Retrieval-Augmented Retrieval: Large Language Models are Strong Zero-Shot Retriever

Tao Shen[1], Guodong Long[1], Xiubo Geng[2], Chongyang Tao[2*], Yibin Lei[3], Tianyi Zhou[4], Michael Blumenstein[1], Daxin Jiang[2]
[1]AAII, FEIT, University of Technology Sydney [2]Microsoft Corporation
[3]University of Amsterdam [4]University of Maryland
`{tao.shen,guodong.long,michael.blumenstein}@uts.edu.au`
`{xigeng,chotao,djiang}@microsoft.com`, `y.lei@uva.nl`, `tianyi@umd.edu`

Abstract

We propose a simple method that applies a large language model (LLM) to large-scale retrieval in zero-shot scenarios. Our method, the Large language model as Retriever (LameR), is built upon no other neural models but an LLM in a retrieval-augmented retrieval fashion, while breaking brute-force combinations of retrievers with LLMs and lifting the performance of zero-shot retrieval to be very competitive on benchmark datasets. Essentially, we propose to augment a query with its potential answers by prompting LLMs with a composition of the query and the query's in-domain candidates. The candidates, regardless of correct or wrong, are obtained by a vanilla retrieval procedure on the target collection. As a part of the prompts, they are likely to help LLM generate more precise answers by pattern imitation or candidate summarization. Even if all the candidates are wrong, the prompts at least make LLM aware of in-collection patterns and genres. Moreover, due to the low performance of a self-supervised retriever, the LLM-based query augmentation becomes less effective as the retriever bottlenecks the whole pipeline. Therefore, we propose to leverage a non-parametric lexicon-based method (e.g., BM25) as the retrieval module to capture query-document overlap in a literal fashion. As such, LameR makes the retrieval procedure transparent to the LLM, thus circumventing the bottleneck.

1 Introduction

Large-scale (or first-stage) is to fetch top relevant documents for a given text query from a huge collection with millions to billions of entries. It is indispensable in information-seeking tasks, such as open-domain question answering (Chen et al., 2017), web search (Shen et al., 2022), knowledge-grounded dialogue(Zhao et al., 2020), etc. Recently, it is also leveraged as a core retrieval-augmenting module to enrich large language models (LLMs) with up-to-date or domain-specific knowledge (Guu et al., 2020; Trivedi et al., 2022), which reduces the hallucination problem (Shuster et al., 2021) and improves the faithfulness of generated texts (He et al., 2023). Thereby, large-scale retrieval is a long-term research problem, attracting research efforts from academia and industry.

In the last decade, large-scale retrieval relies heavily on deep representation learning techniques, from bag-of-words (BoW) (Mikolov et al., 2013) to pre-trained language models (PLMs) (Devlin et al., 2019). Compared to supervised representation learning (Karpukhin et al., 2020; Xiong et al., 2021) that requires labor-intensive annotations on query-document pairs, self-supervised (or zero-shot) learning (Lee et al., 2019; Ni et al., 2021; Izacard et al., 2021; Muennighoff, 2022) on in-domain pseudo pairs can be readily generalized to any corpora without human-crafted annotations. Nonetheless, the zero-shot retriever usually results in an inferior retrieval quality (Zhou et al., 2022a), even worse than a non-parametric term-based BM25 retrieval (Thakur et al., 2021; Zhou et al., 2022a).

Fortunately, recent surging LLMs provide a shortcut to reach zero-shot retrieval by augmenting a query with its potential answering elicited from the LLMs (Gao et al., 2022). Coupled with a self-supervised retriever, Contriever (Izacard et al., 2021), it delivers superior retrieval performance, even surpassing a number of supervisedly fine-tuned retrievers. But, such a brute-force combination of a self-supervised retriever with a versatile LLM leads to a major problem. The answer elicitation is merely based on prompting LLMs with short, intent-ambiguous, and domain-vague retrieval queries. Due to the ambiguity of user queries and unawareness of in-domain corpora, the LLMs are likely to generate spurious and out-of-domain answers to the queries (Asai et al., 2022), making the query augmentation even more toxic.

*Corresponding author.

Findings of the Association for Computational Linguistics ACL 2024, pages 15933–15946
August 11-16, 2024

To circumvent this issue, we propose a brand-new and simple paradigm for large-scale retrieval, called LameR. Essentially, during eliciting LLMs for answers to a query, we inject the query's top answer candidates into the prompt, where the candidates are obtained by applying a vanilla retrieval procedure to the query. As such, the LLMs are prone to distinguish and imitate the candidates (Brown et al., 2020), while summarizing or/and re-writing new ones with internal knowledge of the LLMs. Moreover, despite correct or wrong candidates, they can at least provide demonstrations about in-domain patterns and knowledge (Min et al., 2022; Xie et al., 2022; Lyu et al., 2022).

Moreover, though the LLMs now generate more precise, and reliable query augmentations, the whole pipeline is likely to be bottlenecked by the weak retriever trained on pseudo data in a self-supervised manner. Therefore, we also propose to get rid of any learnable parametric retrievers, while opting for non-parametric term- or lexicon-based retrieval methods (e.g., BM25 in our experiments) in our LameR. In contrast to model-specific compressed and/or latent embeddings from a deep retriever, the lexicon-based retrieval methods capture lexicon overlap between augmented queries and in-collection documents in a literal fashion, thus taking the outputs of LLMs in a transparent mode and bypassing the performance bottleneck.

We evaluate LameR on several benchmark datasets of large-scale retrieval by following Gao et al. (2022). Our results show that our proposed method achieves the best retrieval qualities on most datasets compared to other zero-shot competitors. Also, it can surpass the LLM-based retriever with in-context labeled demonstrations and outperform the baseline retrievers fine-tuned on full datasets.

2 Observation

In our pilot experiments, we observed the brute-force combination of a versatile LLM with weak retriever leads to certain demerits, which primarily

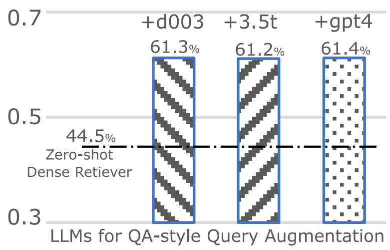

Figure 1: nDCG on DL19 for query augmentation w/ LLMs.

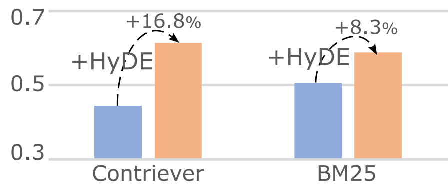

Figure 2: HyDE improving Dense and Term-based Retrieval.

motivates this work.

Bottleneck by Self-supervised Retriever. Due to the weakness of a self-supervised dense retriever in representing capability, the whole pipeline is bottlenecked by the retriever, even though correct answers are likely to be generated by the strong LLM. As illustrated in Figure 1, strengthening LLMs in QA-style query augmentation (i.e., HyDE (Gao et al., 2022), which elicits an LLM to generate answers as query augmentation) hardly improves retrieval performance. Here, '`d003`' and '`3.5t`' denote `text-davinci-003` and `gpt-3.5-turbo` by OpenAI, respectively.

Mismatch w/ Term-based Retriever. Due to unawareness of in-domain corpora, LLM is likely to generate out-of-domain answers to a given context-short and intent-vague query, making the query augmentation even toxic. Thanks to the fuzzy capability of dense retrievers, such query augmentation still bring remarkable improvement in search quality. However, when it comes to lexicon-based retrieval (say BM25), the improvement will be reduced due to out-of-domain augmentations. Quantitatively, as in Figure 2, 'Contriever' is a SoTA self-supervised dense retriever while 'BM25' is a representative lexicon-based retrieval. It is observed that although BM25 can beat Contriever in the vanilla setting, HyDE brings twice more improvement to Contriever than BM25, making BM25 less competitive.

3 Related Work

Although an LLM can directly generate relevant documents and even the final answer for a user query upon its parametric memory, such a generative information-seeking approach is limited by: i) out-of-date corpora are learned in the parametric memory, ii) unreliable, and hallucinative text is frequently generated, and iii) the domain of generated text cannot be specified as demand. In contrast, information retrieval aims to provide in-domain and reliable documents relevant

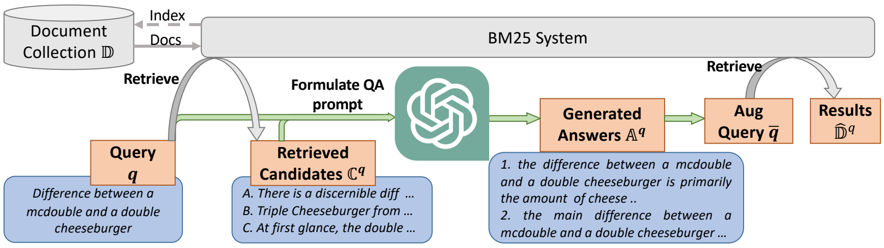

Figure 3: **La**rge language **m**od**e**l as **R**etriever (LameR). Please see Table 1 for the prompt formulation.

to user queries, which dominates people's daily information-seeking methods.

Therefore, many research efforts have recently been dedicated to applying large language models (LLMs), such as the GPT series, to information retrieval tasks for superior search performance. The majority of these works are in few-shot or zero-shot scenarios. Yu et al. (2022) proposed a generate-then-read pipeline instead of the traditional retrieve-then-read pipeline. Dai et al. (2022) introduced a few-shot dense retrieval approach for different tasks with different retrieval intents. Dua et al. (2022) proposed a data augmentation method for domain adaptation for open-domain QA, where a document is passed to LLM for the generation of its possible queries. Gao et al. (2022) focused on zero-shot dense retrieval using the Hypothetical Document Embedding (HyDE) method to generate potential answers by LLM as query augmentation. Jeronymo et al. (2023) and Boytsov et al. (2023) leveraged a fine-tuned ranker (on MS-MARCO in a supervised manner) to filter LLM-generated data for better query-document quality and thus superior performance. Saad-Falcon et al. (2023) designed a two-stage LLM pipeline for zero-shot query generation and reranker-distilled retriever. Wang et al. (2023) utilized a few-shot query-document demonstration to generate documents for a new query as the query's augmentation.

Unlike these works, we focus on the zero-shot retrieval scenario, and neither conduct any in-domain data augmentation for domain-specific retriever training nor introduce any other retrieval or/and intermediate models except for a frozen LLM.

Please refer to Appendix A for more discussions of *§A.1 zero-shot large-scale retrieval* (which is our scope), *§A.2 in-context learning* (as we could regard the BM25-retrieved docs as unlabeled demonstrations), and *§A.3 retrieval & rerank pipeline* (as we have a two-step retrieval pipeline).

4 Language Language Model as Retriever

This section begins with a task definition, followed by elaborations on three components to achieve LameR – non-parametric lexicon-based retriever (§4.1), candidate-prompted answer generation (§4.2), and answer-augmented large-scale retriever (§4.3). LameR is illustrated in Figure 3.

Task Definition: Zero-Shot Large-Scale Retrieval. Providing a huge collection consisting of many documents, $\mathbb{D} = \{d_i\}_{i=1}^{|\mathbb{D}|}$, the goal of 'large-scale retrieval' is to rank the whole $\mathbb{D}$ in descending order according to the relevance score between a given text query q and each d_i. The relevance score is usually derived by a high-efficient retrieval model that operates on a pre-indexed $|\mathbb{D}|$ and an on-the-fly q to satisfy real-time requirements. Meantime, 'zero-shot' means that there is no training set with labeled positive query-document pairs for supervised representation learning.

4.1 Non-parametric Lexicon-based Retriever

To tackle zero-shot retrieval, a recent trend is to train a deep encoder (e.g., BERT) over pseudo query-document pairs in a self-supervised manner, where the pairs are heuristically mined from the target collection $\mathbb{D}$. Although the self-supervised learning process is required to especially repeat or/and design for every retrieval collection (Lee et al., 2019; Zhou et al., 2022a), the resulting retrieval performance is not satisfactory in most cases, lagging far behind fully-supervised retrievers.

In contrast, non-parametric term- or lexicon-based retrieval methods, e.g., TF-IDF and BM25 [1], are free of training heavy neural networks, but depend on lexicon overlap with considering term and document frequency of the lexicons. Even so, the simple BM25 retrieval method can outperform the

[1]Although the two hyper-parameters, i.e., k_1 and b, in BM25 algorithm can be tuned, for example, by grid search, we do not seek to tune them but keep them in defaults, i.e., $k_1 = 0.9$ and $b = 0.4$, in Pyserini (Lin et al., 2021).

Candidate-prompted Instruction.
Give a question "{q}" and its possible answering passages (most of these passages are wrong) enumerated as: \n 1.{c_1^q} \n 2.{c_2^q} \n 3.{c_3^q} ... please write a correct answering passage.

Table 1: Our simple QA prompt to elicit knowledge from LLM for information retrieval in our LameR. Here, the entry with '{·}' represents a placeholder for the corresponding text. $c_l^q \in \mathbb{C}^q$ denotes a retrieved candidate. Please see Appendix B for the prompts for all datasets.

self-supervised retriever in many cases in zero-shot retrieval (Zhou et al., 2022a; Thakur et al., 2021).

Therefore, in this work we leverage the BM25 method (Robertson and Zaragoza, 2009) to perform large-scale retrieval. The core idea of BM25 is to rank documents according to their relevance to a given query by incorporating term frequency and inverse document frequency. In brief, its relevance score between a document $d \in \mathbb{D}$ and a query q is defined as

$$\mathrm{Rel}^{\mathrm{BM25}}(d,q) = \sum_{t\in q} \mathrm{IDF}(t)\cdot \frac{\mathrm{TF}(t,d)\cdot(k_1+1)}{\mathrm{TF}(t,d)+k_1\cdot(1-b+b\cdot\frac{\mathrm{len}(d)}{\mathrm{avgdl}})}, \quad \text{where } \mathrm{IDF}(t)=\log\frac{N-n(t)+0.5}{n(t)+0.5}. \tag{1}$$

Here, t denotes a lexicon term in q, $\mathrm{TF}(t,d)$ is the term frequency of t in document d, and $\mathrm{IDF}(t)$ is the inverse document frequency of term t, $N=|\mathbb{D}|$ is the total number of documents in the collection, $n(t)$ is the number of documents containing term t, $\mathrm{len}(d)$ is the length of d, and avgdl is the average document length across the collection. In the remainder, we define a retrieval procedure as

$$\hat{\mathbb{D}}^q = \mathrm{Retriever}(q, \mathbb{D}, K). \tag{2}$$

$\hat{\mathbb{D}}^q$ is a list of top-K retrieval candidates of q with descending relevance scores, so $|\hat{\mathbb{D}}^q| = K$.

Remark. When employing a strong, non-tunable, generative model, e.g., LLM, for explicit text augmentations of a query, a lexicon-based retrieval method has its own merit in not only high efficiency, but taking the exact augmentations for retrieval without compressed embedding. Therefore, using the lexicon-based method exposes LLMs' outputs to the retrieval collection literally, making the retrieval module transparent to LLMs. By comparison, the neural encoder, trained on heuristically mined pseudo data in a self-supervised manner, is too weak to model the LLM-augmented queries, leaving a performance bottleneck here (see §2).

4.2 Candidate-Prompted Answer Generation

Given a query q, we augment it with its answer(s) a elicited from an LLM, which has been proven effective in improving zero-shot retrieval quality (Gao et al., 2022; Wang et al., 2023). How to conduct the elicitation remains an open question. For example, in a straightforward way, Gao et al. (2022) propose to prompt an LLM with a composition of a QA instruction and the query. However, as the LLM can only receive a short, intent-ambiguous query, joined with a broad and general QA instruction, it is not well instructed by the prompt with both the intent and domain of a query, leading to less precise answers. Wang et al. (2023) add few-shot query-document examples as in-context demonstrations to the prompt for more reasonable answers, which, however, is unavailable in zero-shot settings.

Instead, we propose a new prompt schema, called candidate-prompted answer generation, for query augmentation in large-scale retrieval. As shown in Table 1, besides a task instruction and a retrieval query, a list of top answering candidates is also included in the prompt for elicitation of an LLM. Here, the top candidates are obtained by directly applying a vanilla retrieval process to the query via the retriever (§4.1). So, we first retrieve top-M candidates for q from the whole $\mathbb{D}$ by

$$\mathbb{C}^q = \mathrm{Retriever}(q, \mathbb{D}, M), \tag{3}$$

where M is usually very small (e.g., < 10) to reduce computation overhead for downstream modules. Then, to elicit knowledge from an LLM, we construct a prompt with $\mathbb{C}^q$ and then invoke the LLM for answer generation, i.e.,

$$\mathbb{A}^q = \{a_1^q \dots a_N^q | a^q \sim \mathrm{LLM}\left(p\left(t, q, \mathbb{C}^q\right)\right)\} \tag{4}$$

where $p(\cdot)$ composes the prompt using task an instruction t, the query q, and the retrieved candidates $\mathbb{C}^q$ (see Table 1 for an example and Appendix B for prompts of all tasks). It is noteworthy that we generate multiple (i.e., N) answers by sampling outputs of the LLM, because we'd like to provide

	TREC Deep Leaning 2019			TREC Deep Leaning 2020		
	MAP	nDCG@10	R@1k	MAP	nDCG@10	R@1k
w/o relevance judgment (zero-shot retrieval)						
BM25	30.1	50.6	75.0	28.6	48.0	78.6
Contriever	24.0	44.5	74.6	24.0	42.1	75.4
HyDE	41.8	61.3	88.0	38.2	57.9	84.4
LameR (ours)	**47.2**	**69.1**	**89.9**	**45.6**	**64.8**	**88.7**
reranking w/o relevance judgment						
$\text{BM25}_{\text{top-100}} \rightarrow$LRL (Ma et al., 2023)	-	65.8	-	-	62.2	-
w/ few-shot relevance judgment (few-shot ICL for answer generation)						
Q2D_{BM25}	-	66.2	-	-	62.9	-
w/ relevance judgment (fully-supervised fine-tuning)						
DPR	36.5	62.2	76.9	41.8	65.3	81.4
ANCE	37.1	64.5	75.5	40.8	64.6	77.6
$\text{Contriever}^{\text{FT}}$	41.7	62.1	83.6	43.6	63.2	85.8

Table 2: Results for web search on DL19/20. Best w/o relevance judgment is marked **bold**. DPR, ANCE and $\text{Contriever}^{\text{FT}}$ are in-domain *supervised* models that are finetuned on MS-MARCO training data. We use `gpt-3.5-turbo` by default.

as many potential answers as we can to prevent the 'vocabulary mismatch' problem.

As such, $\text{LLM}(\cdot)$ utilizes the answering candidates $\mathbb{C}^q$ in two aspects: i) If one or many gold documents of q existing in $\mathbb{C}^q$, $\text{LLM}(\cdot)$ serves like a re-ranker and generates the answers $\mathbb{A}^q$ by both summarizing the correct documents from $\mathbb{C}^q$ and eliciting internal parameterized knowledge. ii) Regardless of the correctness of $\mathbb{C}^q$, $\text{LLM}(\cdot)$ also receives in-collection answering information about intents, domains, and units, which are prone to help the LLM generate more precise answers $\mathbb{A}^q$.

4.3 Answer-Augmented Large-Scale Retrieval

Given the generated answers $\mathbb{A}^q$ of q, we use them to augment q and produce a new query $\bar{q}$. Attributed to the non-parametric lexicon-based retriever, we can perform the query augmentation in a very straightforward way, which operates on plain text rather than latent embeddings. That is, we can easily concatenate every $a^q \in \mathbb{A}^q$ with the original q, i.e.,

$$\bar{q} = \text{Concat}(q, a^q_1, q, a^q_2, \dots, q, a^q_N), \qquad (5)$$

where Concat denotes a concatenation operation in text. Lastly, we simply use the augmented query, $\bar{q}$, to conduct a large-scale retrieval,

$$\hat{\mathbb{D}}^{\bar{q}} = \text{Retriever}(\bar{q}, \mathbb{D}, K), \qquad (6)$$

where $\hat{\mathbb{D}}^{\bar{q}}$ is a list of final retrieved documents for query q and $K = 1000$ for metric calculation. Thanks to the high efficiency of the lexicon-based retriever with an inverted index, the augmentation would not cause catastrophic overhead increases, which is still faster than a dense retriever.

5 Experiment

In this section, we will conduct extensive experimental evaluations of the proposed retrieval method and compare it with strong competitors. Implementation of LameR is available at `https://github.com/taoshen58/LameR`.

5.1 Evaluation Setup

Datasets and Metrics. Following the datasets used by Gao et al. (2022), we first employ the widely-used passage retrieval datasets, MS-MARCO (Nguyen et al., 2016) and report performance on TREC Deep Learning 2019 (Craswell et al., 2020) and TREC Deep Learning 2020(Craswell et al., 2021) test sets (DL19 and DL20 for short, respectively). Meantime, we also evaluate our method on BEIR benchmark (Thakur et al., 2021). Here, we follow Gao et al. (2022) to consider low-resource datasets from the BEIR dataset, so we employ six datasets, consisting of one fact-checking task (Scifact), one question-answering task (FiQA), one bio-medical IR task (TREC-COVID), one news retrieval task (TREC-NEWS), one argument retrieval task (ArguAna), and one entity retrieval task (DBPedia). Note that, as a zero-shot retrieval setting, we do not use any training query-document pairs but directly evaluate our method in the test sets. Following previous works, we report MAP, nDCG@10 and Recall@1000 (R@1k) for both TREC Deep Learning 2019 and TREC Deep Learning 2020. And nDCG@10 is reported for all the datasets in the BEIR benchmark.

nDCG@10	Scifact	Arguana	Trec-COVID	FiQA	DBPedia	TREC-NEWS
w/o relevance judgment						
BM25	67.9	39.7	59.5	23.6	31.8	39.5
Contriever	64.9	37.9	27.3	24.5	29.2	34.8
HyDE	69.1	**46.6**	59.3	**27.3**	36.8	44.0
LameR (ours)	**73.5**	40.2	**75.8**	25.8	**39.0**	**50.3**
w/ few-shot relevance judgment						
$Q2D_{BM25}$	68.6	-	72.2	-	37.0	-
w/ relevance judgment						
DPR	31.8	17.5	33.2	29.5	26.3	16.1
ANCE	50.7	41.5	65.4	30.0	28.1	38.2
$\text{Contriever}^{\text{FT}}$	67.7	44.6	59.6	32.9	41.3	42.8

Table 3: Low resource tasks from BEIR. Best performing w/o relevance judgment are marked **bold**.

Experimental Setup. As for the large language model, we use `gpt-3.5-turbo` as the LLM to perform answer generation by default. Meantime, we also involve `gpt-4` to investigate whether stronger LLM will bring more improvement. And, the number of candidates, M in Eq.(3), is set to 10 in our main results, and the number of generated answers, N in Eq.(4) is set to 5. To ensure efficiency, we truncate each of the queries and passages/documents to 128 tokens.

Baselines and Competitors. As we focus on the zero-shot retrieval setting, our main baselines fall into the retrieval methods without dependency on annotated query-document pairs (i.e., *w/o* relevance judgment). In particular, we use BM25 (Robertson and Zaragoza, 2009) and Contriever (Izacard et al., 2021) as strong baselines for zero-shot lexicon and dense retrieval, respectively. And, we also include HyDE (Gao et al., 2022) as the state-of-the-art competitor for LLM-based retrieval. Furthermore, we also employ some baselines not in zero-shot settings to verify the effectiveness of our method. On the one hand, we leverage Q2D+BM25 (Wang et al., 2023) as a few-shot baseline (i.e., *w/ few-shot relevance judgment*), where in-context gold query-document pairs are provided to help LLM generate answers for a query. On the other hand, we consider some popular fully-supervised retrieval models (i.e., *w/ relevance judgment*), including DPR (Karpukhin et al., 2020), ANCE (Xiong et al., 2021), fine-tuned Contriever (Izacard et al., 2021), etc.

5.2 Main Evaluation

DL19 and DL20 Test Sets. As shown in Table 2, we compare our LameR with its baselines and competitors in both TREC Deep Learning 2019 and 2020 test sets. It is observed that our method achieves the best performance in the zero-shot setting, significantly outperforming its strong competitor, HyDE[2]. This clearly verifies the effectiveness of our candidate-prompted answer generation. It is also noteworthy that our LameR is based on a much faster BM25 retriever, in contrast to the heavy dense retriever, Contriever, in HyDE. Meantime, compared to the method ($Q2D_{BM25}$) with few-shot relevance judgment and the methods (DPR, etc.) with full relevance judgment, our proposed LameR achieves the best on most retrieval evaluation metrics. Note LameR can surpass retrieval (w/ BM25) & reranking (w/ LLM) pipeline, i.e., LRL (Ma et al., 2023) (or GPT-reranker (Sun et al., 2023)), showing the superiority of our two-step pipeline.

BEIR Benchmark. Furthermore, we compare our retrieval method with the others on six low-resource tasks from the BEIR dataset. As shown in Table 3, our proposed method performs best on four out of six datasets. It should be highlighted that our LameR achieves superior performance on two TREC retrieval datasets, i.e., TREC-COVID and TREC-NEWS, which verify our proposed method in web information-seeking tasks. Meantime, We found our LameR delivers poor results on 'Argunan', a dataset designed to retrieve counter-argument passages from a collection. Since the queries and documents in the dataset are usually over-long ($>$ 256), this is possibly caused by applying aggressive truncation (cap at 128) to the long queries and passages in the dataset. Besides, we also noticed that the performance of FiQA in zero-shot settings is far from that in the few-shot or fully-supervised settings. This may be caused by

[2] Although HyDE uses `text-davinci-003` as its LLM, we found updating it with `gpt-3.5-turbo` leads to similar retrieval performance. See Figure 1 for details.

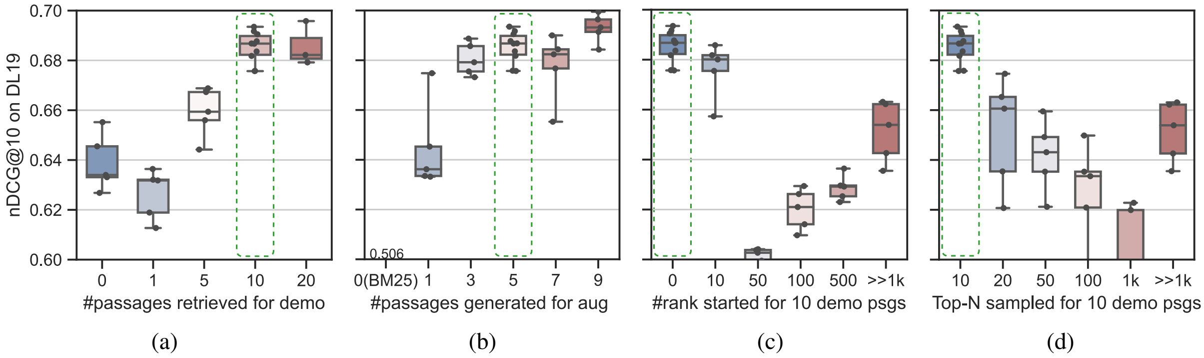

Figure 4: Hyperparameter explorations and ablation studies, where the data points in dashed rectangles denote our default choices. **(a)** The number of retrieved passages as in-context demonstration for answer generation, i.e., M in Eq.(3). **(b)** The number of generated answers as query augmentations for large-scale retrieval, i.e., N in Eq.(4). **(c)** and **(d)** depict the schemes to obtain the 10 demo-passages, where the first is to fetch 10 consecutive passages from a *start index* of the BM25-retrieved passages and the second is to randomly sample 10 passages from *top-N* passages. Note that '$\gg$1k' denotes randomly sampling 10 passages from the whole collection.

DL19	MAP	nDCG@10	R@1k
DPR	36.5	62.2	76.9
Contriever-FT	41.7	62.1	83.6
HyDE$_{\text{LLaMA-2-7B}}$	37.5	57.1	82.0
LameR$_{\text{LLaMA-2-7B}}$	40.6	59.8	83.8
HyDE$_{\text{LLaMA-2-13B}}$	38.8	58.3	83.7
LameR$_{\text{LLaMA-2-13B}}$	42.8	64.9	84.2

Table 4: LameR with open-source LLMs on DL19.

DL19	MAP	nDCG@10	R@1k
BM25	30.1	50.6	75.0
LameR (dflt)	47.2	69.1	89.9
LameR-oracle	60.7	84.0	93.8
◇ 2nd Round	46.7	68.1	87.5

Table 6: Exploring extremes of LameR.

the lack of financial knowledge in general LLM.

Open-Source LLMs. Table 4 shows the effectiveness of integrating open-source LLMs, LLaMA-2-chat-7B and -13B (Touvron et al., 2023b), into the LameR framework for the DL19 dataset. Notably, LameR, when augmented with these LLMs, outperforms both HyDE configurations and traditional methods like DPR and Contriever-FT, showcasing the adaptability and efficiency of LameR with various LLM backbones.

Power of Stronger LLM. To further verify if our LameR will benefit from stronger LLM, we involve the bleeding-edge LLM, GPT-4, in our LameR framework and apply it to DL20 dataset as its results in the main evaluation with GPT-3.5 is not superior enough. As shown in Table 5, after applying GPT-4, our retrieval method achieves significantly high performance and beats all the competitors even with full relevance judgment.

DL20	nDCG@10
BM25	48.0
HyDE	57.9
DPR (supv.)	65.3
LameR$_{\text{GPT-3.5}}$	64.8
LameR$_{\text{GPT-4}}$	**65.9**

Table 5: LameR with GPT4.

5.3 Ablation Study and Further Analysis

Number of Retrieved Demos. First, we investigate whether the number of retrieved passages (as in-context demonstration) affects query augmentation and thus retrieval quality. As shown in Figure 4(a), increasing $M > 0$ consistently brings improvement in answer-augmented large-scale retrieval, and the improvement becomes marginal when the number exceeds 10. Considering that increasing M inevitably causes more computation overheads, we use $M = 10$ for a better trade-off between performance and efficiency. Besides, an interesting point is that LameR with $M = 0$ is surprisingly better than both i) HyDE, which verifies the effectiveness of our query augmentation coupled with BM25 retrieval, i.e., Eq.(5-6), and ii) LameR with $M = 1$, which is likely caused by low recall performance in top-1 and more severe interference of error candidates.

Number of Answers. We also investigate if the number of answers generated by LLM will affect the performance of our LameR. As shown in Figure 4(b), the performance of retrieval grows along with the number of generated answers, but becomes fluctuating and saturated when $N > 5$. Therefore, we use $N = 5$ as the default in our experiments.

Schemes to Obtain Demo-passages. We leverage top-10 retrieved passages as demonstrations as they are likely to provide pivot query-related knowledge in a limited context window of LLMs. To empirically check this intuition, we propose three schemes for demo-passages: i) As shown in Figure 4(d), the performance consistently drops when we increase the sample range because the related knowledge and correct demonstrations are weakened gradually. ii) As shown in Figure 4(c), we fetch 10 consecutive passages from different start indices in BM25 results. Surprisingly, there is a U-shaped curve, which can be explained by 'hard negatives' widely presenting in IR: Basically, hard negatives in top candidates challenge LLMs' distinguishing capability between positives and hard negatives. What's worse, with increasing start indices, the correct passages scarcely appear in the 10 consecutive passages, making the LLMs lose contrastive samples and get fooled by the negatives. iii) More interestingly, as the '$\gg$1k' in both Figure 4(c) & 4(d), randomly sampling 10 entries from the whole collection as demo-passages results in surprisingly high results. This is because they are focused on providing useful information about the knowledge domain (e.g., web, news, Wikipedia, scientific, arguments), task intent (e.g., dialogue, question answering), answering format (e.g., unit, length, pattern), etc., while free from hard negatives or spurious answers.

Exploring Extremes of LameR. As LameR is built upon BM25 retrieval system, the lower bound of LameR would be BM25. Go beyond, it is interesting to find out the upper bound of LameR, which can demonstrate the extreme performance that LameR may deliver. As shown in Table 6, we conduct an experiment called 'LameR-oracle', where 10 demo-passages are instead obtained by gold query-document pairs in the labeled test set. It's seen that compared to our LameR w/ default settings (i.e., dflt), LameR-oracle performs much higher, verifying i) the importance of the correctness of demonstrated passages and ii) a great improvement room left for further research. As an initial exploration, we propose a brute-force attempt that a 2nd-round LameR is applied to the retrieval results by default LameR, but to our surprise, the performance even drops by absolute 1.0% nDCG@10 (see the last row of Table 6). Sharing inspirations with error reinforcement, the query augmented by an LLM (in the 1st round) is prone to return spurious passages that especially confuse the LLM (i.e., hardly distinguished), resulting in wrong answers to poison BM25. This suggests that in the future, we should focus more on introducing multiple retrieval methods to achieve diversity.

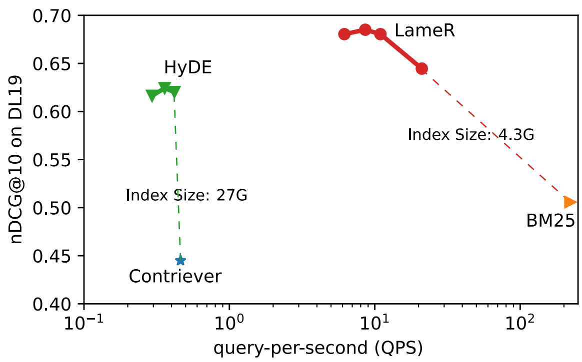

Figure 5: Efficiency of LameR with HyDE in retrieval latency (QPS) and index size (GB). Numbers for LameR sum overheads in two stages, and the variants for each system are achieved by changing the generation number.

5.4 Efficiency Analysis

Overheads with LLMs. Similar to HyDE (Gao et al., 2022) and Q2D (Wang et al., 2023), using LLMs to generate query augmentations inevitably leads to high computation overheads. Optimistically speaking, such inference-only overheads do not increase with the scale of retrieval collection, and a recent trend is to make smaller LLMs competitive (Touvron et al., 2023a; Taori et al., 2023), which would benefit these methods. In the future, we will explore specializing in a smaller LLM to generate query augmentations. Besides, in HyDE and our LameR, introducing LLMs makes the whole retrieval system free from heavy query-document annotations and outperforms fully-supervised baselines. Specifically, as few-shot Q2D and our zero-shot LameR use extra passages in contrast to zero-shot HyDE, they outperform HyDE significantly. Comparing LameR with few-shot Q2D, with similar LLM's overheads (i.e., reducing our retrieved candidates), the LameR achieves 66.7% nDCG@10 on DL19, still surpassing Q2D.

Overheads in Retrieval. Moving to overheads in retrieval, we compare BM25-based zero-shot

	TREC Deep Leaning Track 2019			TREC Deep Leaning Track 2020		
	MAP	nDCG@10	R@1k	MAP	nDCG@10	R@1k
Zero-shot Retriever						
BM25	30.1	50.6	75.0	28.6	48.0	78.6
LameR$_{\text{bm25}}$ (zero-shot)	47.2	$69.1^{+18.5}$	89.9	45.6	$64.8^{+16.8}$	88.7
Contriever	24.0	44.5	74.6	24.0	42.1	75.4
LameR$_{\text{Contriever}}$ (zero-shot)	41.1	$64.3^{+19.8}$	87.3	38.3	$58.2^{+16.1}$	85.5
Fully-supervised Retriever						
Contriever$^{\text{FT}}$	41.7	62.1	83.6	43.6	63.2	85.8
DPR	36.5	62.2	76.9	41.8	65.3	81.4
ANCE	37.1	64.5	75.5	40.8	64.6	77.6
SimLM	-	71.4	-	-	69.7	-
E5$_{\text{base}}$	-	74.3	-	-	70.7	-
LLM-augmented Fully-supervised Retriever						
HyDE$_{\text{Contriever}^{\text{FT}}}$	-	67.4	-	-	63.5	-
Q2D$_{\text{DPR}}$	-	68.7	-	-	67.1	-
Q2D$_{\text{SimLM}}$	-	$72.9^{+1.5}$	-	-	$71.6^{+1.9}$	-
Q2D$_{\text{E5}_{\text{base}}}$	-	$74.9^{+0.6}$	-	-	$72.5^{+1.8}$	-
LameR$_{\text{SimLM}}$†	**54.9**	$\mathbf{76.5}^{+5.1}$	**91.1**	**55.7**	$\mathbf{75.8}^{+6.1}$	**89.5**

Table 7: Results on DL19/20. †Equipping with our implemented SimLM (Wang et al., 2022a). We mark the 'absolute improvement over base retriever' in superscript for key methods. Ref: DPR (Karpukhin et al., 2020) and E5 (Wang et al., 2022b).

LameR with its counterpart, HyDE, equipped with zero-shot dense retriever. As in Figure 5, benefiting from highly-efficient BM25, LameR, with much higher zero-shot retrieval performance, wins in both retrieval latency and index size.

5.5 LameR meets Dense Retriever

Given promising results w/ a simple BM25, we explore replacing the 2nd-stage BM25 w/ an encoder for dense retrieval. Compared to Eq.(5), the embedding of an augmented query is derived by $\bar{\boldsymbol{q}} = 1/N \cdot \sum_{l\in[1,N]}(\text{Enc}(q;\theta^{(\text{den})}) + \text{Enc}(a^q_l);\theta^{(\text{den})})/2$, where $\theta^{(\text{den})}$ parameterizes $\text{Enc}(\cdot)$.

Consistency across Paradigms. Recall the results in §2: Applying HyDE leads to inconsistent improvement on zero-shot dense retrieval (i.e., Contriever) and term-based retriever (i.e., BM25). So, we'd like to check if LameR can overcome this issue by considering in-domain demonstrations. As listed in Table 7(top), applying LameR to Contriever and BM25 results in similar improvement, verifying its effectiveness in query augmentation by demonstrating in-domain knowledge.

LameR w/ SoTA Retriever. To exploit the performance extreme of LameR, we incorporate a SoTA dense retriever, SimLM (Wang et al., 2022a). As shown in Table 7(bottom), LameR$_{\text{SimLM}}$ significantly improves the SoTA performance on DL19 and DL20 and achieves the best effectiveness. Meantime, compared to Q2D$_{\text{SimLM}}$, our LameR brings significantly higher improvement to SimLM than Q2D (by 3.6% and 4.2% on DL19 and DL20, respectively), not to mention Q2D relying on few-shot demonstration.

6 Conclusion

We propose a retrieval method based merely on an LLM and a simple BM25 algorithm, without any dependence on learnable retrieval models. As such, all the operations are performed in the consistent interface of natural language (i.e., language-based query augmentation and lexicon-overlap retrieval relevance), without the performance bottleneck of a fragile self-supervised model-based retriever. Our evaluations verify the effectiveness of the proposed LameR, supporting the LLM can solely serve as a strong retriever without any in-domain annotated query-document pairs.

Limitation

i) *Instruction sensitivity*: Identical to other prompt-based LLM applications, this work would also be sensitive to the instructions with different LLMs, which may consume a lot of human effort on prompt writing. ii) *Computation Overheads*: As stated in 5.4, although the 2-stage retrieval procedure in LameR is very fast by inheriting BM25, LameR is constrained by calling the LLM for answer generation in terms of computation overheads. To overcome these limitations, in the future we will explore specializing in a relatively smaller LLM for query-augmentation purposes.

References

Akari Asai, Timo Schick, Patrick S. H. Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen-tau Yih. 2022. Task-aware retrieval with instructions. *CoRR*, abs/2211.09260.

Leonid Boytsov, Preksha Patel, Vivek Sourabh, Riddhi Nisar, Sayani Kundu, Ramya Ramanathan, and Eric Nyberg. 2023. Inpars-light: Cost-effective unsupervised training of efficient rankers. *CoRR*, abs/2301.02998.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Yinqiong Cai, Yixing Fan, Jiafeng Guo, Fei Sun, Ruqing Zhang, and Xueqi Cheng. 2021. Semantic models for the first-stage retrieval: A comprehensive review. *CoRR*, abs/2103.04831.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1870–1879. Association for Computational Linguistics.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2021. Overview of the TREC 2020 deep learning track. *CoRR*, abs/2102.07662.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M. Voorhees. 2020. Overview of the TREC 2019 deep learning track. *CoRR*, abs/2003.07820.

Zhuyun Dai, Vincent Y. Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith B. Hall, and Ming-Wei Chang. 2022. Promptagator: Few-shot dense retrieval from 8 examples. *CoRR*, abs/2209.11755.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Dheeru Dua, Emma Strubell, Sameer Singh, and Pat Verga. 2022. To adapt or to annotate: Challenges and interventions for domain adaptation in open-domain question answering. *CoRR*, abs/2212.10381.

Luyu Gao and Jamie Callan. 2022. Long document re-ranking with modular re-ranker. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, pages 2371–2376. ACM.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2022. Precise zero-shot dense retrieval without relevance labels. *CoRR*, abs/2212.10496.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: retrieval-augmented language model pre-training. *CoRR*, abs/2002.08909.

Hangfeng He, Hongming Zhang, and Dan Roth. 2023. Rethinking with retrieval: Faithful large language model inference. *CoRR*, abs/2301.00303.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Towards unsupervised dense information retrieval with contrastive learning. *CoRR*, abs/2112.09118.

Vitor Jeronymo, Luiz Henrique Bonifacio, Hugo Abonizio, Marzieh Fadaee, Roberto de Alencar Lotufo, Jakub Zavrel, and Rodrigo Frassetto Nogueira. 2023. Inpars-v2: Large language models as efficient dataset generators for information retrieval. *CoRR*, abs/2301.01820.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. 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 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6086–6096. Association for Computational Linguistics.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*, pages 2356–2362.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for gpt-3? In *Proceedings of Deep Learning Inside Out: The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, DeeLIO@ACL 2022, Dublin, Ireland and Online, May 27, 2022*, pages 100–114. Association for Computational Linguistics.

Xinxi Lyu, Sewon Min, Iz Beltagy, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022. Z-ICL: zero-shot in-context learning with pseudo-demonstrations. *CoRR*, abs/2212.09865.

Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. 2023. Zero-shot listwise document reranking with a large language model. *CoRR*, abs/2305.02156.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 11048–11064. Association for Computational Linguistics.

Niklas Muennighoff. 2022. SGPT: GPT sentence embeddings for semantic search. *CoRR*, abs/2202.08904.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernández Ábrego, Ji Ma, Vincent Y. Zhao, Yi Luan, Keith B. Hall, Ming-Wei Chang, and Yinfei Yang. 2021. Large dual encoders are generalizable retrievers. *CoRR*, abs/2112.07899.

Yasaman Razeghi, Robert L. Logan IV, Matt Gardner, and Sameer Singh. 2022. Impact of pretraining term frequencies on few-shot numerical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 840–854. Association for Computational Linguistics.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 2655–2671. Association for Computational Linguistics.

Jon Saad-Falcon, Omar Khattab, Keshav Santhanam, Radu Florian, Martin Franz, Salim Roukos, Avirup Sil, Md. Arafat Sultan, and Christopher Potts. 2023. UDAPDR: unsupervised domain adaptation via LLM prompting and distillation of rerankers. *CoRR*, abs/2303.00807.

Tao Shen, Xiubo Geng, Chongyang Tao, Can Xu, Xiaolong Huang, Binxing Jiao, Linjun Yang, and Daxin Jiang. 2022. Lexmae: Lexicon-bottlenecked pretraining for large-scale retrieval. *CoRR*, abs/2208.14754.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 3784–3803. Association for Computational Linguistics.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is chatgpt good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 14918–14937. Association for Computational Linguistics.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

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

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *CoRR*, abs/2212.10509.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022a. Simlm: Pre-training with representation bottleneck for dense passage retrieval. *CoRR*, abs/2207.02578.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022b. Text embeddings by weakly-supervised contrastive pre-training. *CoRR*, abs/2212.03533.

Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query expansion with large language models. *CoRR*, abs/2303.07678.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. 2022. Generate rather than retrieve: Large language models are strong context generators. *CoRR*, abs/2209.10063.

Xueliang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, and Rui Yan. 2020. Knowledge-grounded dialogue generation with pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 3377–3390. Association for Computational Linguistics.

Jiawei Zhou, Xiaoguang Li, Lifeng Shang, Lan Luo, Ke Zhan, Enrui Hu, Xinyu Zhang, Hao Jiang, Zhao Cao, Fan Yu, Xin Jiang, Qun Liu, and Lei Chen. 2022a. Hyperlink-induced pre-training for passage retrieval in open-domain question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 7135–7146. Association for Computational Linguistics.

Yucheng Zhou, Tao Shen, Xiubo Geng, Chongyang Tao, Can Xu, Guodong Long, Binxing Jiao, and Daxin Jiang. 2022b. Towards robust ranker for text retrieval. *CoRR*, abs/2206.08063.

A More Related Work

A.1 Zero-Shot Large-scale Retrieval

In the last years, many research efforts have been dedicated to zero-short retrieval due to its independence of labor-intensive query-document annotations. In contrast to zero-shot transfer that supervisedly trains a retriever in one domain and then evaluates it in another domain (Thakur et al., 2021), we focus on an extremer scenario where no supervised data but the raw target collection is accessible. To handle this scenario, previous works construct pseudo query-document pairs from a target retrieval collection, such as inverse cloze task (Lee et al., 2019), hyperlink prediction (Zhou et al., 2022a), bottlenecked autoencoder (Shen et al., 2022), etc. Given the mined pseudo pairs, they train a retriever upon pre-trained language models, e.g., BERT and RoBERTa, via contrastive learning with stochastic negatives. However, the self-supervised retrievers are only comparable to the lightweight non-parametric term- or lexicon-based retrievers, e.g., BM25 (Robertson and Zaragoza, 2009). Even equipped with LLM-based augmentation (Gao et al., 2022), the self-supervised retrievers still lag behind the retrievers fine-tuned on supervised data. In this work, we discard the inferior self-supervised retrievers but choose the highly generalizable non-parametric retrievers, and propose a brand-new method that integrates LLMs into zero-shot retrieval.

A.2 In-context Learning (ICL)

LLMs can be adapted to new tasks by learning input-label pairs (a.k.a. demonstrations) provided in context, without updates of parameters, which is dubbed in-context learning (Brown et al., 2020). Furthermore, some works seek better in-context demonstrations through retrieval, based on an observation that the demonstrations close to the test input help ICL more effectively (Liu et al., 2022; Rubin et al., 2022). Empirically, ICL, with several demonstrations, remarkably outperforms zero-shot methods across a broad spectrum of tasks, however of a prerequisite for mandatory few-shot examples. Fortunately, recent works (Xie et al., 2022; Razeghi et al., 2022; Min et al., 2022) suggest ICL demonstrations are mainly used to specify input-label domains and formats of the target task, rather than supervision signal only. Sharing a similar inspiration with these works, especially Z-ICL (Lyu et al., 2022), we leverage a retriever for unsupervised demonstrations from a huge collection to specify the domain, intent, and unit. However, we stand with a clean-cut motivation: as we exactly target the retrieval task, the retrieved demonstrations are potential labels (answers), orthogonal to retrieving inputs in previous works (Lyu et al., 2022; Wang et al., 2023). As such, the demonstrations are likely to help generate correct answers by correction or/and summarization with a boosting inspiration.

A.3 Retrieval & Rerank Pipeline

Our two-stage procedure is similar to the retrieval & rerank pipeline (Cai et al., 2021). The retrieval & rerank pipeline first employs a high-efficient retriever to fetch top candidates from a collection and then uses a heavy but effective ranker to rerank the candidates for more precise ranking outputs (Gao and Callan, 2022; Zhou et al., 2022b). But, besides requiring supervised data to train both modules, the rerank module is constrained by the upstream retrieval module. In contrast, LameR always lets its retrieval module direct interact with the collection, free of constraint.

B All Prompts

We did not carefully craft the prompts in this work but directly adapted the prompts in (Gao et al., 2022). We write our prompts of LameR for all the datasets in Table 8.

Prompt for **DL19** and **DL20**.
Give a question "{q}" and its possible answering passages (most of these passages are wrong) enumerated as: \n 1.{c_1^q} \n 2.{c_2^q} \n 3.{c_3^q} ... please write a correct answering passage.

Prompt for **scifact**.
Give a question "{q}" and its possible scientific paper passages (most of these passages are wrong) enumerated as: \n 1.{c_1^q} \n 2.{c_2^q} \n 3.{c_3^q} ... please write a correct scientific paper passage.

Prompt for **arguana**.
Give a question "{q}" and its possible counter-argument passages (most of these passages are wrong) enumerated as: \n 1.{c_1^q} \n 2.{c_2^q} \n 3.{c_3^q} ... please write a correct counter-argument passage.

Prompt for **trec-covid**.
Give a question "{q}" and its possible scientific paper passages (most of these passages are wrong) enumerated as: \n 1.{c_1^q} \n 2.{c_2^q} \n 3.{c_3^q} ... please write a correct scientific paper passage.

Prompt for **fiqa**.
Give a question "{q}" and its possible answering financial article passages (most of these passages are wrong) enumerated as: \n 1.{c_1^q} \n 2.{c_2^q} \n 3.{c_3^q} ... please write a correct answering financial article passage.

Prompt for **dbpedia**.
Give a question "{q}" and its possible answering passages (most of these passages are wrong) enumerated as: \n 1.{c_1^q} \n 2.{c_2^q} \n 3.{c_3^q} ... please write a correct answering passage.

Prompt for **trec-news**.
Give a question "{q}" and its possible relevant passages (most of these passages are wrong) enumerated as: \n 1.{c_1^q} \n 2.{c_2^q} \n 3.{c_3^q} ... please write a correct relevant passage.

Table 8: Our prompts for all datasets.