ExpandR: Teaching Dense Retrievers Beyond Queries with LLM Guidance

Anonymous ACL submission

Abstract

Large language models (LLMs) have demonstrated significant potential in enhancing dense retrieval through query augmentation. However, most existing methods treat the LLM and the retriever as separate modules, overlooking the alignment between generation and ranking objectives. In this work, we propose **ExpandR**, a unified LLM-augmented dense retrieval framework that jointly optimizes both the LLM and the retriever. ExpandR employs the LLM to generate semantically rich query expansions, which are leveraged to enhance the retriever's training. Simultaneously, the LLM is trained using Direct Preference Optimization (DPO), guided by a carefully designed reward function that balances retrieval effectiveness and generation consistency. This joint optimization paradigm enables mutual adaptation between the LLM and the retriever, resulting in query expansions that are both informative and well-suited for retrieval. Experimental results on multiple benchmarks show that ExpandR consistently outperforms strong baselines, achieving more than a 5% improvement in retrieval performance. All code will be publicly released on GitHub.

1 Introduction

Dense retrievers (Karpukhin et al., 2020; Xiong et al., 2021a) encode both queries and documents into the same embedding space, enabling efficient similarity-based retrieval via approximate KNN search (Johnson et al., 2019). While effective, their performance remains highly sensitive to the quality of the input query. In practice, user queries (Belkin et al., 1982; Ingwersen, 1996) are often short and ambiguous, leading to a significant semantic gap between the query and relevant documents, making it challenging for dense retrievers to accurately capture the underlying information need.

Recent advances in Large Language Models (LLMs) offer promising solutions to this challenge through query augmentation (Wei et al., 2022b; Huang et al., 2024a; Wei et al., 2022a). Existing methods along this line of research can be categorized into two groups. The first direction leverages LLM-generated reformulations as supervision signals to train dense retrieval models, typically through contrastive training (Zhang et al., 2025; Ma et al., 2025) or ranking probability distillation (Shi et al., 2024; Kim and Baek, 2025). However, the effectiveness of this approach is constrained by the limited capacity and scalability of dense retrievers (Fang et al., 2024). The second direction focuses on augmenting dense retrievers by prompting LLMs to generate additional terms at inference time (Wang et al., 2023a; Mackie et al., 2023). These terms aim to increase lexical overlap with relevant documents, thereby reducing the semantic gap between queries and documents. While such expansions are often semantically rich, they are typically misaligned with the retriever, as the LLM is not explicitly optimized for retrieval objectives. As a result, the retriever struggles to effectively utilize the LLM-augmented content.

In this work, we propose **ExpandR**, a unified LLM-augmented dense retrieval framework that jointly optimizes both the LLM and the dense retriever. ExpandR first prompts the LLM to generate semantically enriched query expansions, which enhance query representations and improve the retriever's ability to rank relevant documents. Rather than treating the LLM and retriever as separate modules, ExpandR integrates generation and retrieval under a shared training objective—promoting higher ranks for ground-truth documents given a query. Specifically, we optimize the dense retriever via contrastive training, and train the LLM using Direct Preference Optimization (DPO) with a combination of self-consistency and retrieval-based rewards. Through this joint optimization, the two components mutually reinforce each other, leading to more effective expansions and improved overall retrieval performance.

Our experiments on the BEIR benchmark (Thakur et al., 2021) demonstrate the effectiveness of ExpandR, yielding over a 5.8% improvement in supervised dense retrieval. Further analysis shows that the query expansions generated by ExpandR lead to better alignment with relevant documents compared to those from baseline methods. By jointly leveraging self-consistency and retrieval-based rewards, the LLM is better optimized to generate expansions that are both semantically rich and retriever-aligned. Specifically, the self-consistency reward encourages the LLM to generate content that is semantically closer to the ground-truth document, while the retrieval-based reward captures the retriever's ranking behavior. Together, these rewards guide the LLM to produce expansions that are both relevant and retriever-friendly.

2 Related Work

Dense retrievers (Karpukhin et al., 2020; Xiong et al., 2021a; Izacard et al., 2021; Yu et al., 2021; Xiong et al., 2021b; Li et al., 2021) conduct semantic matching by encoding queries and documents into a shared embedding space, thereby alleviating the vocabulary mismatch problem (Belkin et al., 1982). To further improve the quality of semantic matching, recent work has focused on refining this embedding space through contrastive learning with relevance supervision (Karpukhin et al., 2020; Zhan et al., 2021) or leveraging weakly supervised training signals (Xie et al., 2023). While effective, a persistent bottleneck in information retrieval lies in the quality of the user-issued queries themselves (Jiang et al., 2025). In particular, queries are often underspecified, ambiguous, or semantically incomplete, which limits the retriever's ability to accurately locate relevant content (Belkin et al., 1982; Ingwersen, 1996).

Recent advances in LLMs (Achiam et al., 2023; GLM et al., 2024) offer new opportunities to address this issue by leveraging their rich knowledge and powerful generative capabilities to enrich or reformulate user queries (Yu et al., 2020; Lin et al., 2020; Ye et al., 2023). These augmented queries are often used as supervision signals or distillation targets to train dense retrievers more effectively. For instance, methods such as LLM-QL (Zhang et al., 2025) and DRAMA (Ma et al., 2025) propose leveraging LLMs to generate new queries or training triplets for dense retriever optimization. RePlug (Shi et al., 2024) has been proposed to distill the knowledge of LLMs into a lightweight retriever. While these approaches enhance supervised retrieval performance, they mainly focus on query synthesis, often overlooking the limited semantic expressiveness of the original queries (Wang et al., 2023b). Moreover, their effectiveness is fundamentally constrained by the limited capacity and scalability of dense retrievers (Huang et al., 2024b).

LLM-based query expansion has emerged as a widely adopted approach for query augmentation, effectively enriching the semantic content of original queries. These methods prompt LLMs to generate query-related documents (Wang et al., 2023a; Jagerman et al., 2023; Gao et al., 2023), leverage Chain-of-Thought (CoT) reasoning results (Wei et al., 2022b; Trivedi et al., 2023), or utilize specific keywords (Li et al., 2024; Jagerman et al., 2023) to expand queries, thereby enhancing the ranking capabilities of lexical matching based retrieval models (Jagerman et al., 2023; Wang et al., 2023a), dense retrieval models (Wang et al., 2023a), and reranking models (Li et al., 2024). However, these LLM-generated expansions are often directly incorporated into the retrieval process without retraining or adapting the retriever. Consequently, the retriever fails to fully leverage the enriched signals of LLMs, resulting in limited improvements in retrieval performance (Wang et al., 2023a).

Moreover, existing approaches that incorporate LLMs into retrieval systems often train the LLM (Jiang et al., 2025) or the retriever independently (Kim and Baek, 2025), resulting in preference misalignment between the generation and retrieval components. Some works, such as RaFe (Mao et al., 2024), attempt to align LLM rewriting with retrieval signals by using reranker scores as feedback. However, these approaches rely on a separate reranking model rather than incorporating direct training signals from dense retrievers. In contrast, our approach introduces a joint training framework that simultaneously optimizes the LLM and the dense retriever, enabling stronger alignment between the two components to conduct a more effective retrieval result.

3 ExpandR: An LLM Augmented Dense Retriever Method

As illustrated in Figure 1, this section introduces ExpandR, our LLM-augmented dense retrieval

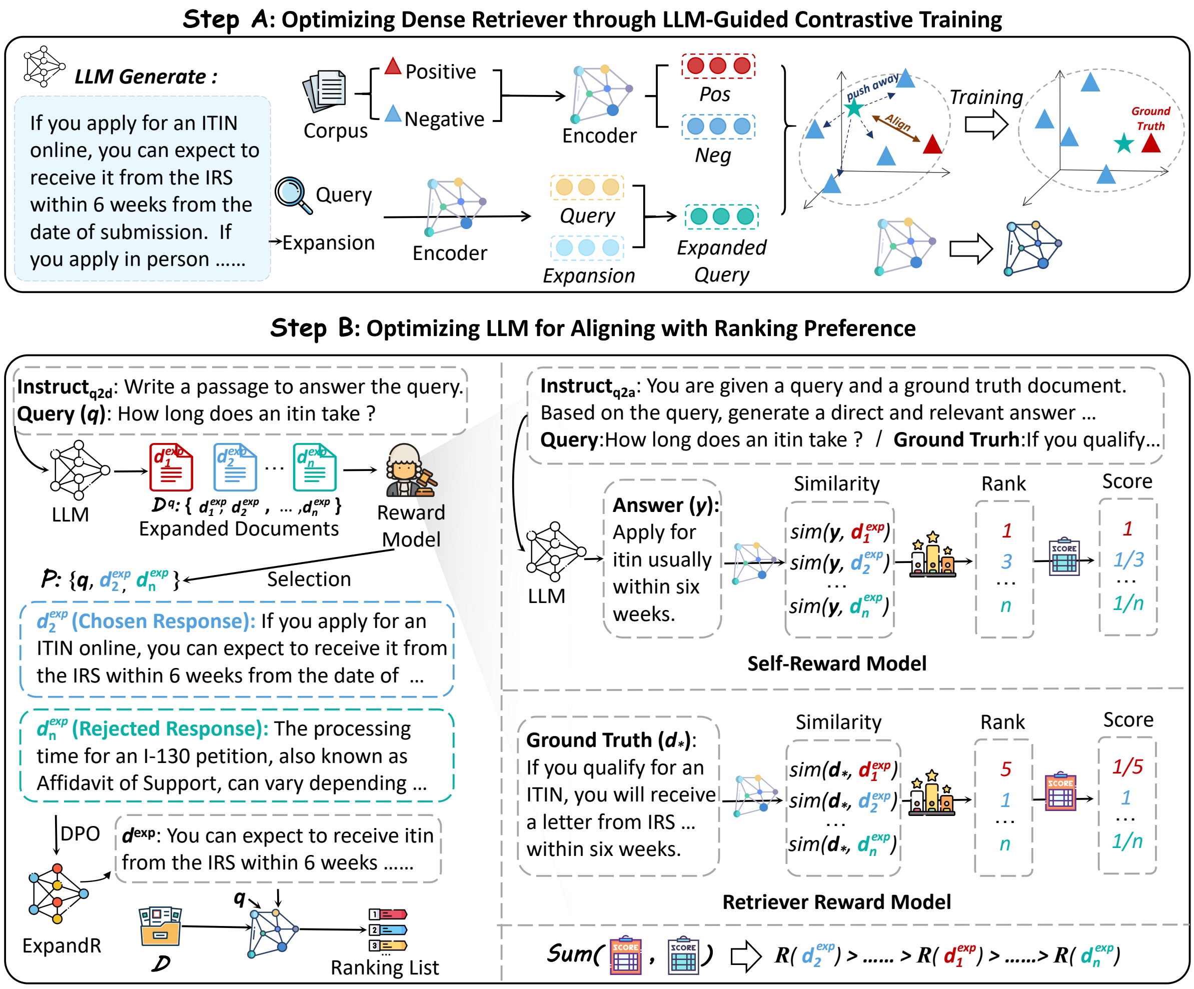


Figure 1: Illustration of Our ExpandR Model. ExpandR optimizes both dense retriever and LLM using the LLM-guided contrastive training method and the ranking preference alignment method.

model that leverages query expansions to improve retrieval performance. We begin by describing the overall architecture of ExpandR (Sec. 3.1). We then present how LLM-generated query expansions are used to guide the training of the dense retriever (Sec. 3.2). Finally, we detail a preference-based optimization strategy for the LLM to generate more effective and tailored query expansions (Sec. 3.3).

3.1 Toward a Framework for LLM-Guided Dense Retrieval

This section illustrates how LLMs can be leveraged to enhance dense retrieval. We first introduce the architecture of a standard dense retriever, and then present our proposed method, ExpandR, which incorporates LLM guidance to improve the retrieval model's effectiveness.

Dense Retrieval. Given a query q and a document collection $\mathcal{D} = \{d_1, ..., d_k\}$, dense retrieval models (Karpukhin et al., 2020; Xiong et al., 2021a; Gao and Callan, 2021) first encode the query q and the i-th document d_i into embeddings $\vec{q}$ and $\vec{d_i}$ using PLMs, such as BERT (Devlin et al., 2019):

$$\vec{q} = \text{BERT}_q(q), \quad \vec{d_i} = \text{BERT}_d(d_i). \tag{1}$$

Then the relevance score $S(q, d_i)$ is calculated to estimate the relevance between q and d_i:

$$S(q, d_i) = \text{sim}(\vec{q}, \vec{d_i}), \tag{2}$$

where sim is the dot product operation. Finally, dense retrieval models conduct a KNN search (Douze et al., 2024) to retrieve the top-ranked documents to satisfy the user needs.

ExpandR. Unlike traditional dense retrieval models (Karpukhin et al., 2020), ExpandR leverages the knowledge encoded in LLM to guide dense retrievers via query expansions d^{exp}, aiming to achieve more accurate retrieval results.

Specifically, we first prompt the LLM $\mathcal{M}$ to generate a query expansion as follows:

$$d^{\text{exp}} = \mathcal{M}(\text{Instruct}_{\text{q2d}}, q), \tag{3}$$

where $\text{Instruct}_{\text{q2d}}$ denotes an instruction prompting the LLM to generate an informative expansion for the input query (Jagerman et al., 2023). We then model the joint probability of retrieving the ground truth document d_* conditioned on the original query q as:

$$P(d_* \mid q; \Phi, \Theta) = P(d_* \mid q, d^{\text{exp}}; \Phi) \cdot P(d^{\text{exp}} \mid q; \Theta), \tag{4}$$

where Φ and Θ represent the parameters of the retriever and the LLM, respectively. This formulation can be rewritten as:

$$\begin{aligned} &\log P(d_* \mid q; \Phi, \Theta) = \\ &\log P(d_* \mid q, d^{\text{exp}}; \Phi) + \log P(d^{\text{exp}} \mid q; \Theta). \end{aligned} \tag{5}$$

Our objective is to jointly optimize the retriever and the LLM–i.e., Φ and Θ–to maximize the above log-likelihood, as described in Section 3.2 and Section 3.3, respectively.

3.2 Optimizing Dense Retriever through LLM-Guided Contrastive Training

To maximize $P(d_* \mid q; \Phi, \Theta)$ by optimizing the retriever parameters Φ, we train the dense retriever using both the original query q and its corresponding expansion d^{exp}:

$$\begin{aligned} &\log P(d_* \mid q; \Phi, \Theta) \\ &= \underbrace{\log P(d_* \mid q, d^{\text{exp}}; \Phi)}_{\text{Optimize w.r.t. } \Phi} + \underbrace{\log P(d^{\text{exp}} \mid q; \Theta)}_{\text{Fixed}}. \end{aligned} \tag{6}$$

To optimize the retriever, we fix Θ and update only Φ by maximizing the retriever-related term:

$$\Phi^* = \arg\max_{\Phi} \log P(d_* \mid q, d^{\text{exp}}; \Phi). \tag{7}$$

To incorporate the knowledge of d^{exp}, we simply average the embeddings of both q and d^{exp} as the final query representation $\vec{q}^{\,\text{exp}}$:

$$\vec{q}^{\,\text{exp}} = \frac{\vec{q} + \vec{d}^{\,\text{exp}}}{2}. \tag{8}$$

Then we treat the expanded query q^{exp} as the new query and compute the similarity score $\text{sim}(q^{\text{exp}}, d)$ between q^{exp} and each candidate document d. The retriever can be contrastively trained using the training loss $\mathcal{L}_{\text{DR}}$:

$$\mathcal{L}_{\text{DR}} = -\log \frac{e^{\text{sim}(q^{\text{exp}}, d_*)}}{e^{\text{sim}(q^{\text{exp}}, d_*)} + \sum_{d_- \in \mathcal{D}^-} e^{\text{sim}(q^{\text{exp}}, d_-)}}, \tag{9}$$

where $\mathcal{D}^-$ represents the set of negative documents, which are sampled from in-batch negatives (Karpukhin et al., 2020).

3.3 Optimizing LLM for Aligning with Ranking Preference

To maximize the probability $P(d_* \mid q; \Phi, \Theta)$, we optimize only the LLM parameters (Θ) while keeping the dense retriever parameters (Φ) fixed.

As shown in Eq. 5, updating Θ alone still affects both terms of the joint probability. Therefore, we optimize Θ as follows:

$$\begin{aligned} \Theta^* = \arg\max_{\Theta} [&\log P(d_* \mid q, d^{\text{exp}}; \Phi) \\ &+ \log P(d^{\text{exp}} \mid q; \Theta)]. \end{aligned} \tag{10}$$

This objective indicates that a well-generated d^{exp} can not only directly increase the likelihood term $\log P(d^{\text{exp}} \mid q; \Theta)$, but also indirectly improve retrieval performance by providing more informative expansions for the term $\log P(d_* \mid q, d^{\text{exp}}; \Phi)$. To realize this dual effect, we optimize the LLM parameters through a reward-driven approach. The optimization process involves two steps: first, we define the reward modeling objective (Eq. 10); then, we train the LLM using the Direct Preference Optimization (DPO) method (Amini et al., 2024).

Reward Modeling. We define a reward function $R(d^{\text{exp}})$ to evaluate each candidate expansion $d^{\text{exp}} \in \mathcal{D}^q$. The reward combines two complementary signals:

$$R(d^{\text{exp}}) = R_{\text{self}}(d^{\text{exp}}) + R_{\text{retriever}}(d^{\text{exp}}), \tag{11}$$

where $R_{\text{self}}(d^{\text{exp}})$ and $R_{\text{retriever}}(d^{\text{exp}})$ represent the self-reward and the retriever reward, respectively.

Self-Reward. To promote the likelihood term $\log P(d^{\text{exp}} \mid q; \Theta)$, we incorporate a self-reward that leverages the LLM's self-consistency. Specifically, we prompt the LLM to generate an answer y according to the query q and the ground-truth document d_*:

$$y = \mathcal{M}(\text{Instruct}_{\text{q2a}}, q, d_*), \tag{12}$$

where $\text{Instruct}_{\text{q2a}}$ guides the LLM to produce an answer y to q. We then treat the answer y as a query and rank the expansion candidates $\mathcal{D}^q$ to compute the self-reward score:

$$R_{\text{self}}(d^{\text{exp}}) = \frac{1}{\text{Rank}(y, d^{\text{exp}})}, \tag{13}$$

where $\text{Rank}(y, d^{\text{exp}})$ denotes the rank of document d^{exp} based on its relevance score $\text{sim}(y, d^{\text{exp}})$. A higher rank indicates stronger semantic similarity and consistency between y and d^{exp}.

Retriever Reward. While the self-reward ensures the semantic plausibility of the candidate expansion d^{exp}, it does not necessarily guarantee its usefulness for retrieval, i.e., contributing to $\log P(d_* \mid q, d^{\text{exp}}; \Phi)$ (Weller et al., 2024). To address this limitation, we incorporate a retriever reward that captures the preferences of the retriever. Specifically, we compute the Mean Reciprocal Rank (MRR) by treating the ground-truth document d_* as a pseudo-query and ranking the expansion candidates $\mathcal{D}^q$:

$$R_{\text{rank}}(d^{\text{exp}}) = \frac{1}{\text{Rank}(d_*, d^{\text{exp}})}, \tag{14}$$

where $\text{Rank}(d_*, d^{\text{exp}})$ denotes the rank of d^{exp} based on the similarity score $\text{sim}(d_*, d^{\text{exp}})$. A higher reward indicates that the expansion is more similar to the ground-truth document, and thus more likely to improve retrieval performance.

LLM Optimization. We fine-tune the LLM $\mathcal{M}$ using preference modeling via DPO. Specifically, we first prompt the LLM to generate a set of expansion candidates $\mathcal{D}^q = \{d_1^{\text{exp}}, \ldots, d_k^{\text{exp}}\}$ for each query q, by sampling with varying temperature:

$$d^{\text{exp}} \sim \mathcal{M}(\text{Instruct}_{\text{q2d}}, q). \tag{15}$$

Then we construct training triples $(q, d_+^{\text{exp}}, d_-^{\text{exp}})$ using the reward model $R(\cdot)$ (Eq. 11):

$$R(d_+^{\text{exp}}) > R(d_-^{\text{exp}}), \tag{16}$$

and follow the DPO method to optimize the LLM $(\mathcal{M})$ using the loss function $\mathcal{L}(\mathcal{M}; \mathcal{M}^{\text{Ref}})$:

$$\begin{aligned} \mathcal{L}(\mathcal{M}; \mathcal{M}^{\text{Ref}}) &= -\mathbb{E}_{(q, d_+^{\text{exp}}, d_-^{\text{exp}}) \sim \mathcal{P}} \Big[\log \sigma \Big(\\ \beta \log \frac{\mathcal{M}(d_+^{\text{exp}} \mid q)}{\mathcal{M}^{\text{Ref}}(d_+^{\text{exp}} \mid q)} &- \beta \log \frac{\mathcal{M}(d_-^{\text{exp}} \mid q)}{\mathcal{M}^{\text{Ref}}(d_-^{\text{exp}} \mid q)} \Big) \Big], \end{aligned} \tag{17}$$

where σ is the sigmoid function, β is a scaling hyperparameter, and $\mathcal{M}^{\text{Ref}}$ is a frozen reference model. The training set $\mathcal{P}$ is composed of preference pairs sampled based on reward scores.

4 Experimental Methodology

In this section, we introduce the datasets, evaluation metrics, baselines, and implementation details used in our experiments.

Dataset. We utilize various datasets for training and evaluation. Data statistics are shown in Table 1. More details on data generation and processing are shown in Appendix A.2.

Training. We use the publicly available E5 dataset (Wang et al., 2024; Springer et al., 2024) to train both the LLMs and dense retrievers. We concentrate on English-based question answering tasks and collect a total of 808,740 queries. From this set, we randomly sample 100,000 queries to construct the DPO data for training LLM, while the remaining queries are used for contrastively training the dense retrieval model. During the construction of DPO preference pairs, we first prompt LLMs to generate documents as query expansions (Wang et al., 2023a). We then filter out queries whose generated documents exhibit low semantic similarity to the original queries. This results in a final dataset comprising 30,000 high-quality queries.

Dataset	Setting	#Query		
		Train	Dev	Test
E5	LLM	27,000	3,000	-
	Retrieval	637,866	70,874	-
MS MARCO	Retrieval	-	-	6,980
Beir	Retrieval	-	-	46,379

Table 1: Statistics of the datasets used in our experiments. The E5 dataset is used for joint training of the LLM and the retriever, while MS MARCO and BEIR are used exclusively for evaluation.

Evaluation. We evaluate retrieval effectiveness using two retrieval benchmarks: MS MARCO (Bajaj et al., 2016) and BEIR (Thakur et al., 2021).

Evaluation Metrics. We use nDCG@10 as the evaluation metric, which is the official evaluation metric of BEIR (Thakur et al., 2021). Statistical significance is tested using a permutation test with $p < 0.05$.

Baselines. We compare our ExpandR model with four representative retrieval models, including BM25 (Robertson et al., 2009), DPR (Karpukhin et al., 2020), CoCondenser (Gao and Callan, 2022), and ANCE (Xiong et al., 2021a).

Then we use different retrievers as backbone models and optimize them using different training strategies. Three encoders as backbone retrievers to examine the generalization ability of our ExpandR, including vanilla BERT (Devlin et al., 2019), Contriever (Izacard et al., 2021), and AnchorDR (Xie et al., 2023). Contriever pretrains PLMs on unlabeled text pairs by encouraging semantically similar sentences to have closer representations in the embedding space. In contrast, AnchorDR leverages the relationships between anchor texts and their linked documents to enhance pretraining. Each retriever is evaluated under three training strategies: (1) **Raw**: directly encoding both queries and documents without fine-tuning; (2) **FT**: standard su-

Task	BM25	DPR	CoCondenser	ANCE	BERT			Contriever			AnchorDR		
					Raw†	FT$^{\diamond}$	ExpandR	Raw†	FT$^{\diamond}$	ExpandR	Raw†	FT$^{\diamond}$	ExpandR
MS MARCO	22.8	17.7	16.2	37.0	0.29	22.68†	23.54†	20.55	32.96†	33.65†	25.66	36.35†	**37.14**†
Trec-COVID	65.6	33.2	40.4	62.1	3.73	19.72†	19.12†	27.45	30.03†	47.98$^{\dagger\diamond}$	51.44	53.71†	**78.85**$^{\dagger\diamond}$
NFCorpus	32.5	18.9	28.9	23.4	2.60	21.02†	23.98$^{\dagger\diamond}$	31.73	32.33	**34.80**$^{\dagger\diamond}$	31.23	31.04	32.13$^{\diamond}$
NQ	32.9	47.4	17.8	42.9	0.40	15.61†	29.64$^{\dagger\diamond}$	25.37	33.72†	50.39$^{\dagger\diamond}$	26.24	40.30†	**55.91**$^{\dagger\diamond}$
HotpotQA	60.3	39.1	34.0	47.1	0.77	16.10†	29.70$^{\dagger\diamond}$	48.07	58.78†	**70.50**$^{\dagger\diamond}$	52.46$^{\diamond}$	47.84	63.40$^{\dagger\diamond}$
FiQA	23.6	11.2	25.1	29.3	0.59	11.16†	15.40$^{\dagger\diamond}$	24.50	26.06†	32.40$^{\dagger\diamond}$	24.04	28.20†	**34.17**$^{\dagger\diamond}$
ArguAna	31.5	17.5	44.4	40.2	8.19	39.36†	37.57†	37.90	53.48†	**55.39**$^{\dagger\diamond}$	29.50	48.51†	49.16†
Touche-2020	**36.7**	13.1	11.7	23.6	0.39	2.82†	5.89$^{\dagger\diamond}$	16.68$^{\diamond}$	10.46	17.38$^{\diamond}$	12.37	13.76†	24.53$^{\dagger\diamond}$
CQADupStack	29.9	15.3	30.9	28.8	1.10	17.10†	16.47†	28.43	31.60†	33.00$^{\dagger\diamond}$	30.30	34.72†	**35.18**†
Quora	78.9	24.8	82.1	84.7	36.29	77.38†	72.04†	83.50	84.98†	84.67†	83.49	**85.06**†	79.34
DBPedia	31.3	26.3	21.5	26.5	1.57	14.08†	23.05$^{\dagger\diamond}$	29.16	36.46†	**42.32**$^{\dagger\diamond}$	33.58	34.55	40.73$^{\dagger\diamond}$
Scidocs	15.8	7.7	13.6	11.3	0.70	6.04†	9.43$^{\dagger\diamond}$	14.91	14.94	**17.85**$^{\dagger\diamond}$	16.57	15.77	16.82$^{\diamond}$
FEVER	75.3	56.2	61.5	68.1	0.24	36.59†	57.49$^{\dagger\diamond}$	68.20	82.49†	**87.07**$^{\dagger\diamond}$	62.98	77.43†	84.57$^{\dagger\diamond}$
Climate-FEVER	21.4	14.8	16.9	19.8	0.61	11.52†	24.63$^{\dagger\diamond}$	15.50	23.04†	29.77$^{\dagger\diamond}$	23.44	26.63†	**31.76**$^{\dagger\diamond}$
Scifact	66.5	31.8	56.1	50.2	2.81	42.35†	46.27$^{\dagger\diamond}$	64.92	68.84†	**69.68**†	59.84	60.51	63.43$^{\dagger\diamond}$
Avg.BEIR14	43.0	25.5	34.6	39.9	4.29	23.63	29.33	36.88	41.94	48.09	38.39	42.72	**49.28**
Avg.All	41.7	25.0	33.4	39.7	4.02	23.57	28.95	35.79	41.34	47.12	37.54	42.29	**48.47**
Best on	1	0	0	0	0	0	0	0	0	**7**	0	1	6

Table 2: Overall Performance of ExpandR. We follow previous work (Izacard et al., 2021) and report the average performance on 14 BEIR tasks (BEIR14) and all tasks (All). **Bold** and underlined scores indicate the best and second-best results. †, ◇ denote significant improvements over the Raw and FT training settings of each retriever.

pervised fine-tuning using query-document triples; and (3) **ExpandR**: it integrates LLM-based query expansion to augment dense retriever and jointly optimizes both LLM and retriever.

Implementation Details. For our query expansion model, we deploy the Meta-LLaMA-3-8B-Instruct (AI@Meta, 2024) as the backbone. The batch size is set to 16, and the learning rate is set to $2e-5$. Optimization is performed using the AdamW optimizer. We employ LoRA (Hu et al., 2022) to efficiently fine-tune the model for 2 epochs. The temperature for the construction of the DPO data varies across $\tau \in \{0.8, 0.9, 1.0, 1.1\}$, with each setting sampled eight times. For the dense retrievers, we utilize three retrievers with different structures: BERT (Devlin et al., 2019), Contriever (Izacard et al., 2021) and AnchorDR (Xie et al., 2023) as the backbone. During training, we set the batch size to 1,024 and the learning rate to $1e-5$, with the model trained for 3 epochs.

5 Evaluation Results

This section presents the overall performance of ExpandR, followed by ablation studies. Then we analyze the semantic distribution of query-document embeddings under different training strategies and evaluate the effectiveness of various reward models. A case study is provided in Appendix A.9.

5.1 Overall Performance

The retrieval performance measured by nDCG@10 across various baselines and training configurations is summarized in Table 2. Additional comparisons with mainstream retriever baselines and extended evaluation results are provided in Appendix A.3 and Appendix A.4, respectively.

As shown in the evaluation results, ExpandR achieves more than 9% improvements over previous retriever models, such as BM25 and ANCE, highlighting its effectiveness. By substituting different retrieval backbone models, ExpandR further demonstrates strong generalization ability, consistently outperforming both zero-shot retrieval (Raw) and standard supervised fine-tuning (FT). Specifically, it achieves an average improvement of 15.6% over Raw and 5.8% over FT across three backbone retrievers on all tasks, validating the benefit of incorporating LLM guidance into dense retrieval.

Notably, ExpandR achieves the best performance on 7 out of 15 tasks when using Contriever, and on 6 tasks with AnchorDR, indicating that its effectiveness holds even with stronger backbone retrievers. The performance gains are particularly pronounced on challenging datasets such as NQ, HotpotQA, and TREC-COVID, where bridging the semantic gap between queries and documents is more difficult. These results illustrate the capability of ExpandR to mitigate the semantic mismatch in complex retrieval scenarios. Additional results using different LLMs as the backbone for query expansion are provided in Appendix A.5, showing consistent improvements and further validating the robustness of ExpandR across model variants.

Model	MARCO	Trec-COVID	NQ	HotpotQA	FiQA	DBPedia	FEVER	Scifact	Avg.
					Contriever				
Query	20.55	27.45	25.37	48.07	24.50	29.16	68.20	64.92	38.53
w/ Retriever Training	32.96	30.03	33.72	58.78	26.06	36.46	82.49	68.84	46.17
ExpandR	**33.65**	47.98	**50.39**	**70.50**	**32.40**	**42.32**	87.07	69.68	**54.25**
w/o LLM Training	33.45	38.64	47.20	66.45	29.74	40.97	85.18	70.55	51.52
w/o Retriever Training	25.20	**59.66**	43.26	65.82	30.12	38.20	82.80	67.74	51.60
w/o Self-Reward	33.05	44.07	47.74	69.62	30.74	42.24	**87.63**	**70.65**	53.21
w/o Retriever Reward	33.47	42.17	49.75	69.12	32.12	40.31	86.52	69.96	52.92
					AnchorDR				
Query	25.66	51.44	26.24	52.46	24.04	33.58	62.98	59.84	42.03
w/ Retriever Training	36.35	53.71	40.30	47.84	28.20	34.55	77.43	60.51	47.36
ExpandR	**37.14**	**78.85**	**55.91**	**63.40**	**34.17**	**40.73**	**84.57**	**63.43**	**57.28**
w/o LLM Training	35.17	70.56	51.24	59.22	29.84	36.11	80.69	61.58	53.05
w/o Retriever Training	29.59	78.50	42.30	57.41	24.91	38.67	79.00	63.40	51.72
w/o Self-Reward	36.56	75.75	54.81	62.74	32.31	40.42	84.41	63.07	56.25
w/o Retriever Reward	37.07	73.75	55.19	61.59	32.97	40.20	82.02	62.47	55.65

Table 3: Ablation Analysis of Key Components in ExpandR on Contriever and AnchorDR. We examine the contributions of LLM training, retriever training, and reward modeling to retrieval performance on 8 important datasets in BEIR. MARCO denotes the MS MARCO dataset.

5.2 Ablation Study

In this subsection, we conduct comprehensive ablation studies under both Contriever and AnchorDR as backbone retrievers to understand the contribution of each component in ExpandR. We evaluate the impact of different reward modeling methods, LLM optimization strategies, and retriever training.

As shown in Table 3, we first include two baselines: "Query" uses raw queries without training, and "w/ Retriever Training" applies contrastive training using raw queries. These settings serve as control groups to isolate the contributions of our LLM optimization and expansion-based retriever training. In both Contriever and AnchorDR backbones, we observe substantial improvements of ExpandR over these baselines, demonstrating that our joint optimization strategy yields significant gains over standard query-only training.

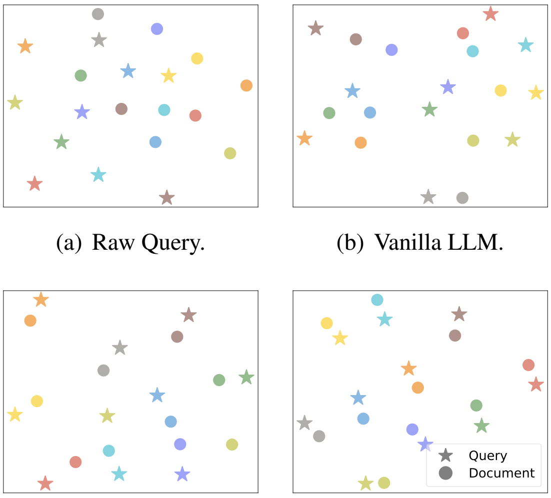
We further assess the role of LLM optimization by removing the DPO training. This results in a 2.73% and 4.23% performance drop on Contriever and AnchorDR, respectively, underscoring the importance of aligning LLM outputs with ranking preferences via preference modeling. Additionally, removing retriever training while retaining LLM optimization significantly impairs performance (2.65 and 5.56 point drops), demonstrating that expansions optimization alone is insufficient unless the retriever is also jointly adapted to leverage them. These findings validate the core motivation of ExpandR that joint optimization of generation and retrieval is key to improving retrieval performance.

Finally, we conduct ablation studies by individually removing the self-reward and retriever reward to assess the impact of each reward modeling strategy during LLM training. We observe performance degradation in both cases, especially on QA benchmarks such as NQ and HotpotQA, demonstrating their complementary benefits in enhancing generation quality and aligning with retriever preferences. Notably, removing the retriever reward results in a slightly larger drop, indicating that retrieval-guided feedback plays a more crucial role in guiding effective query expansion.

5.3 Visualization of Alignment in the Semantic Embedding Space

We visualize the embeddings of queries and documents using T-SNE to investigate how different query expansion strategies and retriever configurations affect their semantic alignment. Specifically, we randomly sample 10 query-document pairs and project their embeddings into a two-dimensional space. Each pair is assigned a unique color, with the query represented by a star and the document by a circle, facilitating a direct visual assessment of semantic proximity under various settings. Throughout this analysis, we employ AnchorDR as the base dense retriever to encode queries and documents.

As shown in Figure 2, when using original queries with the base retriever (Figure 2(a)), we



(a) Raw Query. (b) Vanilla LLM.

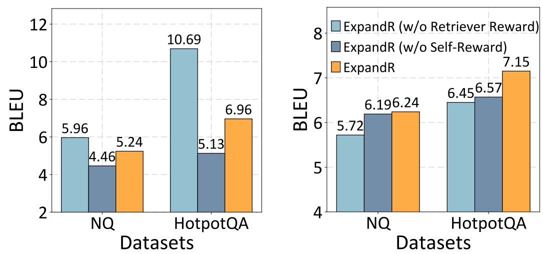
(c) ExpandR (w/o DPO). (d) ExpandR.

Figure 2: Embedding Visualization of Different Models.

observe that query and document embeddings are widely scattered, suggesting a substantial semantic gap between the raw query formulation and its target document. Incorporating expansions generated by a vanilla LLM leads to modest improvements (Figure 2(b)), as some queries shift closer to their corresponding documents. However, the alignment remains inconsistent, and many query-document pairs still appear poorly matched. Fine-tuning the retriever alone results in further improvement (Figure 2(c)), making the embedding space more compact and pulling many expanded queries closer to their paired documents. Nevertheless, the most significant alignment gain is observed when both the query expansion model and the retriever are jointly optimized via preference alignment (Figure 2(d)). In this setting, query-document pairs exhibit significantly tighter and more coherent clustering, suggesting that the combined optimization of the expansion model and the retriever substantially improves semantic consistency and retrieval accuracy. These observations further underscore the importance of jointly aligning both components in dense retrieval systems.

5.4 Effectiveness of Reward Modeling in Optimizing ExpandR

Figure 3 presents an evaluation of the reward model designed in ExpandR, measured by the text similarity between query expansions and either LLM-generated answers or golden documents. We compare three variants: the full model (ExpandR), w/o Retriever Reward, and w/o Self-Reward.



(a) Similarity with Answers. (b) Similarity with Golden Documents.

Figure 3: Effect of Reward Modeling on the Semantic Alignment of Query Expansions.

We first assess the similarity between query expansions and LLM-generated answers (Figure 3(a)). ExpandR w/o Retriever Reward produces expansions most aligned with LLM-generated answers, yielding the highest BLEU score. In contrast, ExpandR w/o Self-Reward achieves the lowest score, indicating that relying solely on the retriever reward is less effective in guiding ExpandR to align with the information in answers, which is particularly important for QA tasks. When the self-reward is incorporated, the BLEU score improves notably, demonstrating its effectiveness in enhancing the factual precision of the expansions.

We then evaluate the similarity between query expansions and ground-truth documents (Figure 3(b)). ExpandR w/o Retriever Reward again performs worst, suggesting that the self-reward alone is insufficient to ensure alignment with golden documents. Conversely, ExpandR w/o Self-Reward performs better, showing the utility of the retriever reward in guiding the model to produce semantically relevant expansions. The full model, integrating both rewards, achieves the highest BLEU score, highlighting the complementary strengths of self-reward and retriever reward in optimizing LLMs to generate high-quality expansions.

6 Conclusion

This paper presents ExpandR, a joint optimization framework that leverages LLM-guided query expansions to enhance retriever training. By jointly training dense retrievers and LLMs, ExpandR improves the effectiveness and compatibility of query expansions within retrieval systems. Experimental results demonstrate that ExpandR consistently boosts performance and offers a new perspective on end-to-end alignment between generative and retrieval components in retrieval pipelines.

Limitations

Despite the effectiveness of ExpandR in improving dense retrieval through LLM-guided query expansions, several limitations remain. First, the quality of expansions is still constrained by the generative capacity of the LLM. If the LLM produces low-quality or biased expansions, the downstream retriever may be misled, even with reward-based supervision. Additionally, although the end-to-end optimization improves alignment between generation and retrieval, it introduces additional computational overhead from both expansion generation and joint training.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *ArXiv preprint*.

AI@Meta. 2024. Llama 3 model card.

Afra Amini, Tim Vieira, and Ryan Cotterell. 2024. Direct preference optimization with an offset. *ArXiv preprint*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. In *Proceedings of NeurIPS*.

Nicholas J Belkin, Robert N Oddy, and Helen M Brooks. 1982. Ask for information retrieval: Part i. background and theory. *Journal of documentation*, 38(2):61–71.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *ArXiv preprint*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

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

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of ACL*, pages 3558–3567.

Yan Fang, Jingtao Zhan, Qingyao Ai, Jiaxin Mao, Weihang Su, Jia Chen, and Yiqun Liu. 2024. Scaling laws for dense retrieval. In *Proceedings of SIGIR*, pages 1339–1349.

Luyu Gao and Jamie Callan. 2021. Condenser: a pre-training architecture for dense retrieval. In *Proceedings of EMNLP*, pages 981–993.

Luyu Gao and Jamie Callan. 2022. Unsupervised corpus aware language model pre-training for dense passage retrieval. In *Proceedings of ACL*, pages 2843–2853.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of ACL*, pages 1762–1777.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, et al. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *ArXiv preprint*.

Peixuan Han, Zhenghao Liu, Zhiyuan Liu, and Chenyan Xiong. 2023. Enhancing dense retrievers' robustness with group-level reweighting. *ArXiv preprint*.

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. In *Proceedings of SIGIR*, pages 113–122.

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

Pengcheng Huang, Yongyu Mu, Yuzhang Wu, Bei Li, Chunyang Xiao, Tong Xiao, and Jingbo Zhu. 2024a. Translate-and-revise: Boosting large language models for constrained translation. In *China National Conference on Chinese Computational Linguistics*, pages 120–139.

Suyuan Huang, Chao Zhang, Yuanyuan Wu, Haoxin Zhang, Yuan Wang, Maolin Wang, Shaosheng Cao, Tong Xu, Xiangyu Zhao, Zengchang Qin, et al. 2024b. Scalingnote: Scaling up retrievers with large language models for real-world dense retrieval. *ArXiv preprint*.

Peter Ingwersen. 1996. Cognitive perspectives of information retrieval interaction: elements of a cognitive ir theory. *Journal of documentation*, 52(1):3–50.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research (TMLR)*.

Rolf Jagerman, Honglei Zhuang, Zhen Qin, Xuanhui Wang, and Michael Bendersky. 2023. Query expansion by prompting large language models. *ArXiv preprint*.

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. *ArXiv preprint*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

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 EMNLP*, pages 6769–6781.

Omar Khattab, Christopher Potts, and Matei Zaharia. 2021. Relevance-guided supervision for openqa with colbert. *Transactions of the association for computational linguistics*, 9:929–944.

Minsang Kim and Seungjun Baek. 2025. Syntriever: How to train your retriever with synthetic data from llms. *ArXiv preprint*.

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 *Proceedings of SIGIR*, pages 2321–2326.

Yizhi Li, Zhenghao Liu, Chenyan Xiong, and Zhiyuan Liu. 2021. More robust dense retrieval with contrastive dual learning. In *Proceedings of SIGIR*, pages 287–296.

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

Xueguang Ma, Xi Victoria Lin, Barlas Oguz, Jimmy Lin, Wen-tau Yih, and Xilun Chen. 2025. Drama: Diverse augmentation from large language models to smaller dense retrievers. *ArXiv preprint*.

Iain Mackie, Shubham Chatterjee, and Jeffrey Dalton. 2023. Generative relevance feedback with large language models. In *Proceedings of SIGIR*, pages 2026–2031.

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 EMNLP Findings*.

Zach Nussbaum, John X Morris, Brandon Duderstadt, and Andriy Mulyar. 2024. Nomic embed: Training a reproducible long context text embedder. *ArXiv preprint*.

Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, Qiaoqiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021. Rocketqav2: A joint training method for dense passage retrieval and passage re-ranking. In *Proceedings of EMNLP*, pages 2825–2835.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, pages 333–389.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024. Replug: Retrieval-augmented black-box language models. In *Proceedings of NAACL*, pages 8371–8384.

Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. 2024. Repetition improves language model embeddings. *ArXiv preprint*.

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*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of ACL*, pages 809–819.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of ACL*, pages 10014–10037.

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*.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving text embeddings with large language models. *ArXiv preprint*.

Liang Wang, Nan Yang, and Furu Wei. 2023a. Query2doc: Query expansion with large language models. In *Proceedings of EMNLP*, pages 9414–9423.

Xiao Wang, Sean MacAvaney, Craig Macdonald, and Iadh Ounis. 2023b. Generative query reformulation for effective adhoc search. *ArXiv preprint*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022.

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

Orion Weller, Kyle Lo, David Wadden, Dawn Lawrie, Benjamin Van Durme, Arman Cohan, and Luca Soldaini. 2024. When do generative query and document expansions fail? a comprehensive study across methods, retrievers, and datasets. In *Proceedings of EACL Findings*, pages 1987–2003.

Yiqing Xie, Xiao Liu, and Chenyan Xiong. 2023. Unsupervised dense retrieval training with web anchors. In *Proceedings of SIGIR*, pages 2476–2480.

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

Wenhan Xiong, Xiang Lorraine Li, Srini Iyer, Jingfei Du, Patrick S. H. Lewis, William Yang Wang, Yashar Mehdad, Scott Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oguz. 2021b. Answering complex open-domain questions with multi-hop dense retrieval. In *Proceedings of ICLR*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *ArXiv preprint*.

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 EMNLP*, pages 2369–2380.

Fanghua Ye, Meng Fang, Shenghui Li, and Emine Yilmaz. 2023. Enhancing conversational search: Large language model-aided informative query rewriting. In *Proceedings of EMNLP Findings*, pages 5985–6006.

Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-shot generative conversational query rewriting. In *Proceedings of SIGIR*, pages 1933–1936.

Shi Yu, Zhenghao Liu, Chenyan Xiong, Tao Feng, and Zhiyuan Liu. 2021. Few-shot conversational dense retrieval. In *Proceedings of SIGIR*, pages 829–838.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of SIGIR*, pages 1503–1512.

Hengran Zhang, Keping Bi, Jiafeng Guo, Xiaojie Sun, Shihao Liu, Daiting Shi, Dawei Yin, and Xueqi Cheng. 2025. Unleashing the power of llms in dense retrieval with query likelihood modeling. *ArXiv preprint*.

A Appendix

A.1 License

The authors of 4 out of the 15 datasets in the BEIR benchmark (NFCorpus, FiQA-2018, Quora, Climate-Fever) and the authors of ELI5 in the E5 dataset do not report the dataset license in the paper or a repository. We summarize the licenses of the remaining datasets as follows.

MS MARCO (MIT License); FEVER, NQ, and DBPedia (CC BY-SA 3.0 license); ArguAna and Touché-2020 (CC BY 4.0 license); CQADupStack and TriviaQA (Apache License 2.0); SciFact (CC BY-NC 2.0 license); SCIDOCS (GNU General Public License v3.0); HotpotQA and SQuAD (CC BY-SA 4.0 license); TREC-COVID (Dataset License Agreement).

All these licenses and agreements permit the use of their data for academic purposes.

A.2 Additional Experimental Details

This subsection outlines the components of the training data and presents the prompt templates used in the experiments.

Training Datasets. Following the setup of Wang et al. (2024), we use the following datasets: ELI5 (sample ratio 0.1) (Fan et al., 2019), HotpotQA (Yang et al., 2018), FEVER (Thorne et al., 2018), MS MARCO passage ranking (sample ratio 0.5) and document ranking (sample ratio 0.2) (Bajaj et al., 2016), NQ (Karpukhin et al., 2020), SQuAD (Karpukhin et al., 2020), and TriviaQA (Karpukhin et al., 2020). In total, we use 808,740 training examples.

Prompt Templates. Table 4 lists all the prompts used in this paper. In each prompt, "query" refers to the input query for which query expansions are generated, while "Related Document" denotes the ground truth document relevant to the original query. We observe that, in general, the model tends to generate introductory phrases such as "Here is a passage to answer the question:" or "This is the answer to the query:". Before using the model outputs as query expansions or answer signals, we first filter out these introductory phrases to ensure cleaner and more precise expansion results.

A.3 Comparison with Mainstream Retrievers

To further contextualize the performance of ExpandR, we compare it with a range of widely used dense retrievers on the BEIR and MS MARCO datasets, as shown in Table 5. The baselines include RocketQA (Ren et al., 2021), BGE-M3-EN (Chen et al., 2024), TAS-B (Hofstätter et al., 2021), Gen-Q, ColBERT (Khattab et al., 2021), E5 (Wang et al., 2022), WebDRO (Han et al., 2023), and Nomic-Embed (Nussbaum et al., 2024), covering both general-purpose and specialized retrieval models.The base retriever of the ExpandR method is AnchorDR.

ExpandR achieves the highest average performance across all datasets (48.5%), consistently outperforming all baselines. Even when excluding MS MARCO—which some retrievers may be specifically optimized for—ExpandR retains its leading position with an average score of 49.3%, suggesting strong generalization across a wide range of domains and task formats.

Among the baselines, E5 and Nomic-Embed stand out as strong retrievers. E5 performs competitively on several QA-style datasets such as MS MARCO and NQ, while Nomic-Embed excels on tasks like ArguAna and HotpotQA. However, both models exhibit noticeable performance drops on other benchmarks—for instance, Nomic-Embed underperforms on MS MARCO and Touche-2020—indicating limitations in generalization. In contrast, ExpandR demonstrates more consistent performance across the board, achieving top-tier results without compromising on robustness. This highlights the robustness and generalizability of our approach across diverse retrieval scenarios.

Query Expansion
Prompt for Q2D: Please write a passage to answer the question: Question: { } Passage:
Question Answering
Prompt for Q2A: You are given a query and a related document. Based on the query, generate a direct and relevant answer using the information in the document. If the query is a statement, expand on it. If it is a question, provide a direct answer. Avoid any extra description or irrelevant content. Query: { } Related Document: { } Answer:

Table 4: Prompt Templates Used in ExpandR. These prompts are used to generate query expansion results and produce the responses to answer the question.

A.4 Evaluating Retrieval Completeness through Recall@100

To more comprehensively assess the retrieval capabilities of ExpandR, we report its performance under the Recall@100 metric on both the BEIR

Task	RocketQA	BGE-M3-EN	TAS-B	Gen-Q	ColBERT	E5	WebDRO	Nomic-Embed	ExpandR
MS MARCO	23.2	35.2	40.8	40.8	40.1	**43.1**	40.6	26.4	37.1
Trec-COVID	67.5	44.6	48.1	61.9	67.7	61.7	78.0	67.1	**78.9**
NFCorpus	29.3	32.7	31.9	31.9	30.5	35.1	31.2	**35.5**	32.1
NQ	59.5	29.8	46.3	35.8	52.4	**60.0**	47.2	51.2	55.9
HotpotQA	35.6	68.3	58.4	53.4	59.3	52.4	57.4	**69.1**	63.4
FiQA	30.2	28.3	30.0	30.8	31.7	37.9	28.4	**37.8**	34.2
ArguAna	45.1	**61.5**	42.9	49.3	23.3	51.4	48.0	54.2	49.2
Touche-2020	24.7	13.5	16.2	18.2	20.2	**28.3**	27.6	19.0	24.5
CQADupStack	19.3	**40.2**	31.4	34.7	35.0	28.3	35.2	49.6	35.2
Quora	31.2	**88.7**	83.5	83.0	85.4	87.9	85.8	88.4	79.3
DBPedia	35.6	19.0	38.4	32.8	39.2	33.8	38.1	39.4	**40.7**
Scidocs	16.5	9.6	14.9	14.3	14.5	19.0	15.3	**19.2**	16.8
FEVER	67.6	64.3	70.0	66.9	77.1	58.2	70.9	60.3	**84.6**
C-FEVER	18.0	18.3	22.8	17.5	18.4	15.4	18.9	27.0	**31.8**
Scifact	56.8	71.5	64.3	64.4	67.1	**73.1**	62.2	71.8	63.4
Avg.BEIR14	38.3	42.2	42.8	42.5	44.4	45.9	46.0	49.2	**49.3**
Av.All	37.3	41.7	42.7	42.4	44.1	45.7	45.6	47.7	**48.5**

Table 5: Performance Comparison of More Mainstream Retriever Baselines on the Beir and MS MARCO Datasets (nDCG@10). The base retriever of the ExpandR method is AnchorDR.

Task	BM25	DPR	CoCondenser	ANCE	BERT			Contriever			AnchorDR		
					Raw	FT	ExpandR	Raw	FT	ExpandR	Raw	FT	ExpandR
MS MARCO	65.8	55.2	58.2	83.8	3.32	67.29	69.06	67.19	82.81	83.64	74.95	84.56	**84.83**
Trec-COVID	**49.8**	21.2	7.0	9.6	0.71	2.05	3.30	3.68	3.19	6.58	10.70	10.67	14.44
NFCorpus	25.0	20.8	29.1	22.3	8.66	21.40	25.79	29.41	15.97	**34.07**	28.72	28.93	30.78
NQ	76.0	88.0	67.9	82.2	2.81	65.82	83.23	77.12	88.18	**94.88**	80.42	89.67	94.30
HotpotQA	74.0	59.1	54.7	58.8	5.97	42.57	60.95	70.45	75.64	**87.33**	65.86	66.89	78.72
FiQA	53.9	34.2	60.3	58.2	4.66	39.45	47.52	56.19	61.04	**70.03**	54.89	61.07	65.44
ArguAna	94.2	75.1	93.0	92.3	45.73	95.45	95.38	90.11	98.43	**99.00**	80.65	96.51	96.80
Touche-2020	**53.8**	30.1	27.1	45.2	1.33	14.30	30.37	37.36	31.52	46.35	39.91	38.30	47.00
CQADupStack	60.6	40.3	60.3	57.1	7.05	42.81	42.78	61.40	65.20	**67.39**	62.41	66.44	66.37
Quora	97.3	47.0	98.5	98.6	70.10	96.96	96.06	98.71	**99.09**	93.55	95.71	98.11	96.15
DBPedia	39.8	34.9	34.8	30.8	3.85	25.92	34.69	45.29	48.22	**54.00**	43.94	43.73	48.83
Scidocs	35.6	21.9	34.1	25.2	5.67	22.55	27.28	35.99	37.10	**40.50**	36.99	35.15	36.78
FEVER	93.1	84.0	89.6	91.1	1.91	78.48	88.69	93.56	95.93	**96.94**	93.65	93.09	95.05
C-FEVER	43.6	39.0	37.0	45.6	4.23	44.01	56.84	44.14	58.56	64.56	60.08	60.25	**64.81**
Scifact	90.8	72.7	91.4	81.4	22.39	80.36	81.47	92.60	94.00	**96.00**	90.77	91.43	93.43
Avg.BEIR14	63.4	48.3	56.1	57.0	13.22	48.01	55.31	59.71	62.29	**67.94**	60.34	62.87	66.35
Avg.All	63.6	47.7	56.2	58.8	12.56	49.29	56.23	60.21	63.66	**68.99**	61.31	64.32	67.58
Best on	2	0	0	0	0	0	0	0	1	10	0	0	2

Table 6: Overall Performance of ExpandR on Recall@100.

and MS MARCO datasets. This metric reflects the model's ability to retrieve a broad set of relevant documents, complementing earlier evaluations based on ranking accuracy. The results are presented in Table 6.

Across all retriever backbones, ExpandR consistently achieves the highest Recall@100 scores, surpassing both the original query (Raw) and supervised retriever (FT) baselines. The improvements are particularly notable on complex multi-hop and fact-seeking datasets such as NQ, HotpotQA, and FEVER, where purely lexical signals are often insufficient for comprehensive retrieval.

These findings suggest that ExpandR not only improves ranking precision but also significantly enhances semantic recall, demonstrating its ability to uncover a wider range of relevant documents. This further validates the robustness and general applicability of our LLM-augmented strategy across diverse retrieval scenarios.

A.5 Robustness under Different LLM Backbones

To examine the robustness of ExpandR across different language model backbones, we replace the LLM used for query expansion with Qwen2.5-7B-Instruct (Yang et al., 2024), a high-quality Chinese-English bilingual model trained with instruction tuning. We keep Contriever as the base retriever. The results are shown in Table 7.

Task	Contriever		
	Raw	FT	ExpandR
MS MARCO	20.55	32.96	**33.32**
Trec-COVID	27.45	30.03	**48.18**
NFCorpus	31.73	32.33	**34.58**
NQ	25.37	33.72	**50.86**
HotpotQA	48.07	58.78	**70.04**
FiQA	24.50	26.06	**31.98**
ArguAna	37.90	53.48	**55.15**
Touche-2020	16.68	10.46	**18.09**
CQADupStack	28.43	31.60	**32.95**
Quora	83.50	**84.98**	84.58
DBPedia	29.16	36.46	**41.47**
Scidocs	14.91	14.94	**17.48**
FEVER	68.20	82.49	**87.21**
C-FEVER	15.50	23.04	**30.50**
Scifact	64.92	68.84	**70.00**
Avg.BEIR14	36.88	41.94	**48.08**
Avg.All	35.79	41.34	**47.09**
Best on	0	1	**14**

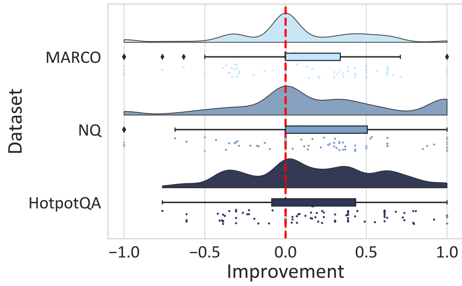
Table 7: Extended Comparison Results under Qwen2.5-7B-Instruct (nDCG@10). The basic retriever in this experiment is Contriever.

The results show that ExpandR consistently outperforms both the original query baseline (Raw) and the supervised retriever trained with raw queries (FT), achieving the best performance on 14 out of 15 datasets. The performance trend closely mirrors that observed in our original experiments using LLaMA, indicating that the improvements are not tied to a specific LLM architecture. Instead, ExpandR captures a generally effective joint optimization strategy that transfers well across different language models.

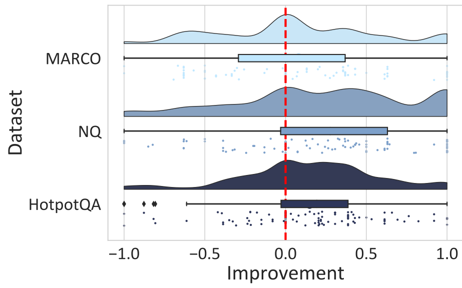
A.6 Query Expansion Quality of ExpandR

This section evaluates the quality of query expansion of ExpandR. As shown in Figure 4, we randomly select 100 samples from each dataset to assess the improvement in retrieval performance before and after applying ExpandR.

Overall, the evaluation results demonstrate that ExpandR consistently improves retrieval performance in both unsupervised (Figure 4(a)) and supervised (Figure 4(b)) settings. However, for the MS MARCO dataset, ExpandR demonstrates limited effectiveness in the supervised setting. This can be attributed to the fact that MS MARCO provides higher-quality training signals, allowing the dense retriever to learn sufficient matching signals from relevance labels. In contrast, ExpandR leads to more substantial performance improvements on the NQ and HotpotQA datasets. This indicates that



(a) Unsupervised Dense Retriever.



(b) Supervised Dense Retriever.

Figure 4: Improvements of ExpandR in Both Unsupervised and Supervised Dense Retrievers. We plot the change of nDCG@10 scores before and after the query expansion using our ExpandR model.

ExpandR provides essential matching signals for dense retrievers, particularly in retrieval scenarios where high-quality training signals are scarce.

A.7 Generalization Analysis of Ranking-Aligned LLM Expansions

To examine the generalizability of our ranking-aligned query expansions beyond the retriever used during training, we evaluate ExpandR under two structurally distinct dense retrievers—AnchorDR and BGE-large-1.5—while keeping the reward signals derived from Contriever fixed.

As shown in Table 8, the results show that retrieval using expansions generated by ExpandR consistently yields better performance than using either the original queries or expansions produced by a vanilla LLM, across both retrievers. Although the LLM is optimized using reward signals from Contriever, it achieves strong performance under both AnchorDR and BGE, obtaining the best results on 12 out of 15 datasets in each setting. Notably, even on BGE—an already highly effective retriever—ExpandR still achieves further gains, indicating that the learned expansions do not simply overfit to the behavior of a specific model, but instead capture a transferable ranking preference that

Task	AnchorDR			BGE-large-1.5		
	Query	Vanilla LLM	ExpandR	Query	Vanilla LLM	ExpandR
MS MARCO	25.7	28.9	**29.4**	**42.0**	39.4	40.3
Trec-COVID	51.4	**77.9**	77.1	64.5	77.8	**78.5**
NFCorpus	31.2	31.3	**31.4**	36.8	37.2	**39.3**
NQ	26.2	39.2	**43.0**	51.7	59.6	**60.8**
HotpotQA	52.5	58.0	**59.3**	74.3	75.2	**76.7**
FiQA	24.0	24.9	**25.4**	44.3	44.3	**46.2**
ArguAna	**29.5**	28.0	28.2	**63.5**	61.6	62.6
Touche-2020	12.4	23.5	**25.6**	24.2	25.3	**26.3**
CQADupStack	30.3	31.1	**31.6**	41.7	42.2	**42.6**
Quora	**83.5**	63.2	66.4	**89.0**	87.9	88.0
DBPedia	33.6	38.8	**39.3**	42.1	45.1	**45.2**
Scidocs	16.6	16.9	**17.0**	20.9	22.9	**23.7**
FEVER	63.0	77.5	**79.7**	84.6	86.5	**88.6**
C-FEVER	23.4	29.7	**30.0**	28.4	30.6	**31.7**
Scifact	59.8	62.4	**63.2**	73.5	75.1	**75.3**
Avg.BEIR14	38.4	43.3	**44.1**	52.8	55.1	**56.1**
Avg.All	37.5	42.4	**43.1**	52.1	54.0	**55.1**
Best on	2	1	12	3	0	12

Table 8: Cross-Retriever Evaluation of Ranking-Aligned Expansions (nDCG@10).

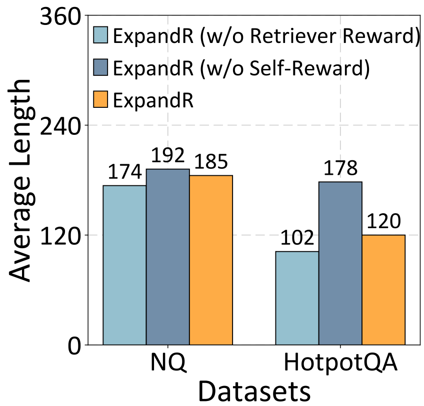


Figure 5: Average Length of Query Expansions Generated by Different Models.

generalizes across different retrieval architectures.

A.8 More Insights into the Self-Reward

While the primary purpose of introducing the self-reward is to enhance the semantic relevance between the generated expansions and the gold answer, we observe that it also serves as an effective regularizer for controlling generation quality. Specifically, we compare the average lengths of the expansions produced by three variants of our model. As shown in Figure 5, removing the self-reward leads to significantly longer generations, which are not necessarily more informative and may introduce hallucinated or off-topic content—a known issue in preference-based tuning methods such as DPO.

With the self-consistency signal in place, the model generates shorter and more focused expansions. To further assess the semantic faithfulness of these generations, we conduct a natural language inference (NLI) based entailment evaluation. As shown in Table 9, although removing the self-reward increases the average length, it results in lower entailment scores, suggesting reduced semantic alignment with the gold answer. In contrast, the full model—trained with both the retriever-based and self-rewards—achieves the highest entailment scores while keeping the generation length moderate, indicating a better balance between informativeness and faithfulness.

These results suggest that the self-reward not only enhances $\log P(d^{\text{exp}} \mid q; \Theta)$, but also implicitly constrains the LLM from over-generating, thereby mitigating hallucination and improving the overall quality of the query expansions during DPO training.

A.9 Case Study

To further demonstrate the effectiveness of ExpandR, we conduct a case study by randomly sampling a query from the evaluation dataset. We then compare retrieval performance using the raw queries, expanded queries by vanilla LLM, and expanded queries by ExpandR.

As shown in Table 10, query expansion significantly improves retrieval effectiveness over using the raw query, with both LLM-generated variants achieving higher nDCG@10. While the vanilla LLM introduces relevant terms such as "temperature" and "humidity", its expansions are often

Model	NQ		HotpotQA	
	NLI Score	Avg. Length	NLI Score	Avg. Length
Vanilla LLM	6.44	221.76	16.38	129.63
ExpandR (w/o Retriever Reward)	8.12	**174.20**	17.81	**102.60**
ExpandR (w/o Self-Reward)	6.65	192.76	13.75	178.29
ExpandR	**8.64**	185.11	**18.67**	120.52

Table 9: Comparison of NLI Entailment Scores and Average Lengths of Extensions Generated by Different Models.

Query: How does the coronavirus respond to changes in the weather?
Golden Pagssage: Epidemics ... occur during the winter months. ... Two major contributing factors are the changes in environmental parameters and human behavior. Studies have revealed the effect of temperature and humidity on respiratory virus stability and transmission rates. More recent research highlights the importance of the environmental factors, especially temperature and humidity ...

Raw Query

nDCG@10: 22.01%
Original Query: How does the coronavirus respond to changes in the weather?

Vanilla LLM

nDCG@10: 76.63%
Expanded Query: The coronavirus, like many other viruses, has been observed to respond to changes in the weather by experiencing fluctuations in transmission and spread. ... the virus tends to thrive in environments with high humidity, typically above 40%, and a temperature range of 37°C to 46°C. ... studies have found that the virus can survive on surfaces for longer periods at lower temperatures and humidity levels, ...

ExpandR

nDCG@10: 100.00%
Expanded Query: The coronavirus responds to changes in the weather by adapting its transmission and spread patterns. This is because temperature, humidity, and other environmental factors can affect the stability and survival of the virus on surfaces, ... research suggests that the virus may thrive in cooler and more humid environments, ... such as air circulation, ventilation, and human behavior.

Table 10: Case Study. All experiments are conducted based on the Contriever model under the zero-shot setting. To facilitate evaluation, we highlight the potential matching phrases between the golden passage and both the original and expanded queries. Different colors are used to annotate these matched phrases for each method: Green for Direct Retrieval, Red for Vanilla LLM, and Blue for ExpandR.

verbose and include redundant or inconsistent content (e.g., conflicting temperature ranges). This reflects a lack of alignment between generation and retrieval utility.

In contrast, ExpandR produces expansions that are more concise and semantically aligned with the golden passage, incorporating key concepts such as “human behavior”, “environmental factors”, and “virus transmission”. These expansions better match the relevance signals favored by the retriever, leading to improved ranking performance. This example illustrates how preference-guided fine-tuning in ExpandR enables the LLM to generate expansions that are both informative and behaviorally aligned with the retrieval model.