QUESTER: Query Specification for Generative Keyword-Based Retrieval

Arthur Satouf[1,2,3,4], Yuxuan Zong[4], Habiboulaye Amadou-Boubacar[3], Pablo Piantanida[1,2], Benjamin Piwowarski[4]

[1] MILA – Quebec AI Institute & ILLS, Canada
[2]Université Paris-Saclay & CentraleSupélec & CNRS, France
[3]Air Liquide, France
[4]Sorbonne Université & ISIR & CNRS, France

arthur.satouf(at)gmail.com, zong@isir.upmc.fr, habiboulaye.amadou-boubacar@airliquide.com,
pablo.piantanida@cnrs.fr, benjamin.piwowarski@cnrs.fr

Abstract

Generative retrieval (GR) differs from the traditional index–then–retrieve pipeline by storing relevance in model parameters and generating retrieval cues directly from the query, but it can be brittle out of domain and expensive to scale. We introduce QUESTER (**QUE**ry **S**pecifica**T**ion g**E**nerative Keyword-Based **R**etrieval), which bridges GR and query reformulation by learning to generate explicit keyword-based search specifications. Given a user query, a lightweight LLM produces a keyword query that is executed by a standard retriever (BM25), combining the generalization benefits of generative query rewriting with the efficiency and scalability of lexical indexing. We train the rewriting policy with reinforcement learning techniques. Across in- and out-of-domain evaluations, QueStER consistently improves over BM25 and is competitive with neural IR baselines, while maintaining strong efficiency.[1]

1 Introduction

Information Retrieval (IR) models aim to retrieve relevant documents from a large database that satisfy a user information need. They are a key component of search engines and of Retrieval-Augmented Generation (RAG) models (Lewis et al., 2020). For a long time, IR BoW (bag-of-words) models, such as BM25 (Robertson et al., 1994), have been based on term-specific statistics, allowing fast retrieval through the use of efficient index structures such as Block-Max WAND (Grand et al., 2020). However, bag-of-words models suffer from the vocabulary mismatch problem, whereby the user query might contain keywords that do not appear in relevant documents.

A standard way to solve this problem is to rely on query rewriting techniques. The first works (Lavrenko and Croft, 2001; Abdul-Jaleel et al., 2004) add keywords extracted from the documents retrieved by the original query. This might lead to query drift, especially if the retrieved documents are not related to the user information need. More recently, (Jagerman et al., 2023; Alaofi et al., 2023) leveraged LLMs to improve the quality of the rewritten query. While improving compared to previous works, these strategies degrade when using a small LLM (e.g. smaller than 4B of parameters). When using large LLMs (with complex prompts), and moreover sampling several times (Zhang et al., 2024a), which decrease the efficiency. Exploring alternatives that balance effectiveness *and* efficiency is important.

Nowadays, a common alternative to increase the effectiveness of IR models is to build models on top of Transformer (Vaswani et al., 2017) architectures. Among the different models, dual encoders (dense and sparse), as well as late-interaction models such as ColBERT (Santhanam et al., 2021), are the most effective and efficient first stage neural rankers. Other architectures such as cross-encoders (Nogueira and Cho, 2019) are only used for re-ranking, and are not the focus of this paper. In addition, learned sparse lexical retrievers (e.g., DeepImpact, RRA) (Basnet et al., 2024; Satouf et al., 2025) predict context-dependent term weights and sometimes lexical expansion, mitigating vocabulary mismatch while retaining inverted-index retrieval.

However, neural IR models have shortcomings since using a neural (first-stage) IR model efficiently requires indices (either dense or sparse, depending on the type of model). This is costly for two reasons. First, the space taken by those indices is much larger than that for BoW models (e.g. a dense index on MS MARCO with 8.8M documents using FP16 for 768-dimensional vectors requires 13 GiB, while the BM25 index only takes 0.67 GiB (Park et al., 2025)). Second, when the model is re-trained, the index must be rebuilt, which is not

[1]Code on Github.

Findings of the Association for Computational Linguistics: EACL 2026, pages 5957–5968
March 24-29, 2026

practical with large collections such as that of the OpenSearch Euro project (Granitzer et al., 2024).

To tackle this issue, Generative Retrieval (GR) models attempt to internalize the index in the parameters (Tay et al., 2022), which avoids entirely the construction of an index. During inference, these models directly predict a document identifier conditioned on the user query. Some GR models rely on arbitrary document identifiers (Tay et al., 2022; Wang et al., 2022), but these models do not generalize well, especially for large collections. Another research direction is to use document meta-data (Zhou et al., 2022; Tang et al., 2023; Zhang et al., 2024b) – such as URL, title, or some keywords from the document – as identifiers. While these approaches generalize better, they are still limited by the metadata they use.

In this paper, we generalize over meta-data based GR approaches, and posit that instead of trying to map queries to some document meta-data, *generative models should generate search specifications*. A simple specification is a set of keywords that can be processed by e.g. a BM25 model, which resembles in this case query rewriting techniques. A more complex system would rely on structured specifications already present in major search libraries such as Lucene.[2]

The advantages are threefold. First, generating search specifications means that we can rely on established search technologies for which retrieval has been thoroughly optimized and for which query languages can be used to express complex user information needs. This leads to potentially more effective and efficient models than relying on document meta-data alone. Second, when the underlying neural network evolves, the index does not need to be rebuilt. Third, a user can analyze the query, which is a major concern in explainable AI-sensitive applications (e.g., law, patents, and medicine).

In this work, we propose a model named QUESTER (for Search Specification Generative Retrieval). QUESTER is based on an LLM whose parameters are optimized, with GRPO (Shao et al., 2024), to generate keyword queries for a search engine. In this work, we restrict the search engine to be a keyword-based BM25. Through our experiments, we show that QUESTER is both effective (+4.0 nDCG@10 in-domain and +5.3 out-of-domain, compared to BM25) *and* efficient ($\approx$ 28 ms/query for BM25). QUESTER reaches the performance of prompt-based methods that depend on much larger LLMs, despite using only a 4B backbone LLM.

[2]An example of a structured query specification can be found, see (Lucene docs).

2 Related Work

Information Retrieval Information Retrieval focuses on efficiently and effectively retrieving relevant documents that satisfy users information needs. Although current neural approaches such as DPR (Karpukhin et al., 2020) or RepLLaMA (Ma et al., 2024) dominate the leaderboards, traditional sparse models such as BM25 (Robertson et al., 1994) are still widely considered by the community due to their excellent tradeoff between efficiency and effectiveness.

Generative Retrieval Models retrieve documents by directly generating their identifiers (DocIDs) using language models, bypassing the indexing stage. A first series of work (Tay et al., 2022; Wang et al., 2022; Sun et al., 2023; Yang et al., 2023; Lee et al., 2023) proposes to associate with a document an arbitrary DocID which comes from a (hierarchical) clustering process or is learned. In both cases, these models have limited power of generalization and their effectiveness decreases with the number of documents in the collection. An alternative is to use "natural" DocIDs, such as document titles (De Cao et al., 2020; Chen et al., 2022a,b), URLs (Zhou et al., 2022), N-grams (Bevilacqua et al., 2022) or document summaries (Tang et al., 2023). Although this offers good interpretability, meta-data are not a perfect representation of document semantic content, and the performance of such models still lags compared to other neural approaches on large-scale datasets (Pradeep et al., 2023).

Query Rewriting Query rewriting (Carpineto and Romano, 2012) is a common solution in IR to improve query precision and expressiveness. Statistical models such as BM25 (Robertson et al., 1994) use pseudo-relevance feedback methods (Lavrenko and Croft, 2001; Abdul-Jaleel et al., 2004) that rely on heuristics and/or statistical analysis of the initial query with its retrieved documents. These methods are often prone to query drift (Mitra et al., 1998), which limits their applicability. Taking advantage of the performance of recent LLMs (Yang et al., 2025; Grattafiori et al., 2024; Team et al., 2025),

various LLM-based query rewriting methods have been proposed. Many works directly leverage the strong capabilities of LLMs to expand or rewrite queries through various prompting strategies, including natural language questions (Alaofi et al., 2023; Ye et al., 2023), keywords (Jagerman et al., 2023; Li et al., 2024; Mackie et al., 2023) or even passages (Gao et al., 2023; Wang et al., 2023; Shen et al., 2024; Zhang et al., 2024a). While this leads to an increased retrieval effectiveness, successful models rely on complex prompts, very large models, and multiple samples, each being costly at inference time. These drawbacks led to the development of approaches more closely related to ours, and which propose to finetune smaller LLMs. Following (Nogueira and Cho, 2017), several works (Mao et al., 2024; Ma et al., 2023; Peng et al., 2024; Hsu et al., 2024; Yao et al., 2025) propose to leverage reinforcement learning techniques using a multi-source reward – e.g. annotated samples, cross-encoder like MonoBERT (Nogueira and Cho, 2019). However, these methods are used within a multi-hop scenario (Hsu et al., 2024), with low efficiency, or are applied in a RAG system (Mao et al., 2024), and did not really look the underlying IR performance (as well as using a weaker reward signal as in our work).

3 Method

3.1 Problem Formulation

Our goal is to build a model able to generate a query specification $p_\theta(q)$ from an initial user query q, where the re-written query leads to both efficient and effective retrieval. To process the query specification $p_\theta(q)$, we use existing BoW search engines se, e.g. BM25 (Robertson et al., 1994), to process the search specification $p_\theta(q)$ depending on the search engine se. In this work, we use a bag-of-words model, namely BM25.

The search engine returns an ordered list of documents $se(p_\theta(q))$, whose quality can only be evaluated using IR metrics, which prevents having a direct signal to learn p_θ. We thus frame the problem as a reinforcement learning problem, where the policy p_θ corresponds to the generation, and the reward corresponds to an IR metric. More precisely, this metric is a scalar reward $R(q, p_\theta(q))$ that reflects the quality of the resulting query. Note that the reward should capture both the effectiveness and the efficiency of the resulting query. The goal is to learn a policy that generates higher-reward candidates.

3.2 Policy Optimization

We optimize the rewriting policy p_θ with Group-Relative Policy Optimization (GRPO) (Shao et al., 2024). For each query q, the policy samples a group of m candidates $\{q_i\}_{i=1}^m$, their associated rewards $\{r_i\}$ are calculated as described in Section 3.3, and group-relative advantages $a_i = r_i - \bar{r}$ (with $\bar{r} = \frac{1}{m}\sum_i r_i$) drive a clipped policy-gradient update. Following prior large-scale GRPO practice (Hu et al., 2025), we standardize rewards within each group and set the KL weight to $\beta = 0$ (omit an explicit KL penalty) to encourage exploration (ablations in Section 4 justify this choice).

Prompt. We conducted a prompt search (50+ candidates) to identify a minimal instruction that performs well zero-shot. Using a 1k-query subset of the MS MARCO training set, we selected the prompt that respects our keyword-writing format and maximized nDCG@10 with the Qwen3-4B policy. Our final prompt is:

> Generate relevant single-word keywords to improve retrieval performance. Only output unique keywords, separated by commas. [QUERY]: `{query}` [KEYWORDS]:

3.3 Reward

To drive the RL algorithm, our reward should be sensitive to any change in the query, even if it marginally improves effectiveness or efficiency. Note that in this work, we did not include any reward component related to efficiency, as our first experiments did not show any difference with efficiency-based rewards.

3.3.1 Soft and Hard nDCG

To comply with those properties, we need a reward that is sensitive not only to the order between documents but also to the magnitude of the search engine scores se. More precisely, if a relevant document is ranked after a non-relevant one, we still want to have a higher reward for the query where the difference in the scores is low rather than large.

An IR metric that matches our needs is SoftRank, which computes the expected nDCG (Taylor et al., 2008) as

$$\mathbb{E}(nDCG@k) = \mathbb{E}\left(\sum_{i=1}^{k} \frac{relevance(d_i)}{\log_2(1+K_i)}\right)$$

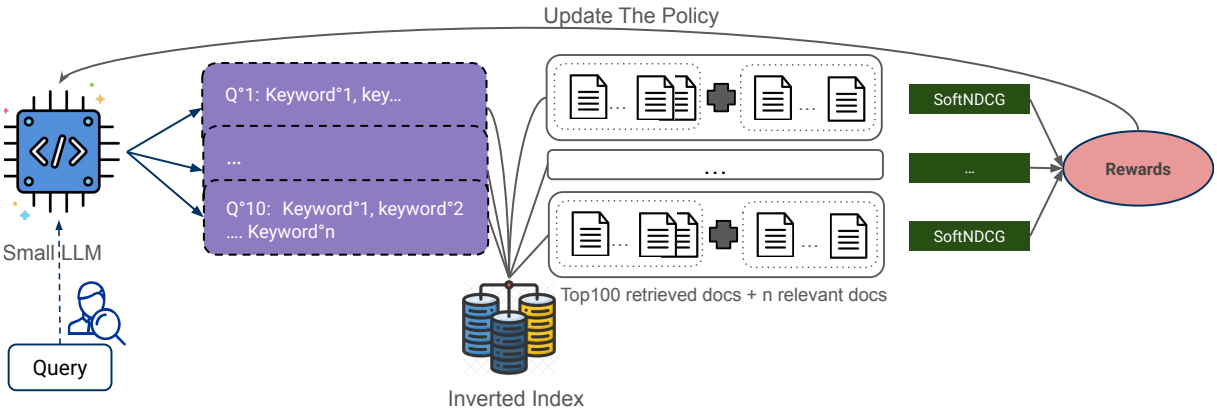


Figure 1: Overview of our query rewriting framework. A LLM generates multiple candidate queries, which retrieve documents using efficient index-based BoW IR models. The top-k retrieved results are annotated with a cross-encoder reference from which the expectation of nDCG, SoftNDCG, can be computed. The resulting rewards are then used to update the policy for improved reformulation.

where $relevance(d_i)$ is the relevance of the document d_i to the query at hand, and K_i a random variable corresponding to the rank of document i. SoftRank is based on the assumption that the scores returned by the search engines are normally distributed, i.e. $S(q,d) \sim \mathcal{N}(se(q,d), \nu Id)$ where σ, the standard deviation, is a hyper-parameter that does not depend on d or q.

Another way to see how this impacts the IR metric is to understand when a document d_i, with score s_i, is ranked before a document d_j with score s_j:

$$p(d_i \succ d_j) = \sigma\left(\frac{s_i - s_j}{\nu}\right), \qquad (1)$$

where σ is the sigmoid function and ν the standard deviation. As $\nu \to 0$, the model tends to behave as nDCG as the rank is directly driven by whether $s_i > s_j$ (the above probability tends towards 0 or 1). When $\nu \to \infty$, $p(d_i \succ d_j) \to \frac{1}{2}$ so that no score impacts $\mathbb{E}(nDCG)$. In practice (see §4.3), a moderate ν yields the best trade-off between stability and discriminability. We name the two metrics HardNDCG (when $\nu \to 0$) and SoftNDCG (when ν has a moderate value).

An alternative metric would be to use the InfoNCE loss (Oord et al., 2018), commonly used to train neural IR models. However, in our preliminary experiments the training was unstable compared to the one using SoftRank (Taylor et al., 2008).

3.3.2 Distillation: Cross-Encoder Assessments

The dataset used most often to train IR models is MS MARCO (Bajaj et al., 2016), where the evaluations of whether a document is relevant to a given user query are based on user clicks. User clicks are known to be incomplete (Gupta and MacAvaney, 2022): many queries are associated with a single relevant document, i.e. MS MARCO contains many *false* negatives. In our case, $\mathbb{E}(nDCG)$ can be zero for all query candidates, which cannot be used for GRPO to learn properly. To tackle this issue, we use distillation (Hofstätter et al., 2021), a standard technique used to train IR models that takes advantage of the most powerful IR models, namely cross-encoders (CE), to indicate whether or not a document is relevant.

4 Experiments, Results and Analysis

In this section, we first describe the experiments we developed to validate our model. Then, we analyze our experimental results, including the main result, the ablation studies, as well as some analysis of the efficiency.

4.1 Experimental Setup

Datasets We train our model on the MS MARCO v1 passage dataset (Bajaj et al., 2016). This dataset contains 8.8M passages. We evaluate the model *in-domain* on the MS MARCO dev set (6980 queries), as well as the TREC DL 2019 (43 assessed topics) and TREC DL 2020 (54 assessed topics) sets. Following common practice, we report out-of-domain (OOD) performance of our method using the BEIR dataset (Thakur et al., 2021).

Baselines We consider the following baselines for our experiments, including the neural dual models:

- ANCE (Karpukhin et al., 2020), a state-of-the-art dense neural IR model;
- SPLADEv2 (Formal et al., 2021), a state-of-the-art sparse neural IR model;
- ColBERTv2 (Santhanam et al., 2021), a state-of-the-art multi-vector retrieval model, which is especially good at generalization.

We also compare with the BM25 variants, with or without query rewriting:

- A standard IR baseline, BM25 (Robertson et al., 1994), which also serve as the retrieving model for our rewritten queries.
- RM3 (Lavrenko and Croft, 2001), a query rewriting model based on relevance feedback for BM25 retrieval.
- HyDE (Gao et al., 2023), a query rewriting model based on the generation of relevant passages;
- Query2Doc (Wang et al., 2023) (abbr. Q2D), similar to HyDE, but that concatenates the query for BM25 retrieval.
- LameR (Shen et al., 2024), a LLM-based query rewriting model that concatenates the top retrieved documents.
- MuGI (Zhang et al., 2024a), a state-of-the-art work that balances the query and pseudo-document for query rewriting. We report both the result based on ChatGPT3.5 and ChatGPT4.

The results for SPLADEv2 and ColBERTv2 are copied from (Santhanam et al., 2021). The results for ANCE, RM3 and Q2D are copied from (Wang et al., 2023). The other baselines are from (Zhang et al., 2024a). We report very few generative retrieval baseline results as they do not usually evaluate on BEIR and on the *full* MS MARCO dataset.[3] Among all the results reported in these papers, we generally select the ones reported in the tables by using the configuration put forward by the author. For the neural dual encoders, we take the result of the model trained **with** distillation (Hofstätter et al., 2021), as we are also distilling from a cross-encoder.

Implementation details We conduct all our experiments[4] with NVIDIA RTX A6000 48GB gpus (total cost $150 - $200). For GRPO training, we fine-tune Qwen3 models (Yang et al., 2025) model[5] across various size (0.6B, 1.7B and 4B) with LoRA (Hu et al., 2022) (rank $r = 40$, scaling $\alpha = 40$, "thinking" mode disabled). The 4B version is our main model as it offers the best efficiency/effectiveness trade-off. For each input query, we sample from the LLM a group of 10 candidates with a temperature $\tau = 1.2$. The model is trained on 96,000 randomly sampled queries from MS MARCO training set. As for the assessments, we either use the original assessments from MS MARCO train set, or the cross-encoder ones from OpenSearch [6]. Cross-Encoder-based relevance assessments are normalized by dividing the maximum score by the maximum score for a given query, leading to values in the range $[0, 1]$.

For optimization, we use the AdamW optimizer with a learning rate of 5e-6, a batch size of 320 (using 20 micro-steps for one gradient update), and trained for one epoch. Our largest model (4B) takes around 2 days to train on a single GPU. When evaluating, we generate one rewritten query with greedy decoding (temperature $\tau = 0$). During inference, processing the MS MARCO dev set (6980 queries) takes only around 5 minutes (batch size of 256). The model parameters as well as the code will be made publicly available.

4.2 Results and Analysis

In this section, we perform a quantitative and qualitative analysis based on our experimental results.

4.2.1 Main results

In this section, we present the results of our model QUESTER which is based on a Qwen-4B backbone, using cross-encoder assessments, a standard deviation $\nu = 0.5$ when computing SoftNDCG and a cut-off at 10,000.[7]

In-domain On MS MARCO Dev and TREC DL'19/'20 (Table 1), our GRPO-trained policy **QUESTER** improves over BM25 and Qwen3-4B zero-shot prompting. Compared to more closely related approaches (Q2D, HyDE, LameR, MuGI), QUESTER has strong effectiveness while avoiding their generation overhead and instability. Compared to the state-of-the-art generative retrieval model like SEAL and MINDER, our model also

[3] We only report MINDER (Li et al., 2023) for which a few results on full MS MARCO are reported.

[4] Huggingface Transformers 4.53.3, TRL 0.19.1

[5] QWEN3

[6] OpenSearch Project – MS MARCO Hard Negatives LLM Scores

[7] Note that because labeling with cross-encoder is costly, we assume that a document is not relevant if not in the top-100 re-labeled by the CE

achieve a better effectiveness, which illustrate our proposed methods could perform better to large scale dataset. Finally, compared to dual encoders (ANCE, SPLADEv2, ColBERTv2), which have been trained on MS MARCO, QUESTER – as the other query rewriting approaches – have a much lower nDCG@10 (on DL'19 and '20, is 71.7 for SPLADEv2 vs 62 for QUESTER) but a competitive R@1K (78.9 vs 81.8).

Out-of-domain. On OOD datasets BEIR (Thakur et al., 2021) , QUESTER outperforms all the neural IR baselines (SPLADEv2, ColBERTv2) as reported in Table 2 – showing the interest of such approaches for OOD retrieval. It is also competitive with query rewriting models (e.g., MuGI/GPT4 and Query2Doc), while being more efficient (see Section 4.2.2). By contrast, prompt-heavy methods (HyDE, MuGI, and LameR) rely on large proprietary LLMs and stochastic decoding (temperature $\tau > 0$), so their outputs are expensive to compute (several subsets are omitted due to the generation cost) and scores vary across runs. We instead decode with temperature $\tau = 0$, producing deterministic, fully reproducible rewrites.

model	Dev set		DL19		DL20	
	RR@10	R@1K	nDCG@10	R@1K	nDCG@10	R@1K
Dual encoder models						
ANCE	33.0	95.9	64.5	75.5	64.6	77.6
SPLADEv2	36.8	97.9	72.9	74.7	68.7	83.0
ColBERTv2	**39.7**	**98.4**	**74.5**	**82.6**	**75.6**	**84.3**
Generative Retrieval Models						
SEAL	12.7	—	—	—	—	—
MINDER	18.6	—	—	—	—	—
BM25 variants						
BM25	18.4	85.3	50.6	73.9	48.0	72.3
+RM3	15.7	86.1	52.2	81.4	49.0	82.4
+Q2D	21.4	91.8	66.2	—	62.9	—
+HyDE	—	—	61.3	88.0	57.9	84.4
+LameR	—	—	67.1	**89.9**	62.7	**88.7**
+MUGI$^{GPT3.5}$	—	—	70.4	—	63.9	—
+MUGI^{GPT4}	—	—	70.4	—	64.4	—
Ours						
Qwen3-base	9.1	80.5	27.1	52.0	23.9	47.5
QUESTER	22.4*	92.1*	63.1*	82.1*	60.8*	81.5*

Table 1: In-domain evaluation on MS MARCO Dev, TREC DL'19, and TREC DL'20; QUESTER is our main GRPO-trained model and Qwen3-base is directly prompting the LLMs for query specification generation. Best performing model is in **bold** and the best performing *BM25 variant* model is underlined. * = statistically significant difference between QUESTER and BM25 with $p < 0.05$

4.2.2 Efficiency Analysis

Figure 2 summarizes the efficiency-effectiveness trade-off of the different models (using one thread

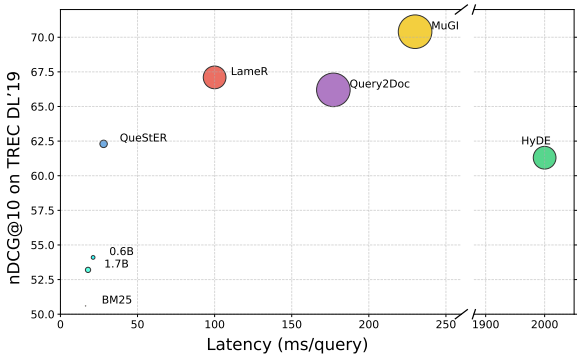


Figure 2: Trade-off between efficiency (ms/query, lower is better) and effectiveness (nDCG@10 on DL19, higher is better) for retrieval (generation time is not reported). Bubble size indicates model size (billions of parameters). Our QUESTER offers a favorable balance, approaching MuGI and LameR in quality while being 4–7× faster.

when retrieving with Lucene, on the 43 queries from DL'19). Leaving out the generation processing time, BM25 has the best efficiency (16.3 ms/query) but is less effective (nDCG@10 is 50.6). Our model QUESTER has a strong balance (62.3 nDCG@10 with 28 ms/query). Our smaller variants lag in quality despite better efficiency. Prompt-expansion methods such as MuGI and LameR achieve a higher nDCG@10 but at the cost of efficiency (more than 100 ms/query).

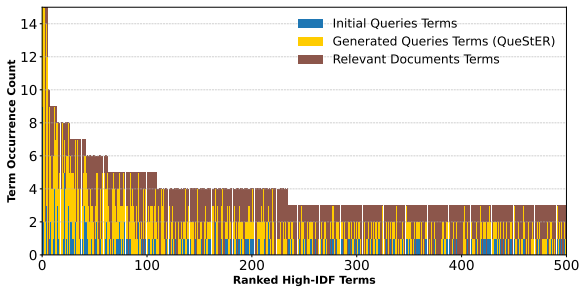
Generation time QUESTER generates a single keyword specification capped at 64 tokens using a small open model ($\leq 4B$ parameters). In contrast, MuGI and LameR rely on large API-based LLMs (e.g., GPT-3.5/4, ∼170B) and can generate substantially longer expansions; for instance, MuGI may produce 4–5 rewrites of up to 512 tokens each per input query. As a result, end-to-end generation latency is not directly comparable across methods, and Figure 2 reports retrieval-time latency only.

4.2.3 Qualitative Analysis

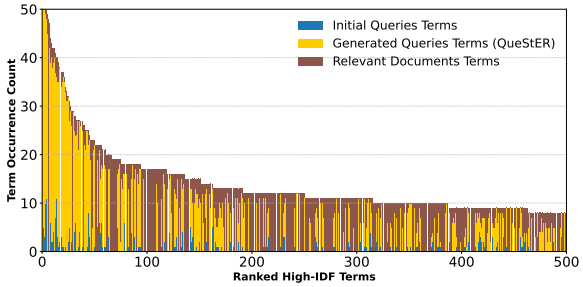
We analyze here the query reformulations that our model generates. We analyze the overlap of high-IDF keywords in-domain (TREC DL'19/20) and out-of-domain (NFCorpus) in Figure 3. In both cases, the word frequency distributions align better between rewritten queries and relevant documents than with original queries. On TREC DL, the overlap is especially pronounced, with generated queries closely mirroring the vocabulary of relevant documents. On NFCorpus, while the alignment is lesser due to domain shift,

Method	TC	NF	TO	DB	SF	SG	NW	RB	FQ	AG	Avg.	Our Avg
ANCE	65.4	23.7	24.0	28.1	50.7	24.9	38.2	39.2	30.0	41.5	36.8	**47.1**
SPLADEv2	71.0	33.4	27.2	43.5	69.3	29.6	39.4	45.8	33.6	**47.9**	44.1	**47.1**
ColBERTv2	73.8	33.8	26.3	**44.6**	69.3	33.2	46.0	47.5	**35.6**	46.3	45.6	**47.1**
BM25 variants												
BM25	59.5	30.8	44.2	31.8	67.9	33.1	39.5	40.7	23.6	39.7	41.1	**47.1**
+ RM3	59.3	33.1	35.0	30.8	64.6	31.5	42.6	44.3	19.2	38.0	39.8	**47.1**
+ Q2D	72.2	34.9	39.8	37.0	68.6	—	—	—	—	—	50.5	**53.1**
+ HyDE	69.1	—	—	36.8	69.1	—	44.0	—	27.3	46.6	48.8	**50.1**
+ LameR	**75.8**	—	—	39.0	73.5	—	50.3	—	25.8	40.2	**50.8**	50.1
+ MuGI$^{\text{GPT3.5}}$	69.7	36.0	46.3	41.2	72.0	35.8	46.6	49.7	—	—	49.6	**50.2**
+ MuGI$^{\text{GPT4}}$	72.9	**37.4**	46.1	42.7	**74.0**	**36.0**	**50.0**	49.2	—	—	**51.0**	49.7
QUESTER (ours)	73.6*	36.0*	**47.7**	38.8*	69.3	34.8	45.3*	**51.7***	27.5*	46.1*		

Table 2: OoD performance (nDCG@10; higher is better) on selected BEIR datasets. A dash (—) indicates a value not reported. The rightmost columns show the macro-average across listed datasets for which a value is reported (Avg). We report the average of QUESTER on the same subset of datasets (Our Avg). Best performing model is in **bold** and the best performing *BM25 variant* model is underlined. * = statistically significant difference between QUESTER and BM25 with $p < 0.05$



(a) In-domain (TREC DL19 and DL20).



(b) Out-of-domain (NFCorpus).

Figure 3: Keyword overlap distribution between original queries (blue), generated queries (yellow), and ground-truth relevant passages (brown). Generated queries consistently inject discriminative vocabulary that aligns with relevant documents, both ID and OOD.

generated queries have a distribution that aligns much more with that of relevant documents. Together, these results reinforce our statement that QUESTER query specifications not only enhance query–document matching in-domain, but also generalize effectively to unseen domains.

An example of a query rewritten by QUESTER on NFCorpus illustrates how the model expands a short, under-specified query into a richer set of semantically related phrases, improving lexical diversity and alignment with relevant documents:

Initial query: veggie chicken

Generated query: veggie chicken recipe; chicken vegetable dish; veggie chicken healthy option; chicken stuffed with vegetables; veggie chicken recipe vegetarian; chicken vegetable mixture; veggie chicken salad; vegetable chicken recipe common; chicken vegetable used in vegetarian meal; veggie chicken substitute meat; chicken dish vegetable; vegetable chicken recipe.

Final query (merging terms) chicken (×12), vegetable (×6), veggie (×5), recipe (×4), dish, salad, stuffed, healthy, mixture, substitute, meal.

4.3 Ablations

In this section, we perform several ablation studies on various hyperparameters of our model. All the ablation results described below are reported Table 3.

(i) Effect of Model size Since the number of parameters directly affects model latency (Figure 2), we examine the trade-off between efficiency and effectiveness across different parameter sizes (0.6B, 1.7B, and 4B). Performance improves consistently with scale. The 4B model achieves the highest average nDCG, followed closely by the 1.7B variant, while the 0.6B version lags behind. Although the 1.7B model offers a favorable latency–accuracy balance, the 4B configuration provides the most robust improvements.

Model	AG	CF	TC	DB	FV	FQ	HP	NW	NF	NQ	RB	SD	SF	SG	TO	Avg.
BM25	39.7	16.5	59.5	31.8	65.1	23.6	63.3	39.5	32.2	30.5	40.8	14.9	67.9	33.0	44.2	40.2
Qwen3-base	29.6	17.9	65.1	24.9	40.5	21.6	33.6	**47.9**	31.3	31.0	35.5	12.2	67.2	16.3	34.1	33.9
QUESTER	**46.1***	19.2*	73.6*	38.8*	70.3*	**27.5***	64.3*	45.3*	36.0*	43.0*	**51.7***	15.1	69.3	34.8	47.7	45.5*
(i) Effect of Model size																
0.6B	40.5	17.6	69.3	34.0	67.1	26.0	62.3	41.5	33.9	36.2	48.1	14.9	69.1	32.6	44.4	42.5
1.7B	44.7	18.0	69.9	34.8	66.9	25.9	62.8	42.4	34.9	36.8	48.7	**15.4**	69.9	33.9	44.1	43.3
(ii) Supervised fine-tuning (SFT)																
SFT	33.9	16.7	59.5	25.2	60.8	20.5	60.4	38.7	30.7	30.9	42.1	14.3	70.3	27.4	33.6	37.2
SFT+GRPO	45.4	17.8	72.6	36.8	67.8	26.6	63.6	42.7	34.8	40.8	49.7	15.0	68.5	33.4	45.2	44.0
(iii) Hard-label assessments vs. CE gains																
Hard	45.7	17.8	71.8	38.3	**70.7**	27.0	63.3	44.8	35.6	42.7	**51.7**	15.3	**70.9**	34.3	46.9	45.1
Hard-w/o CE	43.9	18.0	72.0	36.9	69.1	27.2	62.0	43.1	34.9	42.0	50.9	15.1	70.0	31.4	**48.2**	44.3
Soft-w/o CE	44.5	17.1	72.0	37.1	67.5	26.8	62.8	44.4	35.3	41.3	48.9	15.1	70.3	33.4	46.4	44.2
(iv) Effect of standard deviation ν																
$\nu = 1$	44.9	18.0	71.9	36.9	69.5	25.7	62.8	43.5	36.0	39.6	48.7	15.2	70.0	33.3	39.6	43.7
$\nu = 0.05$	44.8	18.9	72.2	36.4	68.5	27.8	62.7	43.8	33.1	41.2	48.9	14.8	70.3	33.1	41.9	43.9
(v) KL weight β ablation																
KL5%	38.6	19.1	64.4	32.1	65.8	23.2	58.9	40.9	33.6	36.8	46.1	14.3	68.8	32.4	39.4	41.0
KL1%	41.1	22.4	66.7	35.4	70.5	25.7	62.1	45.7	35.5	41.2	52.4	15.2	71.2	30.1	37.9	43.5
(vi) Effect of Cut-off k for nDCG@k																
k=100	45.1	18.3	73.2	37.7	69.0	26.8	62.4	46.1	**36.4**	42.8	**51.7**	15.0	69.2	32.2	46.4	44.8
(vii) concatenating original query during training vs. concatenating original query only in inference																
Training	44.9	18.5	**74.5**	37.5	70.3	27.4	62.7	44.5	36.2	**43.6**	**51.7**	**15.4**	**70.9**	33.0	47.8	45.3
Inference	44.3	**19.4**	74.0	**39.1**	**70.4**	27.3	**65.5**	45.1	35.9	42.5	51.1	15.2	69.9	**35.8**	**47.9**	**45.6**

Table 3: nDCG@10 on 15 BEIR datasets with heterogeneous query profiles (datasets with many queries and longer descriptions vs. short keyword queries). The table reports per-dataset scores and macro-average; per-dataset maxima are in bold (ties bolded). QUESTER is our main GRPO-trained model and Qwen3-base is directly prompting the LLMs for query specification generation. Ablations vary (i) model size (0.6B/1.7B/4B), (ii) the usage of SFT as a training starting point, (iii) supervision (hard qrels vs. CE-derived graded gains), (iv) SoftRank standard deviation ν, (v) KL weight β, (vi) evaluation pool (BM25 top-100 only vs. top-100 plus CE-labeled documents within 10k) and (vii) concatenating the original query during training or inference. (statistically significant for compared to BM25 with $p = 0.05$)

(ii) Original/SFT Checkpoint Training Analysis RaFE (Mao et al., 2024) uses supervised fine-tuning (SFT) to warm up the model. To investigate the effect of SFT in our case, we conduct experiments in which we randomly sample 50,000 queries (from the MS MARCO training dataset, with no overlap with the queries used to train the model with RL). We use a large model (Qwen3-32B) to generate rewritten queries, generating 20 reformulated queries, keeping the best one *if improving retrieval performance*. These couples (query, reformulated query) are used as a supervised dataset.

Using SFT improves over the backbone model, but does not reach BM25 performance. If we further train using GRPO (with SoftNDCG/CE), this further improves effectiveness but does not outperform the QUESTER model without SFT. Looking at generated queries, we hypothesize that using SFT narrows the rewrite space and reduces exploration.

(iii) Effect of different supervision source The results clearly show that replacing the sparse click-based labels with CE-derived labels produces more stable and effective learning. Hard and Soft variants trained without CE perform worse (–1 nDCG@10), confirming that dense, graded CE supervision is crucial to guide GRPO toward consistent improvements as it is to train neural IR models.

(iv) Effect of ν The standard deviation ν in SoftRank (Section 3.3.1) controls the smoothness of pairwise document comparisons in SoftNDCG. As shown in Table 3, moderate smoothing with our main model QUESTER that has ν=0.5 achieves the best average nDCG@10 (45.5), outperforming both sharper ν=0.05 and softer ν=1 variants. A small ν makes the ranking signal too unstable, while a large ν over-smooths relevance differences, weakening the learning signal. These results suggest that moderate variance provides a stable yet discriminative reward for effective policy optimization.

(v) Effect of β We evaluated the impact of the KL-divergence weight (in GRPO) by testing $\beta \in \{0.0, 0.01, 0.05\}$. As shown in Table 3, setting $\beta = 0.0$ (no KL regularization) yields the highest stability and overall nDCG@10 (45.5), while larger

values consistently degrade performance (e.g., 40.9 at $\beta = 0.05$) . A higher β overly constrains the policy to be close to the reference model, limiting exploration and reward optimization.

(vi) Effect of Cut-off k Finally, we need to set the cutoff value k used to compute $\mathbb{E}(nDCG@k)$. We see that using $k = 100$ consistently decreases the performance of the model, compared to the $k = 10,000$ – and this, even when assuming that documents not re-labeled by the CE are not relevant.

(vii) Effect of using the original query Following MuGI (Zhang et al., 2024a), we examine whether reusing the original user query helps retrieval. More precisely, we tried to add it after training (*inference*), but this only provides a negligible gain (+0.1 in nDCG@10). Adding the query during both training and inference has a slight negative effect (-0.2 in nDCG@10). However, both strategies slightly improve the consistency of the performance across datasets, but more work is needed to understand exactly the effect of this modification.

5 Conclusion and Future Work

In this paper, we present QUESTER, a generative retrieval approach that rewrites queries, which is trained with GRPO with a reward based on SoftRank (Taylor et al., 2008) and distillation. We show that QUESTER has the best tradeoff between efficiency and effectiveness in its class of models (LLM-based query rewriting) and is competitive out-of the domain with neural IR models, even if it still lags behind when in-domain. This shows the potential of this type of approach – we plan to explore more structured (and thus expressive) query languages as well as other optimization techniques in the future.

Limitations

Public models (like Qwen or to a lesser extent cross-encoders) may contain data that overlap either with MS MARCO and/or BEIR. Measuring precisely how much data contamination is an issue we share with all the other works presented here.

In this work, we do not consider more structured query languages (e.g. with boolean, phrase, or proximity specifications) or hybrid dense–sparse backends. Although an interesting future avenue, this first work has shown that it was possible to improve existing related works in terms of efficiency while keeping almost the same effectiveness.

Acknowledgments

The authors acknowledge Air Liquide and ANR - FRANCE (French National Research Agency) for its financial support of the GUIDANCE project n°ANR-23-IAS1-0003. This work was granted access to the HPC resources of IDRIS & CINES (Project No. A19) under the allocation made by GENCI.

References

Nasreen Abdul-Jaleel, James Allan, W. Bruce Croft, Fernando Diaz, Leah Larkey, Xiaoyan Li, Mark D. Smucker, and Courtney Wade. 2004. Umass at trec 2004: Novelty and hard. In *Text REtrieval Conference (TREC)*. NIST.

Marwah Alaofi, Luke Gallagher, Mark Sanderson, Falk Scholer, and Paul Thomas. 2023. Can generative llms create query variants for test collections? an exploratory study. In *ACM Conference on Research and Development in Information Retrieval (SIGIR)*.

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*.

Soyuj Basnet, Jerry Gou, Antonio Mallia, and Torsten Suel. 2024. Deeperimpact: Optimizing sparse learned index structures. *Preprint*, arXiv:2405.17093.

Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive search engines: Generating substrings as document identifiers. *Advances in Neural Information Processing Systems*, 35:31668–31683.

Claudio Carpineto and Giovanni Romano. 2012. A survey of automatic query expansion in information retrieval. *Acm Computing Surveys (CSUR)*, 44(1):1–50.

Jiangui Chen, Ruqing Zhang, Jiafeng Guo, Yixing Fan, and Xueqi Cheng. 2022a. Gere: Generative evidence retrieval for fact verification. In *ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 2184–2189.

Jiangui Chen, Ruqing Zhang, Jiafeng Guo, Yiqun Liu, Yixing Fan, and Xueqi Cheng. 2022b. Corpusbrain: Pre-train a generative retrieval model for knowledge-intensive language tasks. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 191–200.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2020. Autoregressive entity retrieval. *arXiv preprint arXiv:2010.00904*.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021. Splade v2: Sparse lexical and expansion model for information retrieval. *arXiv preprint arXiv:2109.10086*.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1762–1777.

Adrien Grand, Robert Muir, Jim Ferenczi, and Jimmy Lin. 2020. From MAXSCORE to Block-Max Wand: The Story of How Lucene Significantly Improved Query Evaluation Performance. In *Advances in Information Retrieval*, Lecture Notes in Computer Science, pages 20–27, Cham. Springer International Publishing.

Michael Granitzer, Stefan Voigt, Noor Afshan Fathima, Martin Golasowski, Christian Guetl, Tobias Hecking, Gijs Hendriksen, Djoerd Hiemstra, Jan Martinovič, Jelena Mitrović, Izidor Mlakar, Stavros Moiras, Alexander Nussbaumer, Per Öster, Martin Potthast, Marjana Senčar Srdič, Sharikadze Megi, Kateřina Slaninová, Benno Stein, and 4 others. 2024. Impact and development of an open web index for open web search. *J. Assoc. Inf. Sci. Technol.*, 75(5):512–520.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Prashansa Gupta and Sean MacAvaney. 2022. On survivorship bias in ms marco. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2214–2219. ACM.

Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling. *arXiv:2104.06967 [cs]*. ZSCC: 0000000 arXiv: 2104.06967.

Sheryl Hsu, Omar Khattab, Chelsea Finn, and Archit Sharma. 2024. Grounding by trying: Llms with reinforcement learning-enhanced retrieval. *arXiv preprint arXiv:2410.23214*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. 2025. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*.

Rolf Jagerman, Honglei Zhuang, Zhen Qin, Xuanhui Wang, and Michael Bendersky. 2023. Query expansion by prompting large language models. *arXiv preprint library*, abs/2305.03653.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781. Association for Computational Linguistics.

Victor Lavrenko and W. Bruce Croft. 2001. Relevance-based language models. In *ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 120–127. ACM.

Sunkyung Lee, Minjin Choi, and Jongwuk Lee. 2023. Glen: Generative retrieval via lexical index learning. *arXiv preprint arXiv:2311.03057*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Minghan Li, Honglei Zhuang, Kai Hui, Zhen Qin, Jimmy Lin, Rolf Jagerman, Xuanhui Wang, and Michael Bendersky. 2024. Can query expansion improve generalization of strong cross-encoder rankers? In *ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 2321–2326.

Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2023. Multiview identifiers enhanced generative retrieval. *arXiv preprint arXiv:2305.16675*.

Xinyu Ma, Yixing Gong, Pengjie He, Heng Zhao, and Nan Duan. 2023. Query rewriting for retrieval-augmented large language models. In *emnlp*.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2024. Finetuning llama for multi stage text retrieval. In *ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 3164–3168.

Iain Mackie, Shubham Chatterjee, and Jeffrey Dalton. 2023. Generative and pseudo-relevant feedback for sparse, dense and learned sparse retrieval. *arXiv preprint library*, abs/2305.07477.

Shengyu Mao, Yong Jiang, Boli Chen, Xiao Li, Peng Wang, Xinyu Wang, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. 2024. Rafe: Ranking feedback improves query rewriting for rag. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Mandar Mitra, Amit Singhal, and Chris Buckley. 1998. Improving automatic query expansion. In *Proceedings of the 21st Annual International ACM SIGIR*

Conference on Research and Development in Information Retrieval, SIGIR '98, page 206–214, New York, NY, USA. Association for Computing Machinery.

Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-oriented query reformulation with reinforcement learning. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Seongwan Park, Taeklim Kim, and Youngjoong Ko. 2025. Decoding dense embeddings: Sparse autoencoders for interpreting and discretizing dense retrieval. *arXiv preprint arXiv:2506.00041*.

Wenjun Peng, Guiyang Li, Yue Jiang, Zilong Wang, Dan Ou, Xiaoyi Zeng, Derong Xu, Tong Xu, and Enhong Chen. 2024. Large language model based long-tail query rewriting in taobao search. In *Companion Proceedings of the ACM Web Conference 2024*, pages 20–28.

Ronak Pradeep, Kai Hui, Jai Gupta, Adam D Lelkes, Honglei Zhuang, Jimmy Lin, Donald Metzler, and Vinh Q Tran. 2023. How does generative retrieval scale to millions of passages? *arXiv preprint arXiv:2305.11841*.

Stephen Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at trec-3. In *''*, pages 0–.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2021. Colbertv2: Effective and efficient retrieval via lightweight late interaction. *arXiv preprint arXiv:2112.01488*.

Arthur Satouf, Gabriel Ben Zenou, Benjamin Piwowarski, Habiboulaye Amadou Boubacar, and Pablo Piantanida. 2025. Rational retrieval acts: Leveraging pragmatic reasoning to improve sparse retrieval. *Preprint*, arXiv:2505.03676.

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*.

Tao Shen, Guodong Long, Xiubo Geng, Chongyang Tao, Tianze Zhou, and Daxin Jiang. 2024. Retrieval-augmented retrieval: Large language models are strong zero-shot retriever. *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten Rijke, and Zhaochun Ren. 2023. Learning to tokenize for generative retrieval. *Advances in Neural Information Processing Systems*, 36:46345–46361.

Yubao Tang, Ruqing Zhang, Jiafeng Guo, Jiangui Chen, Zuowei Zhu, Shuaiqiang Wang, Dawei Yin, and Xueqi Cheng. 2023. Semantic-enhanced differentiable search index inspired by learning strategies. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4904–4913.

Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, and 1 others. 2022. Transformer memory as a differentiable search index. *Advances in Neural Information Processing Systems*, 35:21831–21843.

Michael Taylor, John Guiver, Stephen Robertson, and Tom Minka. 2008. Softrank: optimizing non-smooth rank metrics. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, page 77–86, New York, NY, USA. Association for Computing Machinery.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, and 1 others. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *neurips*.

Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query expansion with large language models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, and 1 others. 2022. A neural corpus indexer for document retrieval. *Advances in Neural Information Processing Systems*, 35:25600–25614.

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*.

Tianchi Yang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, and Qi Zhang. 2023.

Auto search indexer for end-to-end document retrieval. *arXiv preprint arXiv:2310.12455*.

Sijia Yao, Pengcheng Huang, Zhenghao Liu, Yu Gu, Yukun Yan, Shi Yu, and Ge Yu. 2025. Llm-qe: Improving query expansion by aligning large language models with ranking preferences. *arXiv preprint arXiv:2502.17057*.

Fanghua Ye, Meng Fang, Shenghui Li, and Emine Yilmaz. 2023. Enhancing conversational search: Large language model-aided informative query rewriting. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Le Zhang, Yihong Wu, Qian Yang, and Jian-Yun Nie. 2024a. Exploring the best practices of query expansion with large language models. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Peitian Zhang, Zheng Liu, Yujia Zhou, Zhicheng Dou, Fangchao Liu, and Zhao Cao. 2024b. Generative retrieval via term set generation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 458–468.

Yujia Zhou, Jing Yao, Zhicheng Dou, Ledell Wu, Peitian Zhang, and Ji-Rong Wen. 2022. Ultron: An ultimate retriever on corpus with a model-based indexer. *arXiv preprint arXiv:2208.09257*.