passage retrieval in two situations: (1) the generated answer is correct, so a passage containing the same answer could be favored; (2) the generated answer is not a correct answer, but it co-occurs with a correct answer in a passage. This can also help determine the correct passage, and this is indeed the very assumption behind many query expansion approaches used in IR. Motivated by this, we use another PLM to generate the potential answer to expand the current query.

The overview of our proposed ConvGQR is depicted in Fig. 2. It contains three main components: query rewriting, query expansion, and knowledge infusion mechanism. The last component connects query reformulation and retrieval.

3.2.1 Query Reformulation by Combining Rewriting and Expansion

Both query rewriting and expansion use the historical context $H^k = \{q_k, r_k\}_{k=1}^{i-1}$ concatenated with the current query q_i as input. Similar to the input used in Wu et al. (2021), a separation token "[SEP]" is added between each turn and the turns are concatenated in reversed order, as in Eq. 1.

$$\mathrm{S} = \texttt{[CLS]}\ q_i\ \texttt{[SEP]}\ q_{i-1} \cdots q_1\ \texttt{[SEP]}. \quad (1)$$

Query Rewriting The objective of the query rewriting model is to induce a function $\mathcal{M}(H^k, q_i) = q^*$ based on a generative PLM, where q^* is a sequence used as the supervision signal (which is a human-rewritten query in the training data). Then, the information contained in H^k but missing in q_i can be added to approach q^*. Finally, the overall objective can be viewed as optimizing the parameter $\theta_{\mathcal{M}}$ of the function $\mathcal{M}$ by maximum likelihood estimation:

$$\theta_{\mathcal{M}} = \arg\max_{\theta_{\mathcal{M}}} \prod_{k=1}^{i-1} \Pr\left(q^* | \mathcal{M}\{H^k, q_i\}, \theta_{\mathcal{M}}\right). \quad (2)$$

Query Expansion Recent research demonstrates that the current PLMs have the ability to directly respond to a question as a close-book question answering system (Adlakha et al., 2022) through its captured knowledge. Although the correctness of the generated answer is not guaranteed, the potential answer can still act as useful expansion terms (Mao et al., 2021), which can guide the search toward a passage with the potential answer or a similar answer. To train the generation process, we leverage the gold answer r^* for each query turn as the training objection. r^* could be a short entity, a consecutive segment of text, or even non-consecutive text segments, depending on the dataset. In inference for a new query, the potential answers are generated by the query expansion model and used to expand the previously rewritten query.

The final form of the reformulated query is the concatenation of the rewritten query and the generated potential answer. The two generative PLMs for rewriting and expansion are fine-tuned with the negative log-likelihood loss to predict the corresponding target with an input sequence $\{w_t\}_{t=1}^{T}$ as Eq. 3, however, with different training data.

$$\mathcal{L}_{\text{gen}} = -\sum_{t=1}^{T} \log\left(\Pr(w_t | w_{1:t-1}, H^k, q_i)\right). \quad (3)$$

3.2.2 Knowledge Infusion Mechanism

An important limitation of the existing generative conversational query reformulation methods is that they ignore the dependency between generation and retrieval. They are trained independently. To address this issue, we propose a knowledge infusion mechanism to optimize both query reformulation and search tasks during model training. The intuition is to require the generative model to generate a query representation that is similar to that of a relevant passage. If the hidden states of the generative model contain the information of the relevant passage, the queries generated by these representations would be able to improve the search results because of the increased semantic similarity.

To achieve this goal, an effective way is to inject the knowledge included in the relevant passage representation into the query representation when fine-tuning the generative PLMs. Concretely, we first deploy an *off-the-shelf* retriever acting as an encoder to produce a representation $\mathbf{h}_{p_+}$ for the relevant passage. To maintain consistency, the retriever is the same as the one we use for search. Thus, the representation space for passages is kept the same for both query reformulation and retrieval stages. Once the session query representation $\mathbf{h}_S$ is encoded by the generative model, we distill the knowledge of $\mathbf{h}_{p_+}$ and infuse it into the $\mathbf{h}_S$ by minimizing the Mean Squared Error (MSE) as Eq. 4. Both $\mathbf{h}_S$ and $\mathbf{h}_{p_+}$ are sequence-level representations based on the first special token "[CLS]". Finally, the overall training objective $\mathcal{L}_{\text{ConvGQR}}$ consists of query generation loss $\mathcal{L}_{\text{gen}}$ and retrieval loss $\mathcal{L}_{\text{ret}}$. A weight factor α is used to balance the influence of query generation and retrieval.

$$\mathcal{L}_{\text{ret}} = \text{MSE}(\mathbf{h}_S, \mathbf{h}_{p_+}), \tag{4}$$

$$\mathcal{L}_{\text{ConvGQR}} = \mathcal{L}_{\text{gen}} + \alpha \cdot \mathcal{L}_{\text{ret}}. \tag{5}$$

3.3 Training and Inference

Two generative models with different targets for query rewriting and expansion are trained separately. The final output of the ConvGQR is the concatenation of the rewritten query and the generated potential answer. The knowledge infusion mechanism is applied only for the training stage, which guides optimization toward both generation and retrieval. The dense retriever is frozen to encode passages for generative PLMs training.

3.4 Retrieval Models

We apply ConvGQR to both dense and sparse retrieval models. We use ANCE (Xiong et al., 2020) fine-tuned on the MS MARCO (Bajaj et al., 2016), which achieves state-of-the-art performance on several retrieval benchmarks, as the dense retriever. The sparse retrieval is the traditional BM25.

4 Experiments

Datasets Following previous studies (Wu et al., 2021; Kim and Kim, 2022), four conversational search datasets are used for our experiments. The TopiOCQA (Adlakha et al., 2022) and QReCC (Anantha et al., 2021) datasets are used for normal query reformulation training. Two other widely used TREC CAsT datasets (Dalton et al., 2020, 2021) are only used for zero-shot evaluation as no training data is provided. The statistics and more details are provided in Appendix A.

Evaluation Metrics To evaluate the retrieval results, we use four standard evaluation metrics: MRR, NDCG@3, Recall@10 and Recall@100, as previous studies (Anantha et al., 2021; Adlakha et al., 2022; Mao et al., 2022a). We adopt the `pytrec_eval` tool (Van Gysel and de Rijke, 2018) for metric computation.

Baselines We mainly compare `ConvGQR` with the following query reformulation (QR) baselines for both dense and sparse retrieval: (1) `Raw`: The query of current turn without reformulation. (2) `GPT2QR` (Anantha et al., 2021): A strong GPT-2 (Radford et al., 2019) based QR model. (3) `CQE-sparse` (Lin et al., 2021b): A weakly-supervised method to select important tokens only from the context via contextualized query embeddings. (4) `QuReTeC` (Voskarides et al., 2020): A weakly-supervised method to train a sequence tagger to decide whether each term contained in historical context should be added to the current query. (5) `T5QR` (Lin et al., 2020): A strong T5-based (Raffel et al., 2020) QR model. (6) `ConvDR` (Yu et al., 2021): A strong ad-hoc search retriever fine-tuned on conversational search data using knowledge distillation between the rewritten query representation and the historical context representation. (7) `CONQRR` (Wu et al., 2021): A reinforcement-learning and T5-based QR model which adopts both BM25 and conversational fine-tuned T5-encoder as retrievers. Note that `CQE-sparse` and `ConvDR` need to train a new conversational query encoder to determine the relevance between the long context and passages, while the other baseline methods and our `ConvGQR` are based on the off-the-shelf retriever only.

For *zero-shot* scenario, in addition to the `QuReTeC` method originally fine-tuned on QuAC datasets (Choi et al., 2018), we also perform comparisons with (8) `Transformer++` (Vakulenko et al., 2021a): A GPT-2 based QR model fine-tuned on CANARD dataset (Elgohary et al., 2019). (9) `Query Rewriter` (Yu et al., 2020): A GPT-2 based QR model fine-tuned on large-scale search session data. Besides, the results of `Human-Rewritten` queries in the original datasets are also provided.

Implementation Details We implement the generative PLMs for `ConvGQR` based on T5-base (Raffel et al., 2020) models. When fine-tuning the generative PLMs, the dense retriever is frozen and acts as a passage encoder. For the zero-shot scenario, we use the generative models trained on QReCC to produce the reformulated queries and retrieve relevant passages. The dense retrieval and sparse retrieval (BM25) are performed using Faiss (Johnson et al., 2019) and Pyserini (Lin et al., 2021a), respectively. More details are provided in Appendix A and our released code[2].

4.1 Main Results

Main evaluation results on QReCC and TopiOCQA are reported in Table 1.

[2] `https://github.com/fengranMark/ConvGQR`

Type	Method	QReCC				TopiOCQA			
		MRR	NDCG@3	R@10	R@100	MRR	NDCG@3	R@10	R@100
Dense	Raw	10.2	9.3	15.7	22.7	4.1	3.8	7.5	13.8
	GPT2QR	33.9	30.9	53.1	72.9	12.6	12.0	22.0	33.1
	CQE-sparse	32.0	30.1	51.3	70.9	-	-	-	-
	QuReTeC	35.0	32.6	55.0	72.9	11.2	10.5	20.2	34.4
	T5QR	34.5	31.8	53.1	72.8	23.0	22.2	37.6	54.4
	ConvDR	38.5	35.7	58.2	77.8	-	-	-	-
	CONQRR	41.8	-	**65.1**	**84.7**	-	-	-	-
	ConvGQR (Ours)	**42.0**‡	**39.1**‡	63.5	81.8	**25.6**‡	**24.3**‡	**41.8**‡	**58.8**‡
	Human-Rewritten	38.4	35.6	58.6	78.1	-	-	-	-
Sparse	Raw	6.5	5.5	11.1	21.5	2.1	1.8	4.0	9.2
	GPT2QR	30.4	27.9	50.5	82.3	6.2	5.3	12.4	26.4
	CQE-sparse	31.8	29.2	52.9	83.4	-	-	-	-
	QuReTeC	34.0	30.5	55.5	86.0	8.5	7.3	16.0	31.3
	T5QR	33.4	30.2	53.8	86.1	11.3	9.8	22.1	44.7
	CONQRR	38.3	-	60.1	**88.9**	-	-	-	-
	ConvGQR (Ours)	**44.1**‡	**41.0**‡	**64.4**‡	88.0	**12.4**‡	**10.7**‡	**23.8**‡	**45.6**‡
	Human-Rewritten	39.7	36.2	62.5	98.5	-	-	-	-

Table 1: Performance of dense and sparse retrieval with query reformulation methods on two datasets. ‡ denotes significant improvements with t-test at $p < 0.05$ over all compared methods (except CONQRR). **Bold** and underline indicate the best and the second best result (except Human-Rewritten).

	QReCC		TopiOCQA	
	MRR	NDCG@3	MRR	NDCG@3
ConvGQR	**42.0**	**39.1**	**25.6**	**24.3**
- infusion	41.5	38.7	25.0	23.7
- expansion	36.9	33.9	24.6	23.3
- both	36.4	33.5	23.4	22.5

Table 2: Ablation study of different components.

We find that ConvGQR achieves significantly better performance on both datasets in terms of MRR and NDCG@3 and outperforms other methods on most metrics, either with dense retrieval or sparse retrieval. For example, on QReCC with sparse retrieval, it improves 15.1% MRR and 33.9% NDCG@3 over the second best results. This indicates the strong capability of ConvGQR on retrieving relevant passages at top positions. These results demonstrate the strong effectiveness of our method. Besides, we notice that CONQRR, which also leverages the downstream retrieval information but with reinforcement learning, may achieve better performance on some recall metrics, indicating that the downstream retrieval information is helpful to conversational search and should be carefully exploited.

Moreover, we find ConvGQR can even perform better than human-rewritten queries on QReCC. It confirms our earlier assumption that the human-rewritten query (oracle query) is not the silver bullet for conversational search. This finding is consistent with some recent studies (Lin et al., 2021b; Wu et al., 2021; Mao et al., 2023b). The improvements of ConvGQR over human-rewritten queries are mainly attributed to our query expansion and knowledge infusion, which introduce retrieval signals to the learning of query reformulation.

4.2 Ablation Study

Compared to a standard query rewriting method, our proposed ConvGQR has two additional components, *i.e.*, a query expansion component based on generated potential answers and a knowledge infusion mechanism. We investigate the impact of different components by conducting an ablation study on both QReCC and TopiOCQA. The results are shown in Table 2. We observe that removing any component leads to performance degradation and removing all of them drops the most. In fact, when both components are removed, ConvGQR degenerates to the T5QR model. The improvement of ConvGQR over T5QR directly reflects the gains brought by query expansion and knowledge infusion. The above analysis confirms the effectiveness of the added components.