREC CAsT 20	208	334	1.606
TREC CAsT 22	165	279	1.691
TREC iKAT 23	133	270	2.030
TREC iKAT 24	116	184	1.586
TopiOCQA	2514	4868	1.936

(1) *ConvGQR* (Mo et al., 2023) a pre-trained model for expanding the query rewrite with a potential answer; (2) *T5QR* (Lin et al., 2020) a T5-based query rewriting model which is trained on CANARD dataset; (3) *GPT4QR*, using GPT4 as a zero-shot learner for query rewriting based on the prompt shown in Table 6; (4) *LlamaQR*, zero-shot prompting the Llama for query rewriting using the same prompt (Table 6); (5) *HumanQR*, using the resolved-utterance by human; and (6) *LLM4CS* (Mao et al., 2023a), an LLM-based multiple query rewrite generation method. To ensure a fair comparison, we use the same retrieval and reranking pipeline for all baselines and our proposed methods. We reproduce the LLM4CS model, using the same LLM we use for our proposed model (i.e., GPT4) to ensure a fair comparison. We use RAR with mean aggregation and with $N = 5$ to generate LLM4CS queries for dense retrieval.[1] For sparse retrieval, we use Max-Prob strategy and concat the response to the query. For the ConvGQR model, we use the code released by authors to fine-tune the model on the QReCC dataset (Anantha et al., 2021). We use the T5QR

[1]The exact experimental setup they used for reporting the results in the main table of their paper.

Table 8: The statistics of the CS datasets.

Dataset	# conv.	# turns	conv. length
TREC CAsT 19	20	173	8.65
TREC CAsT 20	25	216	8.6
TREC CAsT 22	18	205	11.39
TREC iKAT 23	13	176	13.04
TREC iKAT 24	17	116	12.82
TopiOCQA	205	2,514	12

model available on HuggingFace.[2]We append the persona of the user (in iKAT 23 and 24 datasets) to the context of the conversation for baseline models.

Hyper-parameters. For the first-stage retrieval we employ the BM25 model from Pyserini (Lin et al., 2021a) using the default values for the parameters. For the reranker, we use the pre-trained Cross-Encoder model `ms-marco-MiniLM-L-6-v2` from the `sentence_transformers` library with a maximum length of 512. In MQ4CS$_{\text{ans}}$ and MQ4CS approaches we take the top 100 passages returned by BM25 and pass them to reranker. In other baselines, we rerank the top 100 passages returned by BM25. We use `Llama-3.1-8B-Instruct` with the following parameters: `top_k`=10, `top_p`=0.9, `temperature`=0.75 for LlamaQR method. For dense retrieval, we use ANCE [3] (Xiong et al., 2021) combined with FAISS (Douze et al., 2024) for nearest neighbor search. The lengths of the query, response, flat concatenation, and passage are truncated into 64, 256, 512, and 384, respectively, following (Mo et al., 2024). We conduct our experiments on a single A6000 GPU with 32 GB RAM. We use the GPT4 model as a zero-shot learner using the default values of parameters for all approaches. We use GPT4o model for TopiOCQA dataset. We fine-tune the Llama (`Llama-3.1-8B-Instruct`) model using parameter efficient fine tuning (PEFT) model, 4-bit QLoRA (Dettmers et al., 2023). We fine-tuned the model for 5 epochs using lr=2e-4, alpha=16, and attention=16. For TREC iKAT and TREC CAsT datasets, we use the Oracle dataset from other years as training data. For TopiOCQA dataset, we create the Oracle set on a random sample of 5000 user utterances from train set of TopiOCQA.

Dataset. We report the results on TopiOCQA, TREC iKAT 23 & 24, TREC CAsT 19, 20, & 22 datasets. We did not experiment on CAsT 21 dataset because it has a document-level evaluation rather than passage-level evaluation. The statistics of these datasets are shown in Table 8. The TopiOCQA is a large-size open-domain conversational dataset that incorporates topic shifts and is based on Wikipedia (Adlakha et al., 2022). The TREC iKAT 23 & 24 datasets are one of the few datasets that features complex dialogues and persona of the user where single-query rewriting is not effective. The average length of conversations in iKAT is 13.04 which makes the context modeling task more challenging.

Metrics. We evaluate passage retrieval performance using the official metrics used in the literature, namely, nDCG@3, Recall@100, and MRR. nDCG@3 evaluates the scenarios where the top passages are intended to be presented to the user. We calculate these metrics using the `trec_eval` tool. For TopiOCQA dataset we report MAP instead of nDCG@3 as each query in this dataset is associated with only one relevant passage, and relevance is evaluated at a binary level. We perform statistical significance tests using paired t-tests, comparing each proposed model against the GPT4QR baseline. Significance is assessed with Bonferroni correction. For the TopiOCQA dataset, with four evaluation metrics, there are 12 total comparisons ($\alpha_{\text{corrected}} = 0.0042$) and for the TREC iKAT and CAsT datasets, with three metrics, there are 9 total comparisons ($\alpha_{\text{corrected}} = 0.0167$).

D Results using Dense Retrieval

In this section, we report the performance of our proposed models and baselines over TREC CAsT datasets in Table 9; over TREC iKAT and TopiOCQA datasets in Table 10.

E Human Evaluation

We only chose workers located in the US, with English as their native language and having at least 95% of approval rate as a quality measure. We aligned our payment according to the minimum wage in the US ($7.25 per hour). Based on pilot runs, each assessment took approximately 65 seconds, resulting in a payment of $0.15 per HIT. Since our task involved direct pairwise comparisons between two systems, reporting overall worker agreement is less informative. Nevertheless, each pair of generated queries was assessed by four independent workers. Computing Fleiss' Kappa across all comparisons yields 0.2032, indicating moderate agreement. When excluding ties (cases where the two systems were judged equally effective), Fleiss' Kappa increases to 0.41, reflecting substantially stronger agreement among workers.

F Ranked list Fusion

The comparison between the proposed ranked list fusion method ($Fuse$) and other ranked list fusion models is provided in Tables 12 and 13.

[2]https://huggingface.co/castorini/t5-base-canard
[3]3ricL/ad-hoc-ance-msmarco

Table 9: Performance of sparse retrieval (BM25) with re-ranking and dense retrieval. The best results are shown **bold** and the the best result outperforming human is shown with underline. We use $\phi = 3$ for MQ4CS and MQ4CS$_{\text{ans}}$ models.

Type	Method	LLM	CAsT 20			CAsT 22			CAsT 19		
			nDCG@3	R@100	MRR	nDCG@3	R@100	MRR	nDCG@3	R@100	MRR
Sparse + reranking	T5QR	-	38.7	45.6	53.1	30.2	23.9	45.5	**56.5**	59.8	74.6
	ConvGQR	-	35.7	47.7	49.4	25.0	22.1	41.0	50.2	53.6	67.5
	LLM4CS	GPT4	38.7	51.1	54.5	27.5	30.0	45.3	52.2	55.8	72.7
	LlamaQR	Llama 3.1	36.3	49.3	51.4	30.8	27.3	49.0	-	-	-
	GPT4QR	GPT4	**46.8**	55.2	61.5	34.8	30.1	52.2	**56.5**	62.0	73.8
	HumanQR	-	50.5	61.8	66.8	41.3	37.6	60.4	-	-	-
	MQ4CS$_{\text{ans}}$	GPT4	43.6	59.5	**75.3**	**36.1**	31.3	**69.2**	45.3	59.9	71.4
	MQ4CS	GPT4	44.8	**64.8**	64.3	35.9	**36.3**	59.8	52.6	**64.1**	**75.8**
	MQ4CS [FT]	Llama 3.1	42.6	62.0	61.9	34.1	32.3	55.5	52.1	59.5	70.9
Dense (ANCE)	T5QR	-	34.3	40.1	49.5	27.2	25.2	40.6	45.1	42.0	64.9
	ConvGQR	-	32.4	40.2	46.3	23.7	21.3	36.2	43.2	41.0	60.6
	LLM4CS	GPT4	**44.5**	50.4	**61.8**	27.3	28.3	42.5	**48.9**	**50.6**	**67.4**
	GPT4QR	GPT4	41.5	49.2	57.6	**35.4**	**33.1**	**56.3**	42.7	45.0	61.8
	HumanQR	-	44.7	51.5	61.1	40.9	36.0	58.8	-	-	-
	MQ4CS$_{\text{ans}}$	GPT4	36.2	**51.7**	56.3	29.9	32.8	49.7	38.2	44.9	55.4
	MQ4CS	GPT4	35.9	49.7	55.9	31.2	32.4	53.0	38.4	45.7	58.9
	MQ4CS [FT]	Llama 3.1	34.4	49.0	51.4	30.1	28.7	48.9	39.8	42.6	58.8

Table 10: Performance of sparse retrieval (BM25) with re-ranking and dense retrieval. The best results are shown **bold** and the the best result outperforming human is shown with underline. We use $\phi = 3$ for MQ4CS and MQ4CS$_{\text{ans}}$ models.

Type	Method	LLM	iKAT 23			iKAT 24			TopiOCQA			
			nDCG@3	R@100	MRR	nDCG@3	R@100	MRR	MAP	R@10	R@100	MRR
Sparse + reranking	T5QR	-	14.1	13.8	23.9	23.5	19.8	34.9	33.5	50.2	62.5	33.5
	ConvGQR	-	14.7	13.9	21.9	24.1	19.3	35.6	31.1	49.0	63.2	31.1
	LLM4CS	GPT4	10.5	14.5	16.5	14.9	17.8	25.8	36.8	57.1	75.9	36.8
	LlamaQR	Llama 3.1	9.6	14.2	16.0	-	-	-	-	-	-	-
	GPT4QR	GPT4	21.9	22.1	34.6	45.4	34.3	66.2	45.4	66.9	80.9	45.4
	HumanQR		30.7	35.8	43.3	43.7	32.7	62.3	-	-	-	-
	MQ4CS$_{\text{ans}}$	GPT4	**24.8**	**29.1**	**41.2**	44.8	**35.9**	65.5	46.7	70.6	87.0	46.7
	MQ4CS	GPT4	22.0	27.2	39.1	**46.6**	35.0	**69.2**	**47.5**	**72.6**	**87.8**	**47.5**
	MQ4CS [FT]	Llama 3.1	14.6	14.8	24.1	30.0	31.1	52.5	41.4	67.2	79.6	41.4
Dense (ANCE)	T5QR	-	10.8	12.4	17.3	13.6	15.5	22.6	15.9	28.4	43.3	15.9
	ConvGQR	-	11.7	12.6	19.1	12.7	16.8	21.7	19.9	34.5	50.5	19.9
	LLM4CS	GPT4	10.0	13.4	15.5	14.4	16.8	23.0	**30.4**	47.3	64.4	**30.4**
	GPT4QR	GPT4	**18.2**	22.4	**29.8**	**31.5**	29.6	47.9	27.1	47.6	66.1	27.1
	HumanQR		20.0	24.8	30.5	28.2	25.7	44.4	-	-	-	-
	MQ4CS$_{\text{ans}}$	GPT4	16.9	**24.9**	27.9	27.8	**31.1**	46.4	28.0	**51.4**	**71.7**	28.0
	MQ4CS	GPT4	16.6	22.4	28.7	29.1	30.1	**49.5**	26.9	48.8	69.2	26.9
	MQ4CS [FT]	Llama 3.1	8.1	14.5	17.3	23.3	24.4	42.3	25.2	46.0	66.5	25.2

Table 11: Human evaluation. Pairwise comparison between the queries generated by MQ4CS and LLM4CS on iKAT 23.

	# turns	% turns
MQ4CS wins	59	44.3%
LLM4CS wins	37	27.8%
Ties	37	27.8%

G Latency Analysis

We evaluate latency for both query generation and retrieval using our proposed models across different LLM backbones and report the results in Table 14. All experiments are conducted on the iKAT 24 dataset, using a server equipped with 32 Intel Xeon Gold 5118 CPUs (2.30 GHz) and a single NVIDIA RTX A6000 GPU with 32 GB of memory. Query generation is performed either with locally

Approach	Fusion	R@10	R@100
MQ4CS	our	**69.0**	**80.2**
	MMR	67.3	**80.2**
	RRF	57.5	79.8

Table 12: Performance comparison of rank fusion methods on the TopiOCQA dataset.

hosted models (Llama-3.1, 8B parameters) or via OpenAI API (GPT4 and GPT4o). Retrieval experiments are performed using sparse retrieval (BM25) with re-ranking. When using open source models (e.g., GPT4 and GPT4o), we report end-to-end latency measured from the client side. However, we do not have access to internal model or system latency, and thus these measurements are not directly comparable to locally deployed models (i.e., MQ4CS [FT] using Llama).

Based on Table 14 we observe that our pro-

Dataset	Fusion	nDCG@3	R@100	MRR
iKAT 24	Our	**38.3**	28.9	**62.1**
	MMR	37.0	**29.9**	57.8
	RRF	35.8	28.2	54.5
iKAT 23	Our	**22.1**	21.8	**38.2**
	MMR	21.9	22.0	34.5
	RRF	21.1	**22.4**	33.5
CAsT 20	Our	44.3	**54.8**	62.1
	MMR	**44.9**	54.5	**62.2**
	RRF	41.6	53.6	58.8
CAsT 19	Our	50.1	57.5	**75.7**
	MMR	**54.7**	**58.2**	74.2
	RRF	51.9	57.4	72.4

Table 13: Performance comparison of rank fusion methods on the iKAT and CAsT datasets.

posed MQ4CS model achieves substantially higher retrieval quality (nDCG@3 = 46.6) compared to LLM4CS (14.9), while maintaining a similar generation latency (6.74 s vs. 6.89 s) and a lower retrieval cost (0.632 s vs. 1.033 s). This shows that our approach is both more effective and more efficient in terms of retrieval latency. The GPT4QR baseline model exhibits slightly lower retrieval latency (0.289 s) than MQ4CS; however, its generation latency and cost are comparable (6.66 s and $ 0.0034), indicating that MQ4CS provides better overall quality without additional overhead. Moreover, our fine-tuned model, MQ4CS [FT], does not incur additional API cost and achieves a higher retrieval quality (30.0 nDCG@3) with a lower retrieval latency (0.626 s) compared to LLM4CS, further confirming the efficiency and effectiveness of our approach when using a locally hosted model Llama.

Table 14: Latency of query generation, latency of retrieval, cost, and quality (nDCG@3) of different models using GPT4. All latencies are in seconds.

Model	Latency (Generation)	Latency (Retrieval)	Cost	nDCG@3	Model size
MQ4CS	11.44	0.632	\$0.0367	46.6	unknown
MQ4CS$_{\text{ans}}$	24.46	0.632	\$0.0782	44.8	unknown
GPT4QR	11.31	0.289	\$0.0363	45.4	unknown
LLM4CS	11.71	1.033	\$0.0375	14.9	unknown
MQ4CS [FT]	7.581	0.626	0	30.0	8B parameters