Rethinking On-policy Optimization for Query Augmentation

Zhichao Xu¹, Shengyao Zhuang², Xueguang Ma³, Bingsen Chen⁴, Yijun Tian⁵, Fengran Mo⁶, Jie Cao⁷, Vivek Srikumar¹

¹University of Utah, ²University of Queensland, ³University of Waterloo, ⁴New York University, ⁵University of Notre Dame, ⁶Université de Montréal, ⁷University of Oklahoma

zhichao.xu@utah.edu svivek@cs.utah.edu

Abstract

Recent advances in large language models (LLMs) have led to a surge of interest in query augmentation for information retrieval (IR). Two main approaches have emerged. The first prompts LLMs to generate answers or pseudodocuments that serve as new queries, relying purely on the model's parametric knowledge or contextual information. The second applies reinforcement learning (RL) to fine-tune LLMs for query rewriting, directly optimizing retrieval metrics. While having respective advantages and limitations, the two approaches have not been compared under consistent experimental conditions. In this work, we present the first systematic comparison of prompting-based and RL-based query augmentation across diverse benchmarks, including evidence-seeking, ad hoc, and tool retrieval. Our key finding is that simple, training-free query augmentation often performs on par with, or even surpasses, more expensive RL-based counterparts, especially when using powerful LLMs. Motivated by this discovery, we introduce a novel hybrid method, On-policy Pseudo-document Query Expansion (OPQE), which, instead of rewriting a query, the LLM policy learns to generate a pseudo-document that maximizes retrieval performance, thus merging the flexibility and generative structure of prompting with the targeted optimization of RL. We show OPQE outperforms both standalone prompting and RL-based rewriting, demonstrating that a synergistic approach yields the best results. Our implementation is made available to facilitate reproducibility.1

1 Introduction

In many real-world retrieval scenarios, direct modification of the retriever is not possible — for example, when relying on API-based services such as OpenAI's text-embedding-3 or proprietary sys-

tems like PubMed². In such settings, improving retrieval effectiveness must come from the query side. One common approach is supervised finetuning (SFT), where a model is trained on humanor weakly-annotated pairs of original queries and rewritten (or clarified/expanded) queries (Yu et al., 2020; Mo et al., 2023). However, creating or obtaining high-quality query rewrite training data is nontrivial: humans may disagree about what makes a query "better", domain shifts can limit transfer, and annotated datasets are expensive to build (Yu et al., 2020; Ma et al., 2023; Coelho et al., 2025). Moreover, for dense retrievers the notion of what constitutes a "good" query is less well understood given their black-box neural architectures (Faggioli et al., 2023; Meng et al., 2023). Therefore, there has been growing research interest for unsupervised or weakly-supervised query augmentation method, compared to data-centric SFT (Wu et al., 2022).

Large language models (LLMs) have been applied to query augmentation in two ways: (1) by rewriting queries via SFT using (query, rewritten query) pairs (or weak supervision) to improve query clarity or specificity; and (2) by rewriting or expanding queries in a zero- or few-shot manner, without additional training (Li et al., 2025). Prompt-based query augmentation methods such as HyDE (Gao et al., 2023) and Query2Doc (Wang et al., 2023) fall in the latter category: expanding queries with pseudo documents generated from an LLM's paramatric knowledge can significantly improve retrieval effectiveness. These zero- or fewshot approaches require no additional training and are simple to apply. However, prior works (Abe et al., 2025; Lei et al., 2024) have questioned these methods' robustness, as generating a fluent or seemingly relevant document does not always guarantee a high-quality query representation. LLMs' verbose outputs can introduce spurious terms, which

https://github.com/BaleChen/RethinkQAug

²https://pubmed.ncbi.nlm.nih.gov/

may harm performance, particularly for sparse retrievers like BM25 (Robertson et al., 1995).

Beyond fine-tuning and prompting-based augmentation, another promising direction is to apply reinforcement learning (RL, Sutton et al., 1998), and in particular on-policy optimization, to query generation. Unlike methods that rely purely on prompt-based expansion, RL allows models to adapt their outputs based on retrieval-derived rewards, directly tying the generation process to ranking metrics such as recall, accuracy, or NDCG. This paradigm shifts the supervision signal from hand-crafted query–rewrite pairs to feedback grounded in retrieval effectiveness.

Recent work such as DeepRetrieval (Jiang et al., 2025) represents one instantiation of this idea. It combines reasoning-style generation, inspired by frameworks like ReAct (Yao et al., 2023) and DeepSeek-R1 (Guo et al., 2025), with retrievalbased rewards to train LLMs that produce both intermediate reasoning steps and final queries. While DeepRetrieval demonstrates encouraging results across tasks including literature search, IR benchmarks, and text-to-SQL (Liu et al., 2025), the approach remains resource-intensive and dependent on relevance-labeled data. More broadly, the field still lacks a systematic comparison between prompting-based query augmentation methods and RL-based formulations, leaving open questions about their respective strengths, weaknesses, and applicability in real-world settings.

In this work, we address this gap through a rigorous evaluation and benchmarking. Our contributions are threefold:

- We perform a controlled comparison of Deep-Retrieval against a simple, zero-shot prompting baseline across diverse retrieval tasks, including classical benchmarks and the emerging agent tool retrieval scenario.
- Our experiments reveal a surprising and critical finding: the simple, training-free prompting method often matches or surpasses the complex and expensive RL-trained method. This observation suggests that leveraging the rich, structured knowledge of a powerful LLM via pseudodocument generation is a strong and resource-efficient baseline.
- Based on this insight, we propose a novel hybrid method, On-policy Pseudo-document Query Expansion (OPQE), that combines the strengths

of both paradigms. Our rationale is that while prompting provides a better *structure* for the augmented query (a pseudo-document), RL provides a powerful *optimization* mechanism. Our proposed modification reframes the RL task: instead of training the policy to rewrite a query, we train it to generate a full pseudo-document and use retrieval metrics as the reward. This synergy allows the model to leverage the rich generative format of prompting while fine-tuning the output to maximize retrieval scores. Our experiments confirm that this hybrid approach consistently achieves the best overall performance, outperform both prompt-based expansion method and RL-based query rewriting method.

2 Related Works

Query Expansion and Reformulation. Classical query expansion techniques include pseudorelevance feedback (Rocchio, 1971; Voorhees, 1994), translation-based expansion (Cao et al., 2009), and statistical term weighting (Zhai and Lafferty, 2001). These approaches improve recall by introducing additional surface-level terms, but their effectiveness is often limited by noise sensitivity, especially for sparse retrievers.

More recently, LLMs have enabled zero- and few-shot query rewriting. Methods such as HyDE (Gao et al., 2023) and Query2Doc (Wang et al., 2023) prompt an LLM to generate pseudo-documents to be encoded by retrievers. Related approaches like ThinkQE (Lei et al., 2025) further incorporate reasoning before query generation. These methods are attractive because they are training-free and are directly applicable to API-based retrievers. However, their outputs are not explicitly optimized for retrieval metrics, and spurious terms may harm sparse retrieval performance (Abe et al., 2025; Lei et al., 2024). Refer to Li et al. (2025) for an in-depth survey.

Learning to Rewrite Queries. Beyond static prompting, several works explore trainable models for query reformulation. A prominent approach is supervised fine-tuning (SFT), where pre-trained sequence-to-sequence models like T5 (Raffel et al., 2020) are trained on datasets of original and rewritten queries (Wu et al., 2022; Liu et al., 2021). This technique is widely used in conversational search to transform ambiguous, context-dependent user utterances into clear, self-contained queries suitable for standard retrieval systems. However, the cre-

mance (Wu et al., 2022; Ye et al., 2023). To address this weakness, recent efforts have turned to weakly supervised methods. One such strategy involves knowledge distillation, where a powerful large language model (LLM) acts as a "teacher" to generate a large synthetic dataset of rewritten queries. A smaller, more efficient "student" model is then finetuned on this data, learning the rewriting task without expensive human annotation (Ye et al., 2023). This methodology has proven effective within the "Rewrite-Retrieve-Read" framework for retrievalaugmented generation, where the initial query is adapted to be more effective for the downstream retrieval and reading components (Ma et al., 2023). RL for Query Optimization. A growing line of work applies RL to optimize queries directly against retrieval-based rewards. Unlike supervised reformulation, which learns from a fixed dataset, RL enables on-policy exploration, directly tying the training signal to retrieval effectiveness (Wang et al., 2020; Wu et al., 2022). This is particularly valuable in dynamic contexts like conversational search, where an RL agent can be trained to reformulate queries to maximize the performance of a dense retriever (Zhu et al., 2025). This method's key advantage is its ability to optimize a rewriter for a black-box retrieval system, where the retriever's parameters are inaccessible for fine-tuning. The rewriter is treated as a policy model, and it receives a reward based on the performance of the retriever, allowing it to adapt and generate more effective queries. While powerful, these methods can require substantial compute for exploration and often depend on carefully engineered reward functions to guide the learning process (Sheng et al., 2025; Lambert et al., 2024).

ation of high-quality, human-labeled datasets can

be a significant bottleneck, and human rewrites

may not always be optimal for retrieval perfor-

Our work conducts a rigorous empirical comparison of prompting-based approaches vs. RL-based approaches. We highlight their relative strengths and trade-offs across sparse and dense retrieval settings, focusing on scenarios where retrievers are not finetunable and must be treated as black-box systems.

3 Background

Prompt-based Query Augmentation. Query augmentation aims to rewrite or expand an original query to improve retrieval effectiveness. Sparse

methods such as BM25 (Robertson et al., 1995) benefit directly from term-level expansions, while dense retrievers, which embed queries and documents into a shared vector space, pose a greater challenge since it is unclear what constitutes an effective query embedding. Collecting human-labeled training data for this task is expensive and often infeasible.

To reduce the dependency on labeled data, recent methods exploit the zero- and few-shot generative capabilities of LLMs. A common approach is to prompt an LLM to generate a pseudo-document that might answer the query. This pseudo-document is then encoded by the retriever to perform retrieval. Such prompt-based pseudo-document expansion avoids the need for explicit query-rewrite supervision and has been shown to be effective across open-domain QA and web search benchmarks (Gao et al., 2023; Wang et al., 2023). However, its success is tightly coupled to the scale and internal knowledge of the LLM, raising questions about how to enable smaller models to serve as effective query rewriters.

RL-based Query Augmentation. Another approach is to optimize query rewriting via RL. Here, the query rewriter is treated as a policy $\pi_{\theta}(q' \mid q)$ that generates an augmented query q' given the original query q. The process can be formalized as a Markov Decision Process (MDP), where **State** s_t denotes the current context, typically the original query q and the intermediate reasoning steps. **Action** a_t means generating the next token or subquery in q'. Policy $\pi_{\theta}(a_t \mid s_t)$ denotes LLM parameterized by θ . **Reward** $r(s_t, a_t)$ is derived from retrieval effectiveness of the completed query q', e.g., Recall@K, NDCG. The objective is to maximize the expected retrieval reward, with a KLregularization term relative to a reference policy $\pi_{\rm ref}$ to stabilize training and maintain generation fluency:

$$J_{\mathrm{KL}}(\theta) = \mathbb{E}_{q' \sim \pi_{\theta}(\cdot|q)} \Big[r_{\mathrm{retrieval}}(q') - \beta \, \mathrm{KL} \big(\pi_{\theta}(\cdot \mid q) \, \| \, \pi_{\mathrm{ref}}(\cdot \mid q) \big) \Big]. \quad (1)$$

This objective can be optimized with on-policy RL methods such as Proximal Policy Optimization (PPO, Schulman et al., 2017).

Reasoning-Style Query Generation. Recent works have demonstrated the benefit of generating structured outputs, separating intermediate reasoning from the final answer. Frameworks like

Algorithm 1 Query Augmentation with PPO

- 1: Initialize LLM policy π_{θ} and reference policy π_{ref}
- 2: **for** each query q in dataset **do**
- 3: Generate augmented query $q' \sim \pi_{\theta}(\cdot \mid q)$
- 4: Compute retrieval reward $r_{\text{retrieval}}(q')$
- 5: Compute objective $J_{\text{KL}}(\theta)$
- 6: Update θ via PPO
- 7: end for

ReAct (Yao et al., 2023) and DeepSeek-R1 (Guo et al., 2025) inspire approaches where the policy first generates reasoning steps h_1, \ldots, h_n and then a final augmented query q'_{final} . The retrieval reward is computed against q'_{final} , while intermediate steps may be guided via auxiliary heuristics or shaping rewards. DeepRetrieval (Jiang et al., 2025) exemplifies this paradigm by combining reasoning-style generation with PPO training on retrieval-based rewards. While effective, this approach highlights general challenges in RL-based query augmentation: reward sparsity, exploration vs. fluency tradeoffs, and computational cost for large LLMs.

Terminology. We use the term *RL* broadly to refer to optimization of query-rewriting policies via retrieval-based rewards. Specifically, our experiments employ *on-policy optimization* methods such as PPO (Schulman et al., 2017). While RL encompasses other paradigms (e.g., off-policy learning), we focus on on-policy methods here.

Algorithm Overview. The procedure for RL-based query augmentation can be summarized as Algorithm 1. This formulation allows training a query rewriter even when the retriever is a black-box system, such as an API-based service, and emphasizes the connection between query generation and retrieval performance.

4 Comparison

We perform a systematic empirical comparison of on-policy optimization for query augmentation — exemplified by DeepRetrieval (Jiang et al., 2025) — against a simple prompt-based method (SPQE, Simple Pseudo-document Query Expansion). Our goal is to quantify the relative benefits, limitations, and trade-offs of RL-based and zero-shot query augmentation approaches across different retrieval scenarios.

4.1 Methodology

We begin with a rigorous replication of Deep-Retrieval's experiments. Our setup mirrors the original work, including both sparse and dense retrieval models, where the retriever and corpus together form the "environment" in RL terminology. For sparse retrieval, we use BM25 (Robertson et al., 1995), and for dense retrieval, we use task-appropriate retrievers, as detailed in Section 4.2.

For the prompt-based method (SPQE), we instruct an instruction-following LLM to generate a hypothetical document d^H addressing the query q, and use the concatenated (q,d^H) as the augmented query. This zero-shot approach differs from Query2Doc (Wang et al., 2023), which relies on few-shot prompting. Compared to the similar zero-shot method HyDE (Gao et al., 2023), SPQE skips the followup steps of embedding multiple pseudo documents and aggregating their representations, and is designed to function as a simple, strong baseline for zero-shot query augmentation.

For RL-based query augmentation, we follow DeepRetrieval's reward design: the policy first generates reasoning steps, then output the enhanced query. Detailed reward formulations are provided in Appendix Section C.

4.2 Experimental Setup

Datasets, Retrieval and Evaluations. We follow the experimental setup of Jiang et al. (2025) to include evidence-seeking retrieval and ad hoc retrieval datasets. For evidence-seeking retrieval, we use Wikipedia-18³ as the retrieval corpus, and include three query collections: NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017) and SQuAD (Rajpurkar et al., 2016). For ad hoc retrieval, we use following datasets from BEIR (Thakur et al., 2021): FEVER (Thorne et al., 2018), HotpotQA (Yang et al., 2018), NF-Corpus (Boteva et al., 2016) and SciFact (Wadden et al., 2020), as these datasets include train set for on-policy optimization. We also include evaluation on MS MARCO's dev set with shallow annotation depth (Bajaj et al., 2016), alongside DL19 (Voorhees et al., 2020) and DL20 (Craswell et al., 2025) with in-depth annotations. Refer dataset statistics to Appendix Section A.

For evidence-seeking retrieval, we follow Ma et al. (2021) to measure Hit@20 (answer span hits).

³The 2018-12-20 snapshot of English Wikipedia, chunked into non-overlapping 100-word passages (Ma et al., 2021)

Table 1: Comparison of model performance on evidence-seeking retrieval and ad hoc retrieval datasets. Sparse columns use BM25 retriever; dense columns use E5-base-v2 for evidence-based retrieval datasets and Contriever-msmarco for ad hoc retrieval datasets. Best overall numbers are **bold**, best in-category numbers are <u>underlined</u>.

	Sparse BM25 Retriever			Dense Retriever				
Dataset	Base	+3B RL	+7B RL	SPQE	Base	+3B RL	+7B RL	SPQE
Evidence-seeking Datasets	Dense	= E5-base-	·v2)					
NQ	63.0	73.2	73.8	80.7	<u>86.1</u>	85.8	85.6	85.2
TriviaQA	76.4	79.8	80.9	<u>84.1</u>	83.5	82.8	83.7	<u>85.0</u>
SQuAD	71.1	72.1	<u>72.2</u>	69.7	70.2	<u>70.3</u>	70.2	68.8
Average Evidence-seeking	70.2	75.0	75.6	78.2	<u>79.9</u>	79.6	79.8	79.7
Ad Hoc Retrieval Datasets (Dense = Contriever-msmarco)								
MS MARCO	22.8	23.5	<u>24.1</u>	21.9	40.7	40.7	<u>40.8</u>	31.9
DL19	50.6	50.8	51.3	<u>60.0</u>	67.5	67.7	67.8	<u>70.1</u>
DL20	48.0	49.4	49.9	<u>60.1</u>	66.6	66.6	66.5	<u>69.3</u>
NFCorpus	32.5	34.3	<u>35.9</u>	33.9	32.8	33.5	34.2	<u>34.3</u>
HotpotQA	63.3	64.5	65.2	<u>66.5</u>	63.8	65.0	64.9	<u>68.2</u>
FEVER	65.1	72.3	74.7	<u>79.8</u>	75.8	<u>85.5</u>	84.8	83.9
SciFact	66.5	66.9	66.2	<u>71.4</u>	67.7	66.3	66.9	<u>68.9</u>
FiQA	23.6	<u>25.6</u>	24.7	19.5	32.9	33.1	<u>34.0</u>	26.5
Average Ad Hoc Retrieval	46.6	48.4	49.0	<u>51.6</u>	56.0	57.3	<u>57.5</u>	56.6

We use BM25 as the sparse retriever and E5-base-v2 (Wang et al., 2022) as the dense retriever.

For ad hoc retrieval, we use NDCG@10 as the evaluation metric following Thakur et al. (2021). For retrieval, we use their respective corpus, and use BM25 for sparse retrieval and Contrievermsmarco (Izacard et al., 2022) as the dense retriever. Note that Jiang et al. (2025) use different dense retrievers for different ad hoc retrieval datasets, but we control the dense retriever model. **Backbone Models.** We use Qwen2.5-{3B,7B}-Instruct (Qwen, 2024) as our initial policy. For implementation of SPQE, we use different LLMs. Our main experimental result is based on proprietary GPT-4o-mini (Menick et al., 2024) and we also report results with openweighted GPT-OSS-120B (Agarwal et al., 2025) and Qwen3-32B-Instruct (Yang et al., 2025).

Implementation Details. We use the original code-base from Jiang et al. (2025) and made several modifications to improve training throughput. Our retrieval environment is based on Pyserini (Lin et al., 2021) and Faiss (Johnson et al., 2019). We use veRL 0.1 (Sheng et al., 2025) as the RL framework and PPO (Schulman et al., 2017) as the specific on-policy RL algorithm. We did not include group-based on-policy RL algorithms such as GRPO (Shao et al., 2024; Guo et al., 2025), as we note a major bottleneck of our RL formulation lies in the agent-environment interaction part — in our case the retrieval step. Using group-based algo-

rithm increases the retrieval calls by the factor of group size, significantly slowing down the overall optimization process. We leave the investigation of other on-policy algorithms to future work. Without otherwise specified, we use learning rate 1e-6, global train batch size 128, actor and critic mini batch size 32, rollout temperature 0.6. All experiments are performed on 8x or 16x Nvidia A100 GPUs, each with 40GB VRAM.

4.3 Result and Analysis

We show the results of evidence-seeking retrieval and ad hoc retrieval in Table 1. The data demonstrates that both on-policy optimization and simple prompt-based expansion (SPQE) significantly improve retrieval performance over not using any query augmentation (the "Base" columns).

First, the results confirm the value of using onpolicy optimization. For ad hoc retrieval with a dense retriever, the +7B RL model achieves the highest average score of 57.5, a substantial improvement over the base retriever's score of 51.6. This gain is particularly noticeable on datasets like FEVER, where the score increases from 75.8 to 85.5. Similarly, with a sparse BM25 retriever on evidence-based datasets, the +7B RL model improves the average score from 70.2 to 75.6. These results show that directly training a model to generate queries that maximize retrieval scores is an effective strategy.

Furthermore, Table 1 shows that our simpler

Table 2: Comparison of model performance on tool retrieval datasets. Sparse use BM25 retriever; dense use Contriever-msmarco retriever. Best overall numbers on each dataset are **bold**, best in-category numbers are underlined.

NDCG@10				Recall@10				Completeness@10				
Dataset	Base	+3B RL	+7B RL	SPQE	Base	+3B RL	+7B RL	SPQE	Base	+3B RL	+7B RL	SPQE
Sparse BM25 Retrieve	er											
ToolRet-Web	24.8	28.8	26.7	27.4	33.6	38.3	35.9	38.0	26.0	29.7	27.9	30.2
ToolRet-Code	30.1	30.1	29.8	33.4	44.3	41.6	41.5	<u>46.0</u>	37.1	38.9	38.3	42.7
ToolRet-Customized	35.0	37.3	35.4	<u>39.6</u>	41.7	44.3	41.9	<u>49.4</u>	24.9	28.2	26.4	32.4
Average	30.0	32.0	30.7	33.4	39.9	41.4	39.8	44.4	29.4	32.3	30.8	35.1
Dense E5-base-v2 Rea	triever											
ToolRet-Web	31.3	31.6	31.1	34.2	41.1	41.5	40.9	<u>45.1</u>	31.6	32.0	31.4	<u>35.0</u>
ToolRet-Code	24.5	24.4	24.5	32.2	33.4	33.2	33.5	42.2	30.5	30.5	30.8	38.3
ToolRet-Customized	33.9	33.4	31.3	37.1	41.8	41.0	38.8	47.2	26.8	27.1	26.1	31.2
Average	29.9	29.8	29.0	34.5	38.8	38.6	37.8	44.8	29.6	29.9	29.4	34.8

prompt-based method, SPQE, is also highly effective and competitive. For sparse retrieval, SPQE consistently delivers the best performance, achieving the highest average scores for both evidence-based (78.2) and ad hoc (51.6) datasets. For dense retrieval, its performance is very close to the more complex RL method, even outperforming it on several individual ad hoc datasets like DL19 (70.1) and DL20 (69.3). Additionally, we note that SPQE is flexible about LLMs' choices, as Qwen-3-32B-Instruct and GPT-OSS-120B also achieve strong performance (details in Appendix Section B). We defer a more in-depth discussion about SPQE vs. on-policy optimization to Section 4.5.

4.4 Extending to Tool Retrieval

Retrieving useful tools from tool sets for LLM agents is becoming increasingly critical in real-world applications. However, the main challenge is that candidate tools are usually large-scale and many of them are similar in functionality (Patil et al., 2024; Qin et al., 2024; Qu et al., 2024; Shi et al., 2025). How query augmentation might enhance tool retrieval efficacy is rather underexplored due to the lack of large-scale training and evaluation datasets. We address the gap by studying prompting-based and RL-based query augmentation in the context of tool retrieval.

Datasets. We use a recently released tool retrieval dataset ToolRet (Shi et al., 2025), which is constructed from existing tool learning datasets. The train set contains over 200k instances, which allows us to study on-policy RL training at scale.

Training and Evaluation. We construct the trainset and train corpus from the released trainset. Note that here the train corpus (i.e., environment) is different from the corpus we run evaluation on, due

to the mismatch of candidate tools. We evaluate on three official splits: ToolRet-Web, ToolRet-Code, ToolRet-Customized. We train the same base policy as Section 4.2: Qwen2.5-{3B, 7B}-Instruct and adopt BM25 and E5-base-v2 as the retrievers. We report performance of SPQE as a simple prompting-based method compared to on-policy optimization. For evaluation, we focus on classical retrieval metrics NDCG@10 and Recall@10. We further include Completeness@10. The Completeness metric was proposed by Qu et al. (2024) to measure whether all the required tools have been included in the ranklist. For reward design, we directly optimize against the Completeness@10 metric. We leave further investigation of reward design to future works.

Results and Analysis. We show the results of tool retrieval in Table 2. Similar to our prior results, we observe that query augmentation methods can substantially improve performance over the baseline. On-policy optimization shows some benefits, particularly for the sparse retriever, but also reveals a notable disparity in its effectiveness between sparse and dense models.

For Completeness@10, the metric we optimized for, the +3B RL model improves the average score for the sparse BM25 retriever from a base of 29.4 to 32.3. This demonstrates that RL can successfully steer the model to improve a target metric. However, for the dense retriever, the same RL training shows no benefit, with average Completeness@10 scores (29.9 for +3B RL, 29.4 for +7B RL) failing to surpass the base score of 29.6. This disparity between sparse and dense retriever is critical. While RL provides a modest lift for BM25 on both NDCG@10 (from 30.0 to 32.0) and Completeness@10, it consistently degrades performance

for the dense Contriever across all metrics. For instance, the +7B RL model lowers the average NDCG@10 from 29.9 to 29.0. This observation suggests that the queries generated by the RL policy, while optimized for the Completeness reward, may be producing keyword-focused outputs that help term-matching systems like BM25 but result in less effective embeddings for semantic-focused dense retrievers.

In contrast, the simple prompting method SPQE proves to be the most effective and robust method across the board. It achieves the highest average scores for both NDCG@10 (33.4 for sparse, 34.5 for dense) and Completeness@10 (35.1 for sparse, 34.8 for dense). Furthermore, looking at Recall@10, HyDE, another generative approach, consistently outperforms all other methods. This indicates that for the task of tool retrieval, generating richer, document-like queries via simple prompting is a more reliable strategy for improving performance on both sparse and dense systems than the RL approach tested here.

4.5 Discussions

Our experiments show that both on-policy optimization and prompt-based expansion improve retrieval, but they trade off between targeted optimization and general applicability.

On-policy optimization excels at adapting a model to a specific retriever and domain. This is most evident on the specialized FiQA dataset (Table 1), where the RL-trained model outperforms prompting, likely by learning to generate queries with domain-specific financial terminology. However, this approach is computationally expensive, requires labeled data for rewards, and may not always achieve performance improvement. For instance, our ToolRetrieval policy underperforms no query augmentation setting with dense retrievers.

In contrast, prompt-based expansion (e.g., SPQE) is a robust and resource-efficient baseline that performs well across nearly all settings. We hypothesize that its effectiveness stems from three factors. First, it leverages the LLM's internal knowledge to expand queries with relevant terms, mitigating the vocabulary mismatch problem for sparse retrievers like BM25. Second, it transforms asymmetric query-document retrieval into a symmetric document-document task, which better aligns with the training objectives of many dense retrievers and boosts their performance. Third, unlike RL methods that rewrite the original query, prompt-

based expansion produces a pseudo-document, offering richer context for the retriever to match against. We hypothesize this additional signal makes pseudo-documents especially effective on existing IR benchmarks.

5 On-policy Pseudo-document Query Expansion

Our experiments in Section 4.5 highlighted a key difference between two paradigms: on-policy RL methods rewrite the original query $q \rightarrow q'$, while prompt-based expansion methods like SPQE generate a pseudo-document d^H to provide richer, more structured information to the retriever. This observation motivates a natural follow-up question: can we combine the strengths of both approaches to further improve query augmentation? In this section, we introduce a straightforward modification to achieve this synergy.

5.1 Method

We make a simple modification to the on-policy optimization RL process. Instead of instructing the policy model to rewrite the query $q \rightarrow q'$, we instruct the model to write a hypothetical document d^H , then directly output the concatenated (q,d^H) (prompt template in Appendix Section E). Then this concatenated string is used to retrieve relevant documents from the environment, similar to SPQE, and the reward is computed by the retrieval metrics like NDCG. In this way, we merge the SPQE into the on-policy optimization process. We refer to this method as On-policy Pseudo-document Query Expansion (OPQE).

5.2 Experiment, Result and Analysis

Experimental Setup. We experiment on evidence-seeking retrieval and ad hoc retrieval datasets. Our implementation is the same as Section 4.2 except for the prompt template to instruct the policy model to write a pseudo document.

Main Results. We report the results of our hybrid approach in Table 3. The key finding is that our proposed method, OPQE, successfully integrates the strengths of both prompting and on-policy optimization, achieving the best overall performance, especially in the dense retrieval setting.

For dense retriever, the benefits of our hybrid approach are most apparent. On the ad hoc retrieval results, the OPQE-7B model achieves a new state-of-the-art average score of 58.1. This surpasses

Table 3: Comparison of model performance on evidence-based retrieval and ad hoc retrieval datasets. We apply OPQE on 3B and 7B models. Sparse columns use BM25 retriever; dense columns use E5-base-v2 for evidence-based retrieval datasets and Contriever-msmarco for ad hoc retrieval datasets. Best overall numbers are **bold**, best in-category numbers are underlined.

	Sparse BM25 Retriever					Dense Retriever						
Dataset	Base	+3B	+7B	SPQE	OPQE-3B	OPQE-7B	Base	+3B	+7B	SPQE	OPQE-3B	OPQE-7B
Evidence-base	d Retrie	val Da	tasets (Dense =	E5-base-v2)							
NQ	63.0	73.2	73.8	80.7	74.7	75.2	<u>86.1</u>	85.8	85.6	85.2	85.9	85.9
TriviaQA	76.4	79.8	80.9	84.1	80.6	81.0	83.5	82.8	83.7	<u>85.0</u>	83.3	83.3
SQuAD	71.1	72.1	72.2	69.7	<u>73.3</u>	73.2	70.2	70.3	70.2	68.8	<u>71.4</u>	70.1
Average	70.2	75.0	75.6	78.2	76.2	76.5	79.9	79.6	79.8	79.7	80.2	79.8
Ad Hoc Retrie	val Date	asets (L	ense =	Contrie	ver-msmarco))						
MS MARCO	22.8	23.5	24.1	21.9	23.8	<u>25.1</u>	40.7	40.7	<u>40.8</u>	31.9	40.5	37.5
DL19	50.6	50.8	51.3	60.0	53.9	56.6	67.5	67.7	67.8	<u>70.1</u>	67.8	69.5
DL20	48.0	49.4	49.9	60.1	51.3	55.4	66.6	66.6	66.5	69.3	66.8	<u>70.5</u>
NFCorpus	32.5	34.3	<u>35.9</u>	33.9	<u>35.9</u>	35.7	32.8	33.5	34.2	34.3	33.5	<u>34.6</u>
HotpotQA	63.3	64.5	65.2	<u>66.5</u>	63.3	65.4	63.8	65.0	64.9	<u>68.2</u>	63.7	64.9
FEVER	65.1	72.3	74.7	79.8	81.1	80.7	75.8	<u>85.5</u>	84.8	83.9	84.9	85.4
SciFact	66.5	66.9	66.2	71.4	67.9	68.6	67.7	66.3	66.9	<u>68.9</u>	67.9	68.4
FIQA	23.6	<u>25.6</u>	24.7	19.5	25.0	24.9	32.9	33.1	34.0	26.5	34.1	<u>34.4</u>
Average	46.6	48.4	49.0	<u>51.6</u>	50.3	<u>51.6</u>	56.0	57.3	57.5	56.6	57.4	<u>58.1</u>

the previous best from the standard +7B RL model (57.5) and the prompt-based SPQE (56.6). This improvement is consistent across multiple datasets, with OPQE setting new top scores on benchmarks like DL20 (70.5). Similarly, on the evidence-based retrieval datasets, the OPQE-3B model pushes the average performance to 80.2, edging out all other methods on what is already a very strong baseline. This shows that by using RL to fine-tune the generation of pseudo-documents, our model learns to create query representations that are more effective for dense retrievers than either method could achieve alone.

For the sparse BM25 retriever, the results demonstrate that OPQE is highly competitive. While the zero-shot SPQE remains the top performer on evidence-based datasets (78.2), our OPQE-7B model closes the gap significantly, outperforming the standard +7B RL model by a clear margin (76.5 vs. 75.6). On the Ad Hoc Retrieval Datasets, the OPQE-7B model matches the top performance of SPQE, both achieving an average score of 51.6. In summary, by framing on-policy optimization as generating a pseudo-document rather than a rewritten query, our OPQE method combines the robust generative structure of prompting with the targeted fine-tuning of reinforcement learning. This synergy leads to significantly improved performance, validating our approach as a superior method for query augmentation.

Training Trajectories of OPQE. To better under-

stand the performance gains of our hybrid OPQE method, we analyze its reward trajectories during training compared to the standard RL approach in Fig. 1. These plots provide evidence for our hypotheses about the effectiveness of incorporating a pseudo-document generation structure.

First, the trajectories validate our hypothesis that this structure effectively leverages the LLM's internal knowledge, particularly for sparse retrievers. In Fig. 1a and Fig. 1c (FEVER and TriviaQA with BM25), the OPQE policy (blue line) starts with a significantly higher average reward than the standard RL policy (red line). This demonstrates that the initial, zero-shot pseudo-documents are already high-quality query expansions that provide rich lexical signals for the term-based BM25 retriever. OPQE essentially begins with a "warm start" by using the LLM's knowledge, while the standard RL method must learn to generate effective query terms from a lower baseline.

Second, the plots support the idea that OPQE transforms retrieval into a more effective symmetric task. While the initial reward gap is less pronounced for dense retrievers in Fig. 1b and Fig. 1d, OPQE's reward trajectory is consistently stable and slightly higher. By instructing the model to generate a full document, we provide a structured learning process that is well-aligned with how dense retrievers compare semantic content. This superior framing allows the RL agent to optimize more effectively, leading to the better final performance

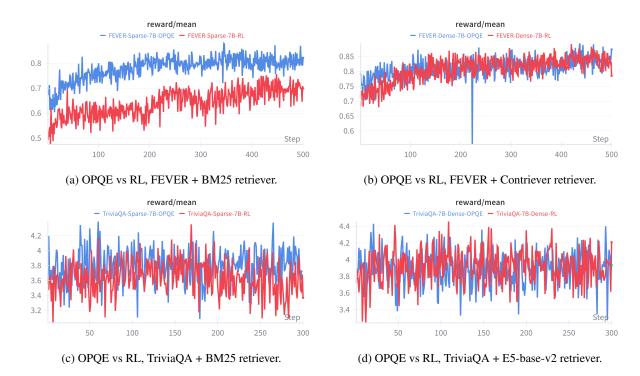


Figure 1: Reward curves of OPQE vs. RL on FEVER and TriviaQA datasets across different retrievers.

observed in our results tables.

In essence, the training dynamics demonstrate OPQE's synergy: it leverages a knowledge-rich starting point from prompting and then uses on-policy optimization to fine-tune this already effective generation process, leading to higher and more stable rewards throughout training.

6 Conclusion and Future Works

In this paper, we systematically evaluated and compared two LLM-based query augmentation paradigms: simple prompting-based query augmentation and RL-based query augmentation, across a variety of IR tasks. Our results suggest that promptbased methods with advanced LLMs can perform on par with, or even surpass, RL-based methods in many scenarios, while RL-based approaches are particularly effective at enabling smaller LLMs to excel at this task. Furthermore, a hybrid approach, where pseudo-document generation prompts from prompt-based methods are used to provide stronger initialization for RL-based training, achieved the best overall performance. These findings suggest that RL-based query augmentation frameworks can be further optimized through improved task reformulation.

While our study highlights the strengths and trade-offs of prompting-based and RL-based query augmentation, several avenues remain open for future exploration. First, developing more lightweight yet effective prompting strategies for zero-shot query augmentation could reduce reliance on large LLMs and broaden applicability in resource-constrained settings. Second, refining RL task formulations, for example, by designing more nuanced reward functions, may help stabilize training and yield stronger performance. Finally, exploring query augmentation prompt design and RL training beyond the classic single-round retrieval paradigm, such as in multi-hop or deep research settings (Chen et al., 2025), remains an underexplored but important direction.

Limitations and Ethical Risks

Given the limited bandwidth, our on-policy RL experiments are based on PPO algorithm and use 3B and 7B-scale models. Our observations and conclusions may be subject to change with more performant base models and different on-policy algorithms. This paper focuses on empirical experiments on public benchmarks. We believe this paper do not incur potential risks.

References

Kenya Abe, Kunihiro Takeoka, Makoto P Kato, and Masafumi Oyamada. 2025. Llm-based query expansion fails for unfamiliar and ambiguous queries. In *Proceedings of the 48th International ACM SIGIR*

- Conference on Research and Development in Information Retrieval, pages 3035–3039.
- Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, and 1 others. 2025. gpt-oss-120b & gpt-oss-20b model card. arXiv preprint arXiv:2508.10925.
- Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, and 1 others. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.
- Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. 2016. A full-text learning to rank dataset for medical information retrieval. In *European Conference on Information Retrieval*, pages 716–722. Springer.
- Xin Cao, Gao Cong, Bin Cui, Christian Søndergaard Jensen, and Ce Zhang. 2009. The use of categorization information in language models for question retrieval. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 265–274.
- Chia-Yuan Chang, Zhimeng Jiang, Vineeth Rakesh, Menghai Pan, Chin-Chia Michael Yeh, Guanchu Wang, Mingzhi Hu, Zhichao Xu, Yan Zheng, Mahashweta Das, and Na Zou. 2025. MAIN-RAG: Multi-Agent Filtering Retrieval-Augmented Generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2607–2622, Vienna, Austria. Association for Computational Linguistics. 10.18653/v1/2025.acl-long.131.
- Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, Sahel Sharifymoghaddam, Yanxi Li, Haoran Hong, Xinyu Shi, Xuye Liu, Nandan Thakur, Crystina Zhang, Luyu Gao, Wenhu Chen, and Jimmy Lin. 2025. Browsecomp-plus: A more fair and transparent evaluation benchmark of deep-research agent. *Preprint*, arXiv:2508.06600.
- João Coelho, Bruno Martins, João Magalhães, and Chenyan Xiong. 2025. Aligning web query generation with ranking objectives via direct preference optimization. In Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 2982–2986.
- Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, Jimmy Lin, Ellen M Voorhees, and Ian Soboroff. 2025. Overview of the trec 2022 deep learning track. *arXiv preprint arXiv:2507.10865*.
- Guglielmo Faggioli, Thibault Formal, Stefano Marchesin, Stéphane Clinchant, Nicola Ferro, and Benjamin Piwowarski. 2023. Query performance prediction for neural ir: Are we there yet? In *European*

- Conference on Information Retrieval, pages 232–248. Springer.
- Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777, Toronto, Canada. Association for Computational Linguistics.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.
- Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.
- Pengcheng Jiang, Jiacheng Lin, Lang Cao, Runchu Tian, SeongKu Kang, Zifeng Wang, Jimeng Sun, and Jiawei Han. 2025. Deepretrieval: Hacking real search engines and retrievers with large language models via reinforcement learning. *Preprint*, arXiv:2503.00223.
- Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.
- Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.
- 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*, 7:452–466.
- Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, and 1 others. 2024. Tulu 3: Pushing frontiers in open language model post-training. *arXiv* preprint arXiv:2411.15124.
- Yibin Lei, Yu Cao, Tianyi Zhou, Tao Shen, and Andrew Yates. 2024. Corpus-steered query expansion with large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 393–401, St. Julian's, Malta. Association for Computational Linguistics.

- Yibin Lei, Tao Shen, and Andrew Yates. 2025. Thinkqe: Query expansion via an evolving thinking process. *Preprint*, arXiv:2506.09260.
- Minghan Li, Xinxuan Lv, Junjie Zou, Tongna Chen, Chao Zhang, Suchao An, Ercong Nie, and Guodong Zhou. 2025. Query expansion in the age of pretrained and large language models: A comprehensive survey. *Preprint*, arXiv:2509.07794.
- Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: An easy-to-use python toolkit to support replicable ir research with sparse and dense representations. *arXiv preprint arXiv:2102.10073*.
- Hang Liu, Meng Chen, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2021. Conversational query rewriting with self-supervised learning. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7628–7632. IEEE.
- Shu Liu, Sumanth Hegde, Shiyi Cao, Alan Zhu, Dacheng Li, Tyler Griggs, Eric Tang, Akshay Malik, Kourosh Hakhamaneshi, Richard Liaw, Philipp Moritz, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. 2025. Skyrl-sql: Matching gpt-4o and o4-mini on text2sql with multi-turn rl.
- Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, Singapore. Association for Computational Linguistics.
- Xueguang Ma, Kai Sun, Ronak Pradeep, and Jimmy Lin. 2021. A replication study of dense passage retriever. *arXiv preprint arXiv:2104.05740*.
- Chuan Meng, Negar Arabzadeh, Mohammad Aliannejadi, and Maarten De Rijke. 2023. Query performance prediction: From ad-hoc to conversational search. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2583–2593.
- Jacob Menick, Kevin Lu, Shengjia Zhao, E Wallace, H Ren, H Hu, N Stathas, and F Petroski Such. 2024. Gpt-40 mini: advancing cost-efficient intelligence. *Open AI: San Francisco, CA, USA*.
- 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)*, pages 4998–5012, Toronto, Canada. Association for Computational Linguistics.
- Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. 2024. Gorilla: Large language model connected with massive apis. *Advances in Neural Information Processing Systems*, 37:126544–126565.

- Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Xuanhe Zhou, Yufei Huang, Chaojun Xiao, and 1 others. 2024. Tool learning with foundation models. *ACM Computing Surveys*, 57(4):1–40.
- Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen. 2024. Towards completeness-oriented tool retrieval for large language models. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 1930–1940.
- Qwen. 2024. Qwen2 technical report. arXiv preprint arXiv:2407.10671, 2.
- 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. *Journal of machine learning research*, 21(140):1–67.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.
- Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, and 1 others. 1995. *Okapi at TREC-3*. British Library Research and Development Department.
- J. J. Rocchio. 1971. Relevance feedback in information retrieval. In G. Salton, editor, *The Smart retrieval sys*tem - experiments in automatic document processing, pages 313–323. Englewood Cliffs, NJ: Prentice-Hall.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint* arXiv:1707.06347.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.
- Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pages 1279–1297.
- Zhengliang Shi, Yuhan Wang, Lingyong Yan, Pengjie Ren, Shuaiqiang Wang, Dawei Yin, and Zhaochun Ren. 2025. Retrieval models aren't tool-savvy: Benchmarking tool retrieval for large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 24497–24524, Vienna, Austria. Association for Computational Linguistics.

- Richard S Sutton, Andrew G Barto, and 1 others. 1998. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.
- Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.
- James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.
- Ellen M. Voorhees. 1994. Query expansion using lexical-semantic relations. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, page 61–69, Berlin, Heidelberg. Springer-Verlag.
- Ellen M Voorhees, Nick Craswell, Bhaskar Mitra, Daniel Campos, and Emine Yilmaz. 2020. Overview of the trec 2019 deep learning track.
- David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550, Online. Association for Computational Linguistics.
- Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv* preprint *arXiv*:2212.03533.
- Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query expansion with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9423, Singapore. Association for Computational Linguistics.
- Xiao Wang, Craig Macdonald, and Iadh Ounis. 2020. Deep reinforced query reformulation for information retrieval. *arXiv* preprint arXiv:2007.07987.
- 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*, pages 10000–10014.

- Zhichao Xu, Ashim Gupta, Tao Li, Oliver Bentham, and Vivek Srikumar. 2024. Beyond Perplexity: Multidimensional Safety Evaluation of LLM Compression. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15359–15396, Miami, Florida, USA. Association for Computational Linguistics. 10.18653/v1/2024.findings-emnlp.901.
- Zhichao Xu, Fengran Mo, Zhiqi Huang, Crystina Zhang, Puxuan Yu, Bei Wang, Jimmy Lin, and Vivek Srikumar. 2025. A Survey of Model Architectures in Information Retrieval. *arXiv preprint arXiv:2502.14822*.
- Zhichao Xu, Zhiqi Huang, Shengyao Zhuang, and Vivek Srikumar. 2025. Distillation versus Contrastive Learning: How to Train Your Rerankers. *arXiv* preprint arXiv:2507.08336.
- Zhichao Xu, Jinghua Yan, Ashim Gupta, and Vivek Srikumar. 2025. State Space Models are Strong Text Rerankers. In *Proceedings of the 10th Workshop on Representation Learning for NLP (RepL4NLP-2025)*, pages 152–169, Albuquerque, NM. Association for Computational Linguistics. 10.18653/v1/2025.repl4nlp-1.12.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.
- 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*, pages 5985–6006.
- Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Fewshot generative conversational query rewriting. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1933–1936, New York, NY, USA. Association for Computing Machinery.
- Chengxiang Zhai and John Lafferty. 2001. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 403–410.

Changtai Zhu, Siyin Wang, Ruijun Feng, Kai Song, and Xipeng Qiu. 2025. Convsearch-r1: Enhancing query reformulation for conversational search with reasoning via reinforcement learning. *arXiv* preprint *arXiv*:2505.15776.

A Dataset Statistics and Licenses

We report statistics of each dataset in Table 4. Most datasets are released under permissive open licenses such as CC BY-SA (Natural Questions, SQuAD, HotpotQA, FEVER), CC BY (MS MARCO, DL19, DL20, SciFact), and Apache 2.0 (TriviaQA), while NFCorpus and FiQA restrict usage to academic or non-commercial purposes. SciFact uses ODC-By 1.0 for its corpus. ToolRet is a composite benchmark repurposed from multiple sources with mixed licensing; details are provided in Appendix B and Table 6 of Shi et al. (2025).

Table 4: Detailed dataset statistics.

Dataset	# Train	# Val	# Test	Corpus Size					
Evidence-based Retrieval Datasets									
Natural Questions	79,168	8,757	3,610	21M					
TriviaQA	78,785	8,837	11,313	21M					
SQuAD	87,599	_	10,570	21M					
Ad Hoc Retrieval Datasets									
MS-MARCO	502,939	-	5,360	8.84M					
DL19	_	_	43	8.84M					
DL20	-	-	54	8.84M					
NFCorpus	2,590	647	323	3.6K					
HotpotQA	85,000	12,852	7,405	5.23M					
FEVER	109,810	13,332	6,666	5.42M					
SciFact	818	440	339	5K					
FIQA	5,500	500	648	57K					
Tool Retrieval Datase	Tool Retrieval Dataset								
ToolRet-Train	208,826	-	_	31,123					
ToolRet-Web	_	-	4,916	36,978					
ToolRet-Code	-	-	950	3,794					
ToolRet-Customized	_	_	1,749	2,443					

B Results of SPQE with Different LLMs

Table 5: Performance (Hit@20) of SPQE with different LLMs on evidence-seeking datasets.

	ъ	CDE 4	0 2.220	CDT OGG 120D					
	Base	GPT-4o-mini	Qwen3-32B	GPT-OSS-120B					
Sparse BM25 Retriever									
NQ	63.0	80.7	77.6	81.1					
TriviaQA	76.4	84.1	81.6	84.5					
Squad	71.1	69.7	70.3	74.3					
Average	70.2	78.2	76.5	80.0					
Dense E5-	base-v2	Retriever							
NQ	86.1	85.2	83.9	85.1					
TriviaQA	83.5	85.0	83.2	85.5					
Squad	70.2	68.8	68.5	68.5					
Average	79.9	79.7	78.5	79.7					

We report performance comparison of SPQE with different LLMs in Table 5, Table 6 and Table 7, respectively.

C Reward Formulations

The reward function r(q') we used in DeepRetrieval's RL training consists of format reward r_{format} and retrieval reward $r_{\text{retrieval}}$, where r_{format} evaluates whether the trained policy follows the required response format and $r_{\text{retrieval}}$ evaluates the retrieval performance of the augmented query. Theformulation is $r(q') = r_{\text{format}}(q') \cdot r_{\text{retrieval}}(q')$ where the format reward is an indicator function:

$$r_{\text{format}}(q') = \begin{cases} 0, & q' \text{ violates the required format} \\ 1, & q' \text{ follows the required format} \end{cases}$$

For retrieval reward, we report the specific instantiation for each task in Table 8.

D Completeness Metric

Let G(q) be the set of ground-truth documents for query q, and $R_k(q)$ be the set of top-k retrieved documents for query q. Then completeness@k is defined as:

Completeness@
$$k(q) = \begin{cases} 1, & \text{if } G(q) \subseteq R_k(q), \\ 0, & \text{otherwise.} \end{cases}$$

Averaged over a query set Q, we have

$$\operatorname{Completeness}@k = \frac{1}{|Q|} \sum_{q \in Q} \mathbf{1}[\,G(q) \subseteq R_k(q)\,]\,.$$

E Prompt Templates

We show RL prompt templates in Fig. 2 and Fig. 3, SPQE prompt template in Fig. 4 and OPQE prompt template in Fig. 5.

Table 6: Performance comparison of SPQE with different LLMs on ad hoc retrieval datasets.

	Base		GPT-4	o-mini	Qwer	13-32B	GPT-OS	GPT-OSS-120B	
	NDCG@10	Recall@100	NDCG@10	Recall@100	NDCG@10	Recall@100	NDCG@10	Recall@100	
Sparse BM25	Retriever								
MSMARCO	22.8	65.8	21.9	67.5	20.3	66.6	21.9	67.3	
DL19	50.6	49.7	60.0	58.8	60.9	59.9	58.0	57.6	
DL20	48.0	56.7	60.1	68.5	59.3	66.9	57.0	65.0	
NFCorpus	32.2	24.6	33.9	30.8	34.4	31.0	33.0	32.1	
HotpotQA	63.3	79.6	66.5	84.8	61.3	81.8	69.3	85.9	
FEVER	65.1	91.8	79.8	95.3	77.9	94.8	78.3	95.7	
SciFact	67.9	92.5	71.4	95.9	71.2	95.9	73.8	96.4	
FiQA	23.6	53.9	19.5	50.0	21.3	54.6	19.2	49.4	
Avg	46.7	64.3	51.6	68.9	50.8	68.9	51.3	68.7	
Dense Contri	ever Retriever								
MSMARCO	40.7	89.1	31.9	81.5	31.7	81.6	28.2	78.2	
DL19	67.5	61.3	70.1	65.6	69.6	65.7	64.6	61.2	
DL20	66.6	71.5	69.3	74.0	66.8	72.8	57.3	68.3	
NFCorpus	32.8	30.1	34.3	33.5	35.0	32.7	29.1	28.9	
HotpotQA	63.8	77.7	68.2	83.7	62.8	79.9	66.9	82.7	
FEVER	75.8	94.9	83.9	96.0	83.2	95.8	80.6	95.0	
SciFact	67.7	94.7	68.9	95.7	69.5	96.3	67.3	92.9	
FiQA	32.9	65.6	26.5	60.9	30.7	66.5	20.4	53.5	
Avg	56.0	73.1	56.6	73.9	56.2	73.9	51.8	70.1	

Table 7: Performance comparison of SPQE with different LLMs on ToolRetrieval datasets.

	Base		GF	GPT-4o-mini		wen3-32B	GPT-OSS-120B	
	Recall@10	Completeness@10	Recall@10	Completeness@10	Recall@10	Completeness@10	Recall@10	Completeness@10
Sparse BM25 Retrieve	er							
ToolRet-Web	33.6	26.0	38.0	30.2	38.1	29.8	39.0	30.5
ToolRet-Code	44.3	37.1	46.0	42.7	47.4	44.4	46.8	43.0
ToolRet-Customized	41.7	24.9	49.4	32.4	46.7	30.1	47.1	31.4
Average	39.9	29.4	44.4	35.1	44.1	34.8	44.3	34.9
Dense Contriever Ret	riever							
ToolRet-Web	41.1	31.6	45.1	35.0	45.7	35.6	44.6	34.6
ToolRet-Code	33.4	30.5	42.2	38.3	44.9	41.9	43.1	40.4
ToolRet-Customized	41.8	26.8	47.2	31.2	45.6	29.6	45.2	29.4
Average	38.8	29.6	44.8	34.8	45.4	35.7	44.3	34.8

Table 8: Retrieval reward function $r_{\text{retrieval}}(q')$.

Task	Retrieval Reward
Evidence-seeking Retrieval	5.0 if rank ≤ 5 4.0 if rank ≤ 20 2.0 if rank ≤ 50 1.0 if rank ≤ 100 0.5 if rank ≤ 1000 0.1 if rank ≤ 3000 -2.5 otherwise
Ad hoc Retrieval	NDCG@10 score
Tool Retrieval	Completeness@10 score

```
RL (DeepRetrieval) Prompt Template
<|im_start|>system
You are a helpful assistant. You first thinks about the reasoning process in the mind and then
   provides the user with the answer.
<|im_end|>
<|im_start|>user
You are a query rewriting expert. Your task is to create query terms for user query to find
    relevant literature in a Wikipedia corpus using BM25.
Show your reasoning inside the <think> </think> tags.
Your final output must be inside the <answer> </answer> tags, and strictly in JSON format.
For example:
<think>
[reasoning process]
</think>
<answer>
    "query": "...."
</answer>
Note: The query should use Boolean operators (AND, OR) and parentheses for grouping terms
    appropriately.
Here is the user query:
[USER_QUERY]
Assistant: Let me rewrite the query with reasoning.
<think>
```

Figure 2: The RL (DeepRetrieval) prompt template for NQ dataset — sparse retrieval. Notice the boolean operators.

```
RL (DeepRetrieval) Prompt Template
<|im_start|>system
You are a helpful assistant. You first thinks about the reasoning process in the mind and then
    provides the user with the answer.
<|im_end|>
<|im_start|>user
You are an expert in generating queries for dense retrieval. Given a web search query, your task
    is to retain the original query while expanding it with additional semantically relevant
    information, to retrieve relevant passages that answer the query. If no useful expansion is
    needed, return the original query as is.
Show your reasoning inside the <think> </think> tags.
Your final output must be inside the <answer> </answer> tags, and strictly in JSON format.
For example:
<think>
[reasoning process]
</think>
<answer>
    "query": "...."
</answer>
Here is the user query:
[USER_QUERY]
Assistant: Let me rewrite the query with reasoning.
<think>
```

Figure 3: The RL (DeepRetrieval) prompt template for NQ dataset—dense retrieval.

```
SPQE Prompt Template

Write a short Wikipedia passage to answer this question.

Question: [USER_QUERY]

Passage:
```

Figure 4: SPQE prompt template for NQ dataset. We use same prompt template for sparse and dense retrieval.

```
OPQE Prompt Template
<|im_start|>system
You are a helpful assistant. First think through the reasoning internally, then provide the
    answer as instructed.
<|im_end|>
<|im_start|>user
You are a helpful assistant. Given a search query, write a concise Wikipedia-style passage (a few
    sentences) that answers the query. This passage will then be concatenated with the original
    query to form an enhanced query.
Show your reasoning inside the <think> </think> tags.
Your final output must be inside the <answer> </answer> tags, and strictly in JSON format.
For example:
<think>
[reasoning process]
</think>
<answer>
    "query": "original query. concise passage"
</answer>
Note: Make sure you write the passage instead of directly answering the question.
Here is the user query:
[USER_QUERY]
Assistant: Let me augment the query with reasoning.
<think>
```

Figure 5: The OPQE prompt template for NQ dataset. We use same prompt template for sparse and dense retrieval.