tRAG: Term-Level Retrieval-Augmented Generation for Domain-Adaptive Retrieval

Dohyeon Lee[1], **Jongyoon Kim**[2], **Jihyuk Kim**[3], **Seung-won Hwang**[12*], **Joonsuk Park**[4,5,6†]
Computer Science and Engineering, Seoul National University[1],
Interdisciplinary Program in Artificial Intelligence, Seoul National University[2],
LG AI Research[3],
NAVER AI Lab[4], NAVER Cloud[5], University of Richmond[6]
{waylight3, john.jongyoon.kim, seungwonh}@snu.ac.kr
jihyuk.kim@lgresearch.ai, park@joonsuk.org

Abstract

Neural retrieval models have emerged as an effective tool for information retrieval, but their performance suffers when there is a domain shift between training and test data distributions. Recent work aims to construct pseudo-training data for the target domain by generating domain-adapted pseudo-queries using large language models (LLMs). However, we identifies that LLMs exhibit a "seen term bias" where the generated pseudo-queries fail to include relevant "unseen" terms as expected for domain adaptation purposes. To address this limitation, we propose to improve the term recall of unseen query terms, by using term-level Retrieval-Augmented Generation (tRAG). Specifically, unlike existing document-level RAG, we propose to generate domain-specific keywords from all documents in the corpus, including those unseen in any individual document. To filter hallucination, generated keywords are retrieved and reranked, leveraging relevance feedback from both retrievers and LLMs. Experiments on the BEIR benchmark show tRAG significantly improves recall for unseen terms by 10.6% and outperforms LLM and retrieval-augmented generation baselines on overall retrieval performance.

1 Introduction

In information retrieval (IR), fine-tuning neural retrieval models (Karpukhin et al., 2020) has been effective, where an encoder learns to project each document d to a dense vector close to that of a relevant query q. For domains without (q, d) pairs available for fine-tuning, pseudo-query generation (PQG), such as GPL (Wang et al., 2022), has been proposed, where a pseudo-query $\tilde{q}$ for d is generated using large language models (LLMs), as demonstrated in Figure 1(a). Based on the assumption that d shares similar terms with q, PQG has shown strong performance improvements, as $\tilde{q}$ from d also becomes similar to q.

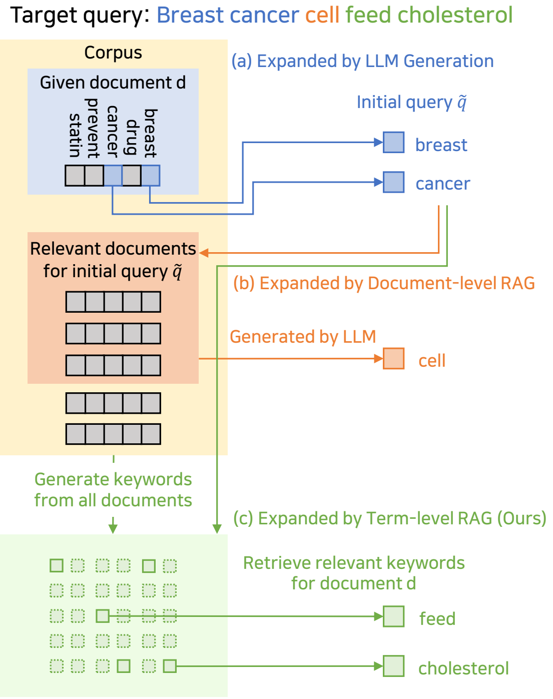
Target query: Breast cancer cell feed cholesterol
Corpus
Given document d
breast
drug
cancer
prevent
statin
(a) Expanded by LLM Generation
Initial query $\tilde{q}$
breast
cancer
Relevant documents for initial query $\tilde{q}$
(b) Expanded by Document-level RAG
Generated by LLM
cell
Generate keywords from all documents
(c) Expanded by Term-level RAG (Ours)
Retrieve relevant keywords for document d
feed
cholesterol

Figure 1: Visualization of how terms that appear in gold queries are added to pseudo-queries. Some terms come from the input document (a) or documents retrieved from the target corpus (b). To add terms in gold queries that do not appear in either types of documents, our method generates and ranks keywords by term-level retrieval (c).

However, as we show in §3.1, existing PQG models struggle to generate terms that are not in d—i.e., *out-of-document* terms—despite their frequent appearance in real queries. For instance, in the NFCorpus dataset of the BEIR benchmark, nearly 90% of generated pseudo query terms are *in-document*. We refer to this issue as **seen term bias**.

To address this issue, we present **<u>t</u>**erm-level **<u>R</u>**etrieval-**<u>A</u>**ugmented **<u>G</u>**eneration (**<u>tRAG</u>**). Unlike existing RAG (Lewis et al., 2020), which retrieves only raw documents from the corpus (see Figure 1(b)), for adding missing query terms in the

[*] Corresponding Authors

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 6566–6578
April 29 - May 4, 2025 ©2025 Association for Computational Linguistics

prompt, or document-level RAG, our approach can generate *out-of-corpus* terms. Existing RAG due to a limited context window size, suffers from low recall for q terms during generation. Even when the window size is sufficient, the problem persists, widely known as the lost-in-the-middle (Liu et al., 2024) bias. tRAG is distinguished from RAG as follows:

First, tRAG uses **generated term as retrieval unit**, rather than the documents themselves, as retrieval units for a closer alignment with query terms. The inspiration comes from the idea to condense each document into centroid keywords, a technique shown to be effective by Wang et al. (2021b) and Tang et al. (2021) for approximating query terms. This approach provides more relevant terms to LLM, significantly improving recall on q terms.

Second, tRAG introduces **collective verification** for a robust generation of *out-of-corpus* terms. While many q terms do not appear in the corpus (e.g., 20% in SciDocs), generating *out-of-corpus* terms that are relevant to the given query is challenging. To address this, standard verification (Weng et al., 2023) assesses the relevance of generated terms to the input d. However, an individual d may lack recall on q terms, resulting in false negatives. To leverage keywords from other documents, we extend it to collectively verify keywords across the entire corpus by reformulating the verification as a ranking task. Finally, the verified, high-recall keywords will be integrated into the LLM prompt, enhancing $\tilde{q}$ to better encompass q terms.

To address the seen term bias, we evaluate models on BEIR benchmark. Results show that tRAG improves term recall of out-of-document and out-of-corpus by +10.6%pt, significantly outperforming GPL and RAG approaches by +1.8%pt and +1.3%pt in retrieval performance (nDCG@10), respectively.

Our contributions are twofold:

1. We unveil the failure of LLMs to generate unseen query terms within a given document, as evidenced by the seen term bias, despite their prevalence in real-world scenarios.

2. We propose tRAG, which augments out-of-document and out-of-corpus query terms with high recall by providing a condensed domain-specific keyword.

2 Related Work

Sharing the goal of predicting gold query terms, we compare two baseline approaches, standard LLM-based query generation (§2.1) and RAG (§2.2), differing in vocabulary referred to for the query term prediction. In addition to LLMs' internal knowledge, the former relies on *in-document vocabulary* and the latter additionally employs *in-corpus vocabulary* by retrieving related documents from the corpus. Our method is distinguished in that we further include *out-of-corpus terms* (§2.3).

2.1 Query Generation from Document

This category aims at utilizing terms from the given document, in conjunction with LLM's internal knowledge, to construct pseudo-queries relevant to the document. This approach has been exemplified by methods such as doc2query and document expansion (Gao et al., 2023; Lei et al., 2024), which automatically generate queries that documents are likely to answer. GPL (Wang et al., 2022) enhances this alignment by evaluating relevance score of each pair using a cross-encoder, thereby facilitating a more accurate document retrieval process.

Leveraging LLM's massive internal knowledge, this has been expected to successfully generate gold query terms, irrespective of their presence in the given document. However, we found that the generated queries are significantly biased toward the seen terms in the input document, failing to generate unseen yet relevant query terms, as highlighted by the seen term bias discussed in the introduction.

2.2 Query Expansion from RAG

To broaden the scope of a query beyond the terms found in a specific document, d, methods like CSQE (Lei et al., 2024) use RAG to incorporate terms from related documents in the LLM context as if they are seen. These methods extract keywords that are absent from the initial query but present in the top retrieved documents, thus enriching the query with new terms not originally considered.

Central to RAG is to augment missing query terms with high recall. However, in practice, the scope of these new terms remains limited to just a handful of documents or, at best, to terms appearing in the corpus.

2.3 Our Distinction: Term-level Retrieval, Generate-then-Rank

The critical challenge is the inability of existing methods to effectively capture and incorporate unseen terms, which are crucial for accurately responding to domain-specific queries. These unseen terms are present, neither in the given document nor in the corpus, yet they frequently appear in real-world queries. Our work is distinguished to retrieve at term-level for better domain recall, based on which generate and rank unseen terms from the entire corpus.

3 Proposed Method: tRAG

The pseudo-query generation task aims to generate a relevant query q for each d in the target corpus, to construct a pseudo-training dataset for the target domain via the relevant q-d pairs. A key limitation of existing LLM-based approaches is the seen term bias, where the generated q is biased to the seen terms in d, and unseen gold query terms are often excluded in q. To overcome this limitation, we refine the initially generated q by augmenting unseen key terms via RAG.

In the subsequent sections, we first discuss the seen term bias of PQG models (§3.1) and the standard RAG approach and its limitations, as our baseline, which employs raw documents as the retrieval unit (§3.2). Then, we present our solution, term-level RAG, proposing keywords as a better alternative for the retrieval unit (§3.3). Finally, we describe our generation-then-ranking approach, which enhances the method's robustness to unseen terms (§3.4).

3.1 Motivation: Seen Term Bias of PQG

Our study on BEIR benchmark reveals a significant challenge where many query terms do not appear in the corresponding documents. This section provides an in-depth analysis of our findings for current PQG-based approaches.

Analysis: Challenging set in BEIR Datasets Our evaluation of BEIR benchmark indicates that a substantial proportion of query terms do not appear in their corresponding documents. As depicted in Figure 2, the ratio of terms is categorized into three groups: out-of-corpus terms, out-of-document terms, and seen terms. On average, approximately 60% of query terms do not appear in the corresponding document, and about 6% are entirely absent from the overall corpus. To rigorously test our models, we selected the datasets with the highest ratios of out-of-corpus terms (NFCorpus, SciFact, SciDocs, FiQA) to form a challenging subset, named BEIR$_{\text{OOC}}$. Additionally, we chose two datasets with low out-of-corpus ratios (Robust04, Trec-Covid) for comparison purposes.

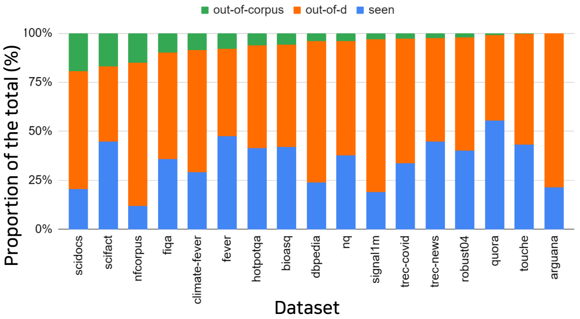

Figure 2: Ratio of query terms categorized by out-of-corpus, out-of-document, and seen terms across the BEIR benchmark.

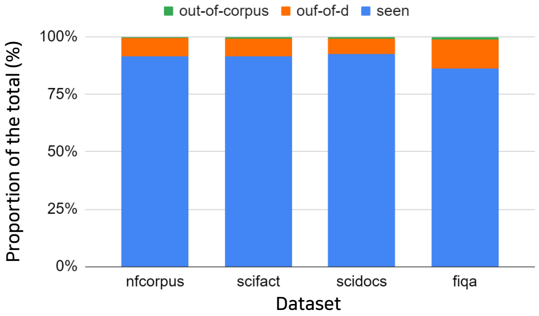

Figure 3: Ratio of generated query terms categorized by out-of-corpus, out-of-document, and seen terms in the challenging subset.

Task-dedicated PQG Our analysis of the challenging subset reveals a significant limitation in current PQG-based approaches like GPL. These models are pretrained to generate terms they have encountered during training, leading to a strong bias towards seen terms. This bias is clearly shown in Figure 3, which displays the distribution of terms in the challenging subset. About 90% of the terms generated by these models come from the corresponding document, highlighting the inherent seen term bias in PQG models. As a result, the recall for unseen terms is low, reducing the effectiveness of these models in handling queries with terms not present in the training data or document corpus.

Seen Term Bias in LLM-based PQG Our exploration extends to evaluating seen term bias of large

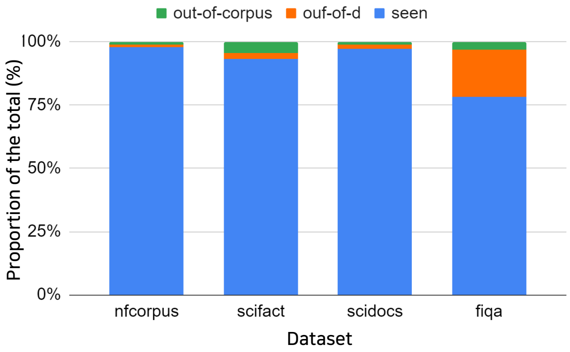

Figure 4: Ratio of LLM-generated query terms categorized by out-of-corpus, out-of-document, and seen terms in the challenging subset.

language models, such as GPT-3[1], on the same challenging subset. Despite the advanced architecture and extensive parameterization of LLMs, our findings indicate that seen term bias persists. As shown in Figure 4, a significant proportion of the terms generated by GPT-3 are seen. Despite the model's large parameter size, about 80% to 90% of the generated terms are found within the corresponding documents. Moreover, for the NFCorpus and SciDocs datasets, this proportion is even higher than that observed in smaller models. This result illustrates that even models with large capacities struggle with capturing unseen terms effectively.

3.2 Baseline: Document-level RAG

In the document-level RAG, the retrieval target is documents that are relevant to a given query. In query generation, RAG can be employed to enlarge the coverage of the LLM context by generating a summarized view of relevant documents. Specifically, given a query q and a target corpus D, standard document-level RAG retrieves the top-n most relevant documents D_q:

$$D_q = \{d_1, d_2, \cdots, d_n\} \subset D, \tag{1}$$

where d_i indicates the i-th relevant document for the given query q.

Following CSQE (Lei et al., 2024), we utilize the CSQE prompt to instruct the LLM to summarize the set of retrieved documents D_q conditioned by the query q, by leveraging the strong relevance-assessing capability of LLM through one-shot prompting. The summary of D_q, denoted by s_{D_q}, aims to capture the key information from the retrieved documents that is relevant to answering the query. More importantly, s_{D_q} can potentially augment the query with missing relevant terms.

However, due to the two key challenges, using raw documents for retrieval is limited in augmenting unseen query terms, which contradicts the objective of RAG: (1) The limited window size of LLMs allows only a few documents to be considered (e.g., $n = 3$ in our experiments), resulting in low recall for gold query terms. (2) Making matters worse, it is impossible to augment query terms if they do not appear in any of the documents. In the subsequent sections, we present our solutions to address both challenges.

3.3 Term-level Retrieval

Our first contribution is enhancing term recall by addressing the first challenge. In contrast to document retrieval, we condense the corpus of raw documents into a set of keyword terms, inspired by the effectiveness of using centroid semantics of documents in improving document representation (Wang et al., 2021b) and query representation (Tang et al., 2021) for retrieval.

Formally, we generate keywords from each document, merge them to build a keyword set K_D, and aim to retrieve relevant keywords instead of raw documents. That is, we replace D_q by K_q:

$$K_q = \{k_1, k_2, \cdots, k_m\} \subset K_D \tag{2}$$

where k_i denotes i-th relevant keyword for the given query q. The maximum value of m, which indicates the number of retrieved keywords, is much larger than n since the length of a keyword is significantly shorter than that of a document ($|k_i| \ll |d_j|$). Therefore, K_q can include more relevant terms for RAG, producing high recall for query terms.

3.4 Generate-then-Rank

Our goal for addressing the second challenge is to realize the potential of K_q by predicting query terms via keywords, including ones that do not appear in the corpus. To achieve this, we propose a two-step process: generation and ranking, targeting two sub-goals of generating unseen keywords in d and filtering relevant ones to d, respectively, elaborated below.

Keyword Generation Tackling the seen term bias of q towards terms in d, our contribution is

[1] We follow the setting of InPars (Bonifacio et al., 2022) for the query generation.

to generate unseen terms for enhanced overall recall on gold query terms of d.

To this end, we provide LLM[2] with few-shot demonstrations that can encourage the LLM to generate unseen keywords. Specifically, each demonstration consists of a randomly selected document and keywords. These keywords are chosen from MS-MARCO test query terms that do not appear in the selected document. The detailed prompt is shown in Appendix A.2. Guided by the few-shot demonstrations, LLM generates a list of keywords K_d for each document d in the corpus D.

Though K_d offers missing query terms in d, it may include terms that are non-relevant to d or even to the target domain. This is because LLM often hallucinates, especially when asked to generalize beyond the knowledge given in the prompt. To tackle the hallucination, we proceed to verify K_d, as discussed below.

Keyword Ranking Given the generated K_d from d, the standard self-verification approach (Weng et al., 2023) can be implemented by inversely verifying the relevance of K_d to d. While preserving relevance, the verified keywords of d are constrained by a few keywords in K_d. Extending the standard verification, our distinction is collective verification, where we collectively verify all keywords across the entire corpus, beyond K_d, to leverage useful keywords generated from other documents.

Specifically, we first merge all keywords in the corpus, producing a corpus-level keyword set K_D:

$$K_D = \bigcup_{d \in D} K_d. \tag{3}$$

We exclude any duplicate keywords from K_D. By using keywords obtained from related documents in the corpus, K_D significantly improves recall on gold query terms of d, compared to K_d.

For each document d, we then collectively verify all keywords in K_D, instead of K_d, regarding their relevance to d. To expedite verification, we first retrieve the top-100 relevant keywords from K_D using q as a query and an existing dense retriever[3]. Then, we leverage the LLM to verify them, by prompting it to filter keywords that are relevant to d and q. We utilize the same LLM for generation and ranking because using the most powerful model in both cases shows the best performance in our experiment. The detailed prompt is presented in Appendix A.3. As a result, the retained keywords, denoted by K_q, improve both relevance to d and recall on unseen query terms.

Query Refinement For refinement, we instruct the LLM to refine the query q by referring to all the given and the augmented vocabularies. These include d, s_{D_q}, and K_q, as in-document, in-corpus, and out-of-corpus vocabulary, respectively. The detailed prompt is shown in Appendix A.4.

When designing the prompt, while placing the summary in a separate field from q, the keywords are concatenated with q as if the keywords are part of the query itself. Empirically, we have found that this encourages the LLM to inject more keywords into the refined query. On the other hand, the LLM often disregards some of the keywords when they are provided separately, irrespective of their relevance. Through the prompt, as an output, we obtain a refined version of the query q, denoted as $\tilde{q}$. Note that tRAG, as a refinement technique, can be integrated with any generation method targeting the initial query q. Examples of refined queries are shown in Appendix A.

For training the retriever, we follow the same procedure as in GPL, but we use $\tilde{q}$ instead of q as it is a better alternative. Specifically, we replace all pseudo queries from GPL with our refined ones. Note that our method only refines the training queries, so inference remains unchanged and incurs no additional cost.

4 Experimental Setup

4.1 Datasets and Evaluation Metrics

As discussed in §3.1, we evaluate our method using four datasets from BEIR benchmark (Thakur et al., 2021) called BEIR$_{\text{OOC}}$: NFCorpus (Boteva et al., 2016), SciFact (Wadden et al., 2020), SciDocs (Cohan et al., 2020), and FiQA (Maia et al., 2018), where the unseen term ratio is greater than 5.0%. To observe the effectiveness in datasets with fewer unseen query terms, we also included Robust04 (Voorhees, 2004) and Trec-Covid (Wang et al., 2020; Voorhees et al., 2020) which have lower than 5.0% unseen (out-of-corpus) term ratio. The detailed statistics are presented in Appendix A.5.

We employed nDCG@10 (Järvelin and Kekäläi-

[2] We use GPT-3.5-Turbo in our experiment as it is the most powerful LLM available within our budget.

[3] We used sentence-transformers/msmarco-distilbert-base-v2 for the keyword retrieval.

nen, 2002) as the evaluation metric, following conventions, which is widely accepted for assessing the quality of the top-10 retrieved documents. We report a single run result for all experiments.

4.2 Baselines

InPars InPars (Bonifacio et al., 2022) employs larger models, such as GPT-3, to generate more diverse and contextually rich pseudo queries. In our setting, we generate queries using InPars and then train the retrieval model with the GPL process for a fair comparison.

GPL GPL (Wang et al., 2021a) generates pseudo queries and labels using the DocT5Query (Nogueira et al., 2019) query generator and retriever with cross-encoders pre-trained on the MS-MARCO dataset (Bajaj et al., 2016).

Contriever Contriever (Izacard et al., 2021) is trained without the use of annotated data, enabling it to generate high-quality embeddings by learning directly from the structure of the corpus.

DRAGON DRAGON (Lin et al., 2023) is a dense retrieval model that improves performance in both supervised and zero-shot settings by using a novel data augmentation approach.

RAG We compared our approach to the standard RAG method by removing K_q from our prompt for refinement, using only s_{D_q} for refining q.

4.3 Implementation Detail

Following GPL (Wang et al., 2021a), we started with the distilbert-base-uncased checkpoint, which was pre-trained on the MS-MARCO dataset as an in-domain corpus. We use GPT-3.5-Turbo to generate and rank keywords and the CSQE prompt to summarize retrieved documents. We use 3 examples for few-shot prompting, matching the number used by our baseline, InPars, to ensure fair evaluations. We constructed the input documents by concatenating the document titles and bodies and truncating the concatenated sequences longer than 256 tokens. The queries were truncated to a maximum length of 64 tokens. The training process was performed on a single RTX 3090 GPU. Any unspecified details followed the same settings as our baseline models.

5 Result and Analysis

5.1 Does tRAG better generate unseen terms than baselines?

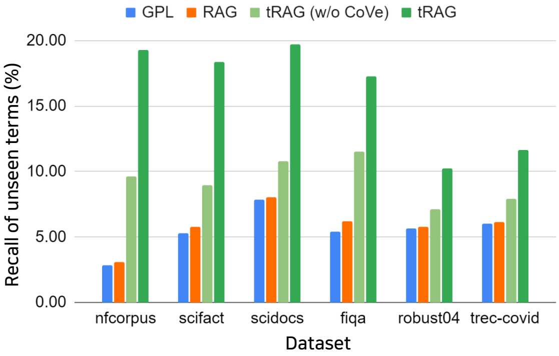

Figure 5: A graph of the recall of unseen terms for generated and expanded queries on the BEIR benchmark. tRAG shows consistent improvement in all datasets. CoVe denotes collective verification during the ranking keywords of our approach.

Figure 5 compares the recall of terms, unseen in document d, between queries generated by baseline methods and the refined queries produced by our proposed approach, tRAG. To strictly ensure the relevance of unseen terms, we calculate recall only for the unseen terms included in the relevant query terms. The results demonstrate that tRAG significantly outperforms the GPL baseline in generating unseen terms across all datasets.

The results show that the baseline approach, GPL, has a relatively low average recall rate of only 5.5% for unseen terms, illustrating the seen term bias. Notably, the RAG baseline, which augments terms in related documents within the corpus, exhibits an average unseen term recall of 5.8%. This performance is almost similar to that of the vanilla GPL baseline, indicating that the standard RAG helps little to alleviate the seen term bias. This is because document-level RAG retrieves only a few relevant documents, and the generated terms are highly biased towards the given retrieval result, making them struggle to generate unseen terms effectively.

In contrast, our proposed method demonstrates markedly superior performance, attaining an average recall of 16.1% for unseen terms, outperforming both baselines. This constitutes a substantial average improvement of 10.6%pt over the GPL baseline. The higher recall achieved by our approach highlights the effectiveness of our method in generating and ranking unseen terms, to inject

Backbone Method	Refinement Method	NFCorpus	SciFact	SciDocs	FiQA	Average
InPars	-	33.1	61.7	14.9	32.9	35.7
	RAG	33.6	61.6	14.8	33.0	35.8
	tRAG	**34.1**	**62.0**	**15.2**	**33.5**	**36.2**
GPL	-	34.2	66.4	16.1	32.8	37.4
	RAG	34.5	66.9	16.3	34.0	37.9
	tRAG	**34.9**	**67.3**	**16.8**	**37.6**	**39.2**
Contriever	-	32.8	**67.7**	16.5	32.9	37.5
	RAG	33.0	67.5	**16.6**	35.4	38.1
	tRAG	**33.5**	67.6	16.5	**36.2**	**38.5**
DRAGON	-	32.2	**67.8**	15.9	35.6	37.9
	RAG	33.6	67.4	16.3	36.8	38.5
	tRAG	**34.8**	**67.8**	**16.8**	**37.5**	**39.2**

Table 1: Comparative evaluation of nDCG@10 scores between baselines and tRAG on BEIR_{OOC} which have a large portion of unseen query terms. Experimental settings and parameter configurations used for each algorithm are described in §4.1. The best performance on each dataset for each method is highlighted in bold.

them into the refined queries.

Meanwhile, ablating the collective verification from tRAG, denoted as tRAG (w/o CoVe), which employs keywords from each document individually for refinement (i.e., K_d instead of K_D), shows worse performance compared to tRAG. This suggests that the collective verification aspect is crucial, as it allows our full method to retrieve unseen yet relevant terms from other documents in the corpus.

Hallucination-free Refinement The additional words outside the given top-100 keywords during the query refinement process can be considered a kind of hallucination. To demonstrate that this concern is impractical, we measure the frequency of hallucinations occurring in a challenging subset. Table 2 shows that such hallucinations occur at a minimal rate of about 1%. This result also indicates that our performance improvement is not due to the query refinement effect of using LLM, but rather is derived from high-quality unseen terms obtained through keyword generation and collective verification.

5.2 Does augmented unseen terms in refined queries improve performance?

To assess the effectiveness of our proposed query refinement technique in zero-shot retrieval tasks, we evaluate the retrieval performance of fine-tuned retrievers. Table 1 shows the nDCG@10 scores on BEIR_{OOC}. tRAG demonstrates significant improvements in average performance, outperforming GPL and InPars by 1.8%pt and 0.5%pt, respectively. This suggests that the effectiveness of our method is consistent with other LLMs and PQG approaches.

Dataset	Hallucination freq. (%)
NFCorpus	1.34
SciFact	0.98
SciDocs	0.80
FiQA	0.47

Table 2: Frequency of hallucination in BEIR_{OOC}. Hallucination is defined as the additional words outside the top-100 keywords during query refinement.

We observe that tRAG demonstrates consistent improvements across various subsets of the BEIR benchmark. For instance, on NFCorpus, which shows the highest unseen query term ratio, our method achieved a notable nDCG@10 gain of 0.7%pt and 1.0%pt over the GPL and InPars, respectively. This suggests that tRAG is particularly effective when most of the relevant query terms are unseen in the documents.

The improvement in FiQA is the largest, given that the unseen term ratio is the lowest. This is influenced by various factors, including the average document length. Specifically, FiQA has the shortest document length, which makes it easier to generate relevant keywords. This finding aligns with recent studies (Tohalino et al., 2023; Liu et al., 2020), suggesting that shorter texts, like abstracts, tend to produce more cohesive and relevant keywords due to the concentration of essential information with less distracting content (Wang et al., 2024).

Model	Ex.	NF	SF	SD	FQ	Avg.
InPars	X	33.6	61.8	15.1	32.9	35.9
	O	**34.1**	**62.0**	**15.2**	**33.5**	**36.2**
GPL	X	34.6	67.1	16.5	34.7	38.2
	O	**34.9**	**67.3**	**16.8**	**37.6**	**39.2**

Table 3: Results based on the absence or presence of few-shot examples in query generation. 'Ex' indicates the presence or absence of examples, while 'NF' stands for NFCorpus, 'SF' for SciFact, 'SD' for SciDocs, and 'FQ' for the FiQA dataset.

Ablation Study for Few-shot Examples We conducted an ablation study to assess the impact of few-shot examples on tRAG keyword generation. Table 3 shows that including examples in the prompt improves retrieval performance by guiding LLMs to generate unseen rather than seen terms. Notably, tRAG outperforms baselines even without few-shot examples, indicating that our performance gains are not solely dependent on them. Additionally, we selected the few-shot examples randomly, and we anticipate further performance gains with more careful selection, which we leave for future study.

Fewer Unseen Term Datasets To evaluate the impact of domain shift on our approach, we performed additional experiments on datasets having fewer out-of-corpus terms, as categorized in §3.1. In these datasets, fewer than 5% of the query terms were unseen in the corpus. Our approach consistently outperforms GPL and GPL + RAG on datasets with fewer unseen query terms, demonstrating its robustness and adaptability in handling domain shifts, even with minimal out-of-corpus terms. Please refer Appendix A.7 for details.

Performance Scaling with Various LLMs To analyze the performance based on the LLM used for query refinement, we measure the performance of tRAG using combinations of three models (GPL, Contriever, DRAGON) and three LLMs (llama-3-8b, llama-3.1-8b, gpt-3.5-turbo). Table 12 consistently demonstrates that our method shows performance improvements across various combinations of models and LLMs. This suggests that our method is robust and can leverage the increased capabilities of stronger LLMs to achieve better outcomes. Please refer to Appendix A.8 for details.

5.3 Does tRAG with K_q better enhance query quality than standard RAG?

To validate that our verified K_q improves overall query quality, we evaluate the retrieval performance of K_q when used as expansion query terms to augment the real query given at test time. As a baseline, we compare it to CSQE, the state-of-the-art query expansion method. CSQE adopts the standard RAG approach, retrieving top-k documents from the corpus of raw documents and verifying their relevance to exclude non-relevant documents from expansion.

For evaluation, we used the NFCorpus dataset, where queries are much shorter than those in other datasets. Most of the queries consist of only three words on average. In this scenario, query expansion is expected to be effective by clarifying search intents through augmented expansion terms. Results are reported in Table 4.

Method	NFCorpus
CSQE	32.2
Ours: tRAG	**33.8**

Table 4: Comparative evaluation of nDCG@10 scores on NFCorpus between CSQE and tRAG for the query expansion task. The best performance on each dataset for each retriever is highlighted in bold.

The result shows that our verified K_q improves query term quality after expansion and thus demonstrates better retrieval performance than CSQE. This validates the effectiveness of K_q in test-time query expansion as well.

6 Conclusion

This paper studied how to generate pseudo-queries to better align with the target corpus in zero-shot retrieval. Our first contribution is focusing on a notable limitation of generated queries, based on seen terms within documents, failing to include relevant "unseen" terms , as expected for generating queries for domain adaptation purposes. To address this limitation, we proposed tRAG, to optimize the term recall of unseen query terms. Our experiments showed significant improvements on BEIR benchmark, validating the critical role of unseen terms in mitigating the domain shift in zero-shot IR.

7 Limitations

Although our method is straightforward and easy to apply, adopting more powerful and advanced LLMs could potentially enhance the overall performance. As the field of natural language processing continues to rapidly evolve, incorporating state-of-the-art LLMs may lead to better keyword generation, collective verification, and overall term-level RAG performance. Despite these limitations, our work presents a novel and effective approach for term-level RAG, paving the way for future research in this area. Addressing the aforementioned limitations through further exploration and development could potentially lead to more powerful and versatile RAG systems.

Acknowledgements

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. RS-2024-00414981). This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-00077/RS-2022-II220077, AI Technology Development for Commonsense Extraction, Reasoning, and Inference from Heterogeneous Data).

References

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. *arXiv preprint arXiv:1611.09268*.

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Unsupervised dataset generation for information retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2387–2392.

Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. 2016. A full-text learning to rank dataset for medical information retrieval. In *Proceedings of the European Conference on Information Retrieval (ECIR)*. Springer.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. 2020. SPECTER: Document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282, Online. Association for Computational Linguistics.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446.

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 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Yibin Lei, Yu Cao, Tianyi Zhou, Tao Shen, and Andrew Yates. 2024. Corpus-steered query expansion with large language models. *arXiv preprint arXiv:2402.18031*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. 2023. How to train your dragon: Diverse augmentation towards generalizable dense retrieval. *arXiv preprint arXiv:2302.07452*.

Feng Liu, Xiaodi Huang, Weidong Huang, and Sophia Xiaoxia Duan. 2020. Performance evaluation of keyword extraction methods and visualization for student online comments. *Symmetry*, 12(11):1923.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Macedo Maia, S. Handschuh, A. Freitas, Brian Davis, R. McDermott, M. Zarrouk, and A. Balahur. 2018. Www'18 open challenge: Financial opinion mining and question answering. *Companion Proceedings of the The Web Conference 2018*.

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to docttttquery. *Online preprint*, 6(2).

Hongyin Tang, Xingwu Sun, Beihong Jin, Jingang Wang, Fuzheng Zhang, and Wei Wu. 2021. Improving document representations by generating pseudo query embeddings for dense retrieval. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5054–5064, Online. Association for Computational Linguistics.

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*.

Jorge AV Tohalino, Thiago C Silva, and Diego R Amancio. 2023. Using citation networks to evaluate the impact of text length on keyword extraction. *Plos one*, 18(11):e0294500.

E. Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, W. Hersh, Kyle Lo, Kirk Roberts, I. Soboroff, and Lucy Lu Wang. 2020. Trec-covid: Constructing a pandemic information retrieval test collection. *ArXiv*, abs/2005.04474.

Ellen Voorhees. 2004. Overview of the trec 2004 robust retrieval track. In *TREC*.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine an Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550, Online. Association for Computational Linguistics.

Bin Wang, Chengwei Wei, Zhengyuan Liu, Geyu Lin, and Nancy F Chen. 2024. Resilience of large language models for noisy instructions. *arXiv preprint arXiv:2404.09754*.

Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. 2021a. Gpl: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval. *arXiv preprint arXiv:2112.07577*.

Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. 2022. Gpl: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2345–2360.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Darrin Eide, K. Funk, Rodney Michael Kinney, Ziyang Liu, W. Merrill, P. Mooney, D. Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, B. Stilson, A. Wade, K. Wang, Christopher Wilhelm, Boya Xie, D. Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. 2020. Cord-19: The covid-19 open research dataset. *ArXiv*.

Xiao Wang, Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 2021b. Pseudo-relevance feedback for multiple representation dense retrieval. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 297–306.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. 2023. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2550–2575, Singapore. Association for Computational Linguistics.

A Appendix

A.1 Prompt for Summary

The prompt for summary can be found on Table 5.

tRAG Prompt (Summary)

The provided passages are the ones retrieved by searching for the query. Please summarize the content of these passages into a single passage.

Query: {query}

Retrieved Documents: {retrieved documents}

Table 5: Prompt of tRAG to summarize the retrieved documents. $\{\cdot\}$ denotes the placeholder for the corresponding text.

A.2 Prompt for Keyword Extraction

The prompt for summary can be found on Table 6.

tRAG Prompt (Keyword Extraction)

Generate relevant keywords to the given document.

Document-1: {document-1}

Keywords-1: {keyword-1-1}, {keyword-1-2}, $\cdots$

Document-2: {document-2}

Keywords-2: {keyword-2-1}, {keyword-2-2}, $\cdots$

Document-3: {document-3}

Keywords-3: {keyword-3-1}, {keyword-3-2}, $\cdots$

Document-4: {given document}

Keywords-4:

Table 6: Prompt of tRAG to generate the keywords from a document. $\{\cdot\}$ denotes the placeholder for the corresponding text.

A.3 Prompt for Keyword Selection

The prompt for summary can be found on Table 7.

A.4 Prompt for Query Refinement

The prompt for summary can be found on Table 8.

tRAG Prompt (Keyword Selection)
Select keywords that are relevant to the given query and document.
Keywords: {keyword-1}, {keyword-2}, $\cdots$
Query: {query}
Document: {document}

Table 7: Prompt of tRAG to select the keywords from a document. $\{\cdot\}$ denotes the placeholder for the corresponding text.

tRAG Prompt (Query Refinement)
Refine the query based on the given summary of retrieved documents related to the query.
Query: {query} {keywords}
Summary: {summary}

Table 8: Prompt of tRAG to refine the query q by utilizing the summary S_{D_q} and keywords K_q. $\{\cdot\}$ denotes the placeholder for the corresponding text.

A.5 Target Datasets

Notable statistics of the 6 target datasets can be found on Table 9.

A.6 Examples of Refined Query

Example queries refined by tRAG.

A.7 Performance on Fewer Unseen Term Datasets

The low out-of-corpus term ratio is due to the large corpus size and the high number of relevant documents per query. Each query has many relevant documents (e.g., an average of 493.5 per query in Trec-Covid), increasing the likelihood that query terms appear in the relevant documents. The large corpus size also lowers the ratio since it covers more terms than other datasets. These attributes make GPL and RAG (employing in-document and in-corpus vocabulary, respectively) effective for datasets having fewer out-of-corpus terms.

Table 11 shows our approach consistently outperforms GPL and GPL + RAG on datasets with fewer unseen query terms. This improvement demonstrates the robustness and adaptability of our approach in handling domain shifts, even in scenarios with a minimal presence of out-of-corpus terms.

A.8 Performance on Various LLMs

To validate the effectiveness of our method across different models and LLMs, we have conducted additional experiments. Specifically, we evaluated three models (GPL, Contriever, DRAGON) with three LLMs (llama-3-8b, llama-3.1-8b, GPT-3.5-turbo). The results consistently demonstrate that our method shows performance improvements across various combinations of models and LLMs. Notably, there is a clear trend indicating that the use of more advanced LLMs generally results in higher performance. This suggests that our method is robust and can leverage the increased capabilities of stronger LLMs to achieve better outcomes. These findings reinforce the versatility and scalability of our approach, confirming that it performs well across different settings and benefits from the enhanced features of more powerful LLMs.

A.9 Usage of AI Assistants

ChatGPT was employed to enhance the clarity and grammatical accuracy of the text, offering suggestions for sentence rephrasing and correction of grammatical errors.

	High Out-of-corpus Ratio (BEIR$_{\text{OOC}}$)				**Low Out-of-corpus Ratio**	
	SciFact	**SciDocs**	**FiQA**	**NFCorpus**	**Robust04**	**Trec-Covid**
Domain	Scientific	Scientific	Financial	Bio-Medical	News	Bio-Medical
Total # Queries	300	1000	648	323	249	50
Total # Documents	5.2k	25.7k	57.6k	3.6k	528k	171k
Average Query Length (words)	12.4	9.4	10.8	3.3	15.3	10.6
Average Document Length (words)	213.6	176.2	132.2	232.3	466.4	160.8
Relevant Document / Query	1.1	4.9	2.6	38.2	69.9	493.5
Unseen (Out-of-corpus) Term Ratio	16.8	19.4	9.8	15.1	3.0	2.6
Unseen (Out-of-d) Term Ratio	55.1	79.6	64.2	88.0	59.8	66.3

Table 9: Detailed statistics of the six datasets included in the BEIR Benchmark as employed in our experiments as we categorized in 3.1.

Original query	**Refined Query**
Breast cancer	How breast cancer cells feed on cholesterol?
Key factors of cancer	What role do invadopodia play in cancer?
RNA-binding during stress	What happens to RNA-binding proteins during stress?

Table 10: Examples of refined query.

Method	**Robust04**	**Trec-Covid**	**Average**
GPL	41.4	71.8	56.6
GPL + RAG	41.6	72.1	56.8
Ours: GPL + tRAG	**41.8**	**72.2**	**57.0**

Table 11: Comparative evaluation of nDCG@10 scores between baselines and tRAG on datasets with fewer out-of-document terms from BEIR benchmark. The best performance on each dataset for each method is highlighted in bold.

Model	LLM	NFCorpus	SciFact	SciDocs	FiQA	Average
GPL	llama-3-8b	34.3	66.8	16.4	34.1	37.9
	llama-3.1-8b	34.7	67.0	16.2	34.4	38.1
	gpt-3.5-turbo	**34.9**	**67.3**	**16.8**	**37.6**	**39.2**
Contriever	llama-3-8b	33.1	67.2	16.3	34.9	37.9
	llama-3.1-8b	**33.6**	**67.7**	16.3	35.3	38.2
	gpt-3.5-turbo	33.5	67.6	**16.5**	**36.2**	**38.5**
DRAGON	llama-3-8b	34.6	67.5	15.7	35.9	38.4
	llama-3.1-8b	34.5	67.6	16.4	36.0	38.6
	gpt-3.5-turbo	**34.8**	**67.8**	**16.8**	**37.5**	**39.2**

Table 12: Comparative evaluation of nDCG@10 scores on various LLMs. The best performance on each dataset for each method is highlighted in bold.