Exploring the Best Practices of Query Expansion with Large Language Models

Le Zhang[1,2*], **Yihong Wu**[2,*], **Qian Yang**[1,2], **Jian-Yun Nie**[2]
[1] Mila - Québec AI Institute
[2] Université de Montréal

Abstract

Large Language Models (LLMs) are foundational in language technologies, particularly in information retrieval (IR). In this paper, we thoroughly explore the best practice of leveraging LLMs for query expansion. To this end, we introduce a training-free, straightforward yet effective framework called Multi-Text Generation Integration (MUGI). This approach leverages LLMs to generate multiple pseudo-references, which are then integrated with the original queries to enhance both sparse and dense retrieval methods. Additionally, we introduce a retrieval pipeline based on MUGI, which combines the strengths of sparse and dense retrievers to achieve superior performance without the need for costly pre-indexing. Our empirical findings reveal that: (1) Increasing the number of samples from LLMs benefits IR systems; (2) A balance between the query and pseudo-documents, and an effective integration strategy, is critical for high performance; (3) Contextual information from LLMs is essential, even boost a 23M model to outperform a 7B baseline model; (4) Pseudo relevance feedback can further calibrate queries for improved performance; and (5) Query expansion is widely applicable and versatile, consistently enhancing models ranging from 23M to 7B parameters. Our code and all generated references are made available at https://github.com/lezhang7/Retrieval_MuGI.

1 Introduction

Information retrieval (IR) is crucial for extracting relevant documents from large databases, serving as a key component in search engines, dialogue systems (Yuan et al., 2019; Qian et al., 2021), question-answering platforms (Qu et al., 2020; Zhang et al., 2023b; Yang et al., 2023), recommendation systems (Fan et al., 2023; Zhang et al., 2023a), and Retrieval Augmented Generation (RAG) framework (Lewis et al., 2021; Izacard and Grave, 2020; Zhang et al., 2022; Liu, 2022).

Query expansion, a key technique for enhancing IR efficacy (Abdul-Jaleel et al., 2004; Robertson and Jones, 1976; Salton, 1971), traditionally employs Pseudo-Relevance Feedback (PRF) (Li et al., 2022; Lavrenko and Croft, 2017) from initial retrieval results. However, its effectiveness is constrained by the quality of these coarse results. Recently, Large Language Models (LLMs), such as ChatGPT, have demonstrated exceptional capabilities in language understanding, knowledge storage, and reasoning (Brown et al., 2020; Touvron et al., 2023). Motivated by these advancements, some studies have explored leveraging LLMs for zero-shot query expansion (Ma et al., 2023b; Gao et al., 2022; Wang et al., 2023b). While these methods have shown empirical effectiveness, they also present certain limitations.

LameR (Shen et al., 2023) generates potential answers by utilizing LLMs to rewrite BM25 candidates for expansion. However, its performance is highly dependent on the quality of the initial retrieval. Both HyDE (Gao et al., 2022) and Query2Doc (Wang et al., 2023b) leverage the knowledge stored in LLMs. While HyDE demonstrates effective performance with contriver, it performs poorly with lexical-based retrievers (Shen et al., 2023). Conversely, Query2Doc is effective with both sparse and dense retrieval methods, but strong models may not benefit as much as weaker ones (Li et al., 2024; Weller et al., 2023). Moreover, the integration and balance between pseudo references and queries are under-explored in these studies.

To address these limitations, we explore best practices for utilizing query expansion with LLMs for information retrieval. In this paper, we delve into several specific research questions: **RQ1**: Is there a universal query expansion method that effectively serves both lexical-based and neural-based

[*] equal contribution

Findings of the Association for Computational Linguistics: EACL 2024, pages 1872–1883

retrievers, applicable to both weak and strong models without prior constraints? **RQ2**: Are multiple pseudo-references more beneficial than a single one? **RQ3**: How can the query and pseudo-references be balanced for lexical-based retrievers? **RQ4**: What is the most effective method for integrating multiple pseudo-references with a query in dense retrievers?

We introduce a framework named **Mu**lti-**T**ext **G**eneration **I**ntegration (**MUGI**) to explore these questions. MUGI employs a zero-shot approach to generate multiple pseudo-references from LLMs, integrating them with queries to enhance IR efficiency. Our empirical experiments demonstrate that: (1) Increasing the number of samples from LLMs benefits IR systems. (2) MUGI demonstrates versatility and effectiveness across both lexical and dense retrievers and models of various sizes. Remarkably, it enables a 23M-parameter dense retriever to outperform a larger 7B baseline. (3) MUGI proposes an adaptive reweighting strategy that considers the lengths of both the pseudo-references and the query, critically improving the performance of lexical retrievers. (4) MUGI investigates different integration strategies and proposes contextualized pooling, which has been overlooked in previous methods. Additionally, drawing inspiration from the Rocchio algorithm (Schütze et al., 2008), MUGI implements a calibration module that leverages pseudo relevance feedback to further enhance IR performance. Notably, using ChatGPT4, MUGI significantly enhances BM25 performance, with an 18% improvement on the TREC DL dataset and 7.5% on BEIR, and boosts dense retrievers by over 7% on TREC DL and 4% on BEIR.

2 Related Work

Information Retrieval focuses on the efficient and effective retrieval of information in response to user queries. Best Matching 25 (BM25) (Robertson et al., 1994) advances beyond earlier probabilistic models by incorporating document length normalization and non-linear term frequency scaling, thereby enhancing the alignment of queries with documents. Dense retrievers such as DPR (Karpukhin et al., 2020) employ deep neural networks to identify semantic relationships between queries and documents by measuring the cosine similarity of their text embeddings.

Existing efficient IR systems typically use a retrieval & rerank pipeline (Nogueira and Cho, 2020; Karpukhin et al., 2020; Guo et al., 2022; Reimers and Gurevych, 2019): Initially, a retrieval mechanism, such as BM25 or a bi-encoder, identifies a broad set of potentially relevant documents. Subsequently, a stronger ranker, usually a cross-encoder, meticulously scores the relevance of these documents, enhancing the precision of the final results.

LLMs for IR The use of LLMs in IR falls into two primary categories (Zhu et al., 2023): fine-tuning LLMs as retrieval models and employing them for zero-shot IR. This paper concentrates on zero-shot IR, where typical approaches involve leveraging the reasoning capabilities of LLMs for direct document ranking (Sun et al., 2023; Ma et al., 2023b) or relevance assessment (Sachan et al., 2022). While effective, these methods are limited by LLMs' input length constraints, making them better suited for the rerank phase.

Another line of research focuses on using LLMs to synthesize additional high-quality training datasets to improve existing models (Bonifacio et al., 2022; Izacard et al., 2021; Wang et al., 2023a; Jeronymo et al., 2023). Other works, such as HyDE (Gao et al., 2022), query2doc (Wang et al., 2023b), and LameR (Shen et al., 2023), explore query expansion. They leverage LLMs to create pseudo-references or potential answers, enhancing queries for better retrieval outcomes.

MuGI is a query expansion framework that leverages LLMs to enhance queries. Unlike previous works, which are limited by inherent constraints, MuGI offers broader applicability and versatility as it seamlessly integrates with both lexical and dense retrievers. By utilizing and intergrating a wealth of contextualized information from multiple references, MuGI surpasses existing techniques in both in-domain and out-of-distribution evaluations by more effectively capturing essential keywords and enriching the background context.

3 Method

We begin by discussing IR preliminaries and then introduce our MuGI framework.

3.1 Preliminaries

Non-parametric Lexical-based Methods BM25 is a fundamental non-parametric lexical method that calculates document relevance using term frequency (TF) and inverse document

frequency (IDF):

$$\sum_{i=1}^{n} \frac{\text{IDF}(q_i)\text{TF}(q_i, D)(k_1 + 1)}{\text{TF}(q_i, D) + k_1(1 - b + b\frac{|D|}{\text{avgdl}})}, \quad (1)$$

where q_i are query terms, $\text{TF}(q_i, D)$ is term frequency, $\text{IDF}(q_i)$ is inverse document frequency, $|D|$ is document length, avgdl is average document length, and k_1 and b are tunable parameters.

Neural Dense Retrieval Methods Dense retrieval leverages deep learning to identify semantic similarities between queries and documents by encoding them into high-dimensional embeddings, typically measured by (Huang et al., 2013):

$$\text{Sim}(q, D) = \frac{\boldsymbol{f}(q)^\top \boldsymbol{f}(D)}{\|\boldsymbol{f}(q)\|\|\boldsymbol{f}(D)\|}, \quad (2)$$

where $\boldsymbol{f}(\cdot)$ maps text to embedding space $\mathbb{R}^d$. BM25 is fast and generalizes well, suited for sparase retrieval, while dense retrieval excels at capturing semantic connections but is slower and less generalized due to neural network dependency.

3.2 Multi-Text Generation Integration

Recognizing that both lexical-based and dense retrieval methods depend on a certain degree of information overlap between the query and document, we introduce the **Multi-Text Generation Integration** (MUGI) method. This approach aims to augment the query's information content by leveraging multiple samplings from LLMs. MUGI enriches queries with additional background information and broadens the keyword vocabulary to encompass out-of-domain terms, thereby bridging the semantic gap between queries and documents on both lexical-based and dense retrievers.

Upon receiving a query q, MUGI initially applies a zero-shot prompt (see fig. 1) technique to generate a set of pseudo-references, denoted as $\mathcal{R} = \{r_1, r_2, r_3, ..., r_n\}$, which are then integrated with query for subsequent IR operations.

Zero-shot Generation Prompt

```
You are PassageGenGPT, an AI capable of generating concise,
informative, and clear pseudo passages on specific topics.

Generate one passage that is relevant to the following
query: '{query}'. The passage should be concise,
informative, and clear
```

Figure 1: Zero-Shot Prompting for Relevant Passage Generation: It emphasizes generating contextually relevant content to enhance background knowledge density for multiple-text integration.

3.2.1 MUGI for BM25

This component evaluates relevance by analyzing lexical overlaps between the query and references. Given the longer lengths of documents compared to queries and BM25's sensitivity to word frequency, achieving a careful balance to ensure the appropriate influence of each element in text is crucial. The variation in the lengths of queries and passages makes the constant repetition of query used in previous studies, which typically handles single pseudo-references, ineffective (Wang et al., 2023b; Shen et al., 2023), particularly when dealing with multiple references.

To address this issue, we implement an adaptive reweighting strategy that adjusts according to the length of the pseudo-references. This adjustment is governed by a factor β, as illustrated by the following equation:

$$\lambda = \left\lfloor \frac{\text{len}(r_1) + \text{len}(r_2) + \ldots + \text{len}(r_n))}{\text{len}(q) \cdot \beta} \right\rfloor. \quad (3)$$

Since BM25 does not account for word order, we enhance the query by repeating query λ times and concatenating it with all pseudo-references:

$$q_{\text{sparse}} = \text{concat}(q * \lambda, r_1, r_2, r_3..., r_n). \quad (4)$$

This enhanced query is then processed by BM25 to produce the ranking results I_{bm25}.

3.2.2 MUGI for Dense Retriever

MUGI also enhances dense retrievers, specifically bi-encoders. In this section, we discuss how to integrate pseudo-references with queries. We present two approaches to integrate queries with pseudo-references to obtain a contextualized query embedding.

I. **Concatenation** has been commonly used in prior studies (Shen et al., 2023; Wang et al., 2023b), where the query is simply concatenated with all references as in BM25:

$$q_{\text{cat}} = \text{concat}(q, r_1, r_2, ..., r_n). \quad (5)$$

This enhanced query is then processed by the dense retriever $\boldsymbol{f}$ to produce embeddings, i.e., $\boldsymbol{e}_{\text{cat}} = \boldsymbol{f}(q_{\text{cat}})$. However, as the number and length of references increase, the typical input length limitation of 512 tokens can hinder the integration process. Consequently, only one to two passages can be incorporated into q_{cat}.

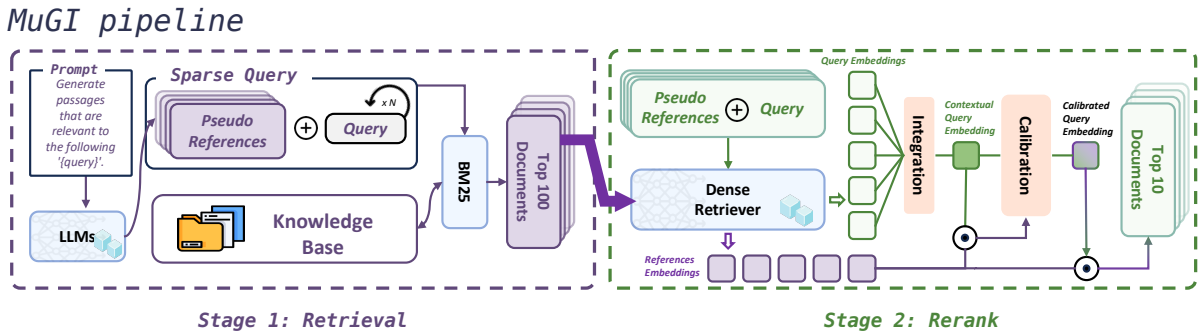

Figure 2: **Method overview of MuGI.** Left part is initial retrieval using BM25 for initial retrieval, right part indicates re-rank output from first stage using a dense retriever.

II. **Feature Pooling** addresses the model's input length limitations, particularly when multiple references are involved. A straightforward method is to average the embeddings in the feature space, as demonstrated by HyDE (Gao et al., 2022):

$$\boldsymbol{e}_{\text{mean-pool}} = \frac{\boldsymbol{f}(q) + \sum_i \boldsymbol{f}(r_i)}{n+1}. \quad (6)$$

Empirically, we use another variant termed contex-pool:

$$\boldsymbol{e}_{\text{contex-pool}} = \frac{\sum_i \boldsymbol{f}(\text{concat}(q, r_i)}{n}. \quad (7)$$

We then calculate the similarity between the query and all documents, ranking them accordingly. We denote these rankings as I_{dense}.

3.3 MuGI Pipeline

While recent neural dense retrievers often outperform traditional sparse retrieval methods like BM25, we observe that pre-indexing and storing embeddings is computationally and memory intensive for large-scale databases, particularly with the advent of larger models. For instance, encoding the academic-purpose MS-MARCO dataset with 8.8 million passages using a 7B LLM-based text encoder requires over 35 hours on a single A100 GPU, even with Flash Attention (Dao et al., 2022), and consumes 120GB of storage in float16 precision. This process becomes impractical when new models are introduced, as it requires re-indexing the entire dataset. Given that MuGI enhances both BM25 and dense retrievers, it is natural to combine them to leverage the strengths of generated references for Information Retrieval.

To address these challenges, we introduce a fast and high-quality retrieval pipeline, termed the MuGI Pipeline (see fig. 2). This pipeline begins by retrieving the top-k references using BM25+MuGI, followed by reranking these references with enhanced query embeddings from a bi-encoder model. Specifically, we restrict the bi-encoder's search scope to the top 100 references retrieved by BM25+MuGI, which is manageable enough to enable efficient online reranking. This approach enhances flexibility, as it facilitates the integration of the latest models into the retrieval system without the computational overhead of re-indexing the entire dataset. By reusing the lightweight BM25 for the initial retrieval stage and performing reranking with a neural dense retriever, the MuGI pipeline effectively mitigates the heavy computational load of dense retrieval. Moreover, with the support of large language models for query expansion, we observe that MuGI pipeline achieves not only significantly faster search speeds but also retrieval results comparable to those obtained by a much slower process of directly applying a dense retriever on the full database.

It is important to note that the MuGI pipeline serves as an initial retrieval method, complementing rather than contradicting the retrieval-then-rerank framework, since it utilizes a bi-encoder solely during the retrieval stage. To enhance retrieval accuracy, a subsequent cross-encoder can be employed to rerank the outputs of the MuGI pipeline.

Calibration The MuGI pipeline utilizes retrieval results from BM25, which provides pseudo-reference information that can be further leveraged to refine query embeddings using relevance feedback techniques. To achieve this, we consider the Rocchio algorithm (Schütze et al., 2008), which refines the query vector using relevance feedback

as follows :

$$\boldsymbol{e}'_q = a\boldsymbol{e}_q + \frac{b}{|\mathcal{D}_+|}\sum_{\boldsymbol{d}_i \in \mathcal{D}_+} \boldsymbol{d}_i - \frac{c}{|\mathcal{D}_-|}\sum_{\boldsymbol{d}_j \in \mathcal{D}_-} \boldsymbol{d}_j, \quad (8)$$

where $\boldsymbol{e}'_q$ is the calibrated query vector, $\boldsymbol{e}_q$ is the original query vector, $\boldsymbol{d}_i, \boldsymbol{d}_j$ are document vectors, $\mathcal{D}_+/\mathcal{D}_-$ is the set of positive/negative documents, and a, b, c are weights.

Since the Rocchio algorithm is primarily designed for bag-of-words models and does not directly align with the architecture of neural retrievers, we propose an adapted post-processing operation suitable for neural dense retrievers.

Specifically, we first run the MUGI pipeline and get top 100 BM25 results I_{bm25} as well as initial dense rank results I_{init} as shown in fig. 2. We construct a negative feedback set $\mathcal{N}$ consisting of the last n documents from the BM25 retrieval results, and a positive feedback set $\mathcal{R}+$, which includes all generated references along with the K-reciprocal documents—defined as the intersection of the top-K documents from both I_{bm25} and I_{init}. The adjusted query calibration is then performed as follows:

$$\boldsymbol{e}'_q = \frac{1}{W}\Big(\sum_{r \in \mathcal{R}_+} \boldsymbol{f}(\text{concat}(q, r)) - \alpha \sum_{n' \in \mathcal{N}} \boldsymbol{f}(n')\Big), \quad (9)$$

where $W = |\mathcal{R}_+| + |\mathcal{N}|$ is the total number of positive and negative feedback documents, and α is a weight factor controlling the influence of negative feedback. The refined query vector $\boldsymbol{e}'_q$ is then used for a reranking process among I_{bm25} to produce the final retrieval results. It is worth noting that this calibration process does not introduce additional computational overhead, as it directly operates on the output embeddings, ensuring efficiency while improving the quality of retrieval.

4 Experiments

In this section, we provide answers for **RQ1** and **RQ2** in §4.2 and §4.3 and answers for **RQ3** and **RQ4** in §4.4.

4.1 Setup

Implementation Details We employ ChatGPT and Qwen (Bai et al., 2023) to generate pseudo-references. For BM25 searches, we use the Pyserini toolkit (Lin et al., 2021) with default settings. We select multiple dense bi-encoders based on their performance in the MTEB leaderboard (Muennighoff et al., 2022), including: ALL-MINILM-L6-V2, ALL-MPNET-BASE-V2, BGE-LARGE-EN-V1.5 (Xiao et al., 2023), GTE-LARGE-EN-V1.5 (Li et al., 2023), EMBER-V1, and E5-MISTRAL-INSTRUCT (Wang et al., 2023a), which vary in size from 23M to 7B parameters. We also incorporate strong cross-encoders, including MonoT5 (Nogueira et al., 2020) and RankLLaMA (Ma et al., 2023a) as baselines.

In our experiments, unless specified otherwise, MUGI uses a $\beta = 4$ for BM25 and a context feature pool with a calibration factor of $\alpha = 0.2$ and 4-reciprocal for dense retrieval. Each retrieval process involves using $n = 5$ generated pseudo-references from ChatGPT-4-turo-1106. All experiments are conducted with 1 NVIDIA A100 GPU.

Evaluation Following (Wang et al., 2023b), we adopt nDCG@10 as the metric and evaluate on the TREC DL19 (Craswell et al., 2020) and DL20 (Craswell et al., 2021) datasets for in-domain analysis, and on nine low-resource datasets from BEIR (Thakur et al., 2021) for out-of-distribution (OOD) evaluation.

4.2 Result

Applicability is crucial for query expansion methods. Previous approaches like HyDE targeted only dense retrievers, while Query2Doc yields inconsistent results across models of varying capacities (Li et al., 2024; Weller et al., 2023). Conversely, our MUGI framework enhances both BM25 and dense retrievers across all evaluations, as demonstrated in Table 1. Unlike Query2Doc, MUGI consistently enhances models across a wide range of sizes, from 23M to 7B parameters, demonstrating its broad applicability.

MUGI for Sparse Retriever For sparse retrieval evaluation, we include strong baselines such as HyDE (Gao et al., 2022), Query2Doc (Wang et al., 2023b), and LameR (Shen et al., 2023), alongside compact dense retrievers like ANCE (Xiong et al., 2020) and DPR (Karpukhin et al., 2020). The results, presented in Table 1, show that MUGI outperforms all baseline models and existing query expansion techniques on the TREC DL dataset. Specifically, it boosts BM25 performance by 19.8% and 16.4% in nDCG@10 for the TREC DL 19/20 datasets, respectively.

MUGI also shows substantial improvements on the BEIR dataset, where queries are typically short

Methods	In-domain		Out-of-Domain								
	DL19	DL20	Covid	NFCorpus	Touche	DBPedia	SciFact	Signal	News	Robust04	BEIR (Avg)
ANCE †	64.5	64.6	65.4	23.7	24.0	28.1	50.7	24.9	38.2	39.2	36.8
DPR†	62.2	65.3	33.2	18.9	13.1	26.3	31.8	15.5	16.1	25.2	22.5
DocT5query†	64.2	-	71.3	32.8	34.7	33.1	67.5	30.7	42.0	43.7	44.5
HyDE†$_{\text{Contriever}}$	61.3	57.9	59.3	-	-	36.8	69.1	-	44.0	-	-
RepLLaMA-7B †	74.3	72.1	84.7	37.8	30.5	43.7	75.6				
Sparse + Query Expansion											
BM25	50.6	48.0	59.5	30.8	44.2	31.8	67.9	33.1	39.5	40.7	43.4
+ *RM3*†	52.2	47.4	-	-	-	-	-	-	-	-	-
+ *Query2Doc*†	66.2	62.9	72.2	34.9	39.8	37.0	68.6	-	-	-	-
+ *LameR*†	67.1	62.7	72.5	-	-	38.7	73.5	-	49.9	-	-
+ *MuGI (ChatGPT-3.5)*	70.4	63.9	69.7	36.0	46.3	41.2	72.0	35.8	46.6	49.7	49.6
+ *Query2Doc (ChatGPT-4)*‡	67.0	63.2	70.8	36.0	44.2	39.1	73.3	-	-	-	-
+ *LameR (ChatGPT-4)*‡	70.1	64.2	-	-	-	-	-	-	-	-	-
+ *MuGI (ChatGPT-4)*	**70.4**$^{+19.8}$	**64.4**$^{+16.4}$	**72.9**$^{+13.4}$	**37.4**$^{+6.6}$	**46.1**$^{+1.9}$	**42.7**$^{+10.9}$	**74.0**$^{+6.1}$	**36.0**$^{+2.9}$	**50.0**$^{+10.5}$	**49.2**$^{+8.5}$	**51.0**$^{+7.6}$
Dense + Query Expansion (ChatGPT-4)											
all-MiniLM-L6-v2	63.7	65.6	50.7	30.9	19.7	33.4	65.1	-	-	-	-
+ *Query2Doc* ‡	70.8	69.9	68.2	34.7	23.2	40.4	70.5	-	-	-	-
+ *MuGI*	**75.3**	**71.5**	**75.2**	**35.9**	**24.4**	**42.7**	**72.2**				
AMB-v2	66.7	67.2	55.6	33.5	23.6	34.6	65.7	-	-	-	-
+ *Query2Doc*‡	73.7	73.5	74.2	38.0	23.4	41.2	70.7	-	-	-	-
+ *MuGI*	**76.8**	**75.7**	**75.3**	38.0	**26.7**	**42.6**	**71.4**	-	-	-	-
Ember-v1	72.7	70.5	76.5	38.8	27.0	42.9	75.2	-	-	-	-
+ *Query2Doc*‡	74.1	73.2	81.8	39.5	25.1	43.6	77.2	-	-	-	-
+ *MuGI*	**76.9**	**74.5**	**82.1**	39.5	**28.5**	**44.9**	**77.3**	-	-	-	-
GTE-Large-EN-v1.5	71.6	72.2	76.1	36.9	24.5	44.6	79.6	-	-	-	-
+ *Query2Doc*‡	76.2	74.3	82.9	39.0	27.5	43.7	81.1	-	-	-	-
+ *MuGI*	**78.2**	**75.0**	**84.3**	**39.7**	**27.6**	**45.9**	**81.5**	-	-	-	-

Table 1: **Retrieval Results (nDCG@10)**. The top half presents MuGI results for sparse retrievers, while the bottom half shows MuGI results for dense retrievers. The best-performing results are highlighted in bold. †indicates cited results, and ‡represents reproduced results. 'ChatGPT-4' refers to ChatGPT4-1106, and 'ChatGPT-3.5' corresponds to ChatGPT3.5-turbo.

and ambiguous. By integrating multiple pseudo-references, MuGI effectively enriches the context, leading to a 7.6% enhancement over the baseline BM25 and outperforming other query expansion strategies across all tested datasets.

MuGI for Dense Retriever For dense retrieval, we employ FAISS (Douze et al., 2024) to perform inner product search across four different-sized bi-encoders, as presented in Table 1. The MuGI pipeline significantly enhances performance for all models in both in-domain and out-of-domain benchmarks. We also observe a consistent trend where MuGI outperforms the single pseudo-reference method by a considerable margin. Notably, with MuGI, the smaller 434M GTE model outperforms the larger 7B RepLLaMA model in all in-domain and most out-of-domain tasks. This suggests that compact models, when supplemented with pseudo-references generated by larger models, can serve as highly effective retrievers.

4.3 MuGI Pipeline Retrieval Results

MuGI pipeline is a fast and effective retrieval pipeline combining benefit of sparse retriever and dense retriever, it involves reordering passages initially retrieved by sparse retriever using advanced neural models. Our findings, shown in Table 2, underscore the MuGI pipeline's substantial enhancement of dense retrievers' performance. The *BM25 rerank* baseline reranks the top 100 references from BM25, while *Query2Doc* applies the Query2Doc[1] method in both sparse retrieval and reranking phases. The *MuGI pipeline*, outlined in fig. 2, integrates calibration into the reranking process.

Integrating MuGI into the reranking process consistently boosts performance, outperforming Query2Doc in all scenarios. This integration leads to over a 7% improvement in ***in-domain evaluations*** across various model sizes compared with BM25 re-rank baseline. Notably, the compact MiniLMv2 model, with just 23M parameters, surpasses the larger 7B E5-Mistral baseline and even 3B cross-encoder monoT5 by achieving an average of 11% improvement over the baseline on the DL 19/20 datasets.

In ***OOD scenarios***, MuGI continues to outperform BM25 re-rank baseline, with gains exceeding 4% across all models. Specifically, the 23M MiniLMv2 shows that compact bi-encoders can be an effective and robust retriever when equipped

[1]Only the TREC DL dataset is released for Query2Doc.

Methods	#Param	In-domain		Out-of-Domain								
		DL19	DL20	Covid	NFCorpus	Touche	DBPedia	SciFact	Signal	News	Robust04	BEIR(Avg)
Dense Cross-Encoder + BM25 re-rank												
monoT5 †	220M	71.5	67.0	78.3	37.4	30.8	42.4	73.4	31.7	46.8	51.7	49.1
monoT5†	3B	71.8	68.9	80.7	39.0	32.4	44.5	76.6	32.6	48.5	56.7	51.4
RankLLaMA‡	7B	75.6	77.4	85.2	30.3	40.1	48.3	73.2	-	-	-	-
Cohere Rerankv2 †	API	73.2	67.1	81.8	36.4	32.5	42.5	74.4	29.6	47.6	50.8	49.5
Dense Bi-Encoder + Query Expansion (ChatGPT-4)												
all-MiniLM-L6-v2	23M											
+ *BM25 re-rank*		64.2	60.8	73.6	30.8	26.2	35.9	67.6	28.8	**52.0**	51.4	45.8
+ *Query2Doc*		73.1	70.2	76.6	34.2	29.1	39.6	69.2	-	-	-	-
+ *MuGI Pipeline*		**75.4**$^{+11.2}$	**72.1**$^{+11.3}$	**81.0**$^{+7.4}$	**37.6**$^{+6.8}$	**34.1**$^{+7.9}$	**45.5**$^{+9.6}$	**74.8**$^{+7.2}$	**28.9**$^{+0.1}$	51.4	**55.2**$^{+3.8}$	**51.0**$^{+5.2}$
AMB-v2	110M											
+ *BM25 re-rank*		68.3	64.2	75.8	34.3	28.9	37.6	68.0	29.8	52.0	51.4	47.2
+ *Query2Doc*		75.3	74.0	78.9	36.2	29.5	38.9	71.4	-	-	-	-
+ *MuGI Pipeline*		**77.6**$^{+9.3}$	**75.1**$^{+10.9}$	**82.6**$^{+6.8}$	**39.2**$^{+4.9}$	**30.3**$^{+1.4}$	**45.7**$^{+7.8}$	**73.9**$^{+5.9}$	**30.2**$^{+0.4}$	**55.2**$^{+3.2}$	**58.2**$^{+6.8}$	**51.9**$^{+4.7}$
Ember-v1	335M											
+ *BM25 re-rank*		71.3	64.5	80.3	35.8	31.6	41.7	75.2	32.0	48.8	51.3	49.6
+ *Query2Doc*		73.5	71.1	82.1	37.9	33.8	42.9	79.2	-	-	-	-
+ *MuGI Pipeline*		**78.3**$^{+7.0}$	**73.3**$^{+8.8}$	**84.7**$^{+4.4}$	**40.7**$^{+4.9}$	**35.8**$^{+4.2}$	**46.6**$^{+4.9}$	**78.3**$^{+3.1}$	**32.6**$^{+0.6}$	**54.0**$^{+5.2}$	**58.0**$^{+6.7}$	**53.8**$^{+4.2}$
GTE-Large-EN-v1.5	434M											
+ *BM25 re-rank*		70.4	66.0	81.3	35.2	28.3	41.5	78.9	29.8	50.9	54.9	50.1
+ *Query2Doc*		73.6	72.8	83.2	37.7	29.9	45.0	79.9	-	-	-	-
+ *MuGI Pipeline*		**77.9**$^{+7.5}$	**74.1**$^{+8.1}$	**86.9**$^{+5.6}$	**40.8**$^{+5.6}$	**31.9**$^{+3.6}$	**47.4**$^{+5.9}$	**80.6**$^{+1.7}$	28.0	**54.0**$^{+3.1}$	**58.5**$^{+3.6}$	**53.5**$^{+3.4}$
E5-Mistral-instruct	7B											
+ *BM25 re-rank*		70.0	66.7	81.4	36.0	29.5	42.4	75.8	32.8	53.0	52.8	50.5
+ *Query2Doc*		73.8	71.4	83.2	39.1	32.4	45.3	76.8	-	-	-	-
+ *MuGI Pipeline*		**77.3**$^{+7.3}$	**+74.9**$^{+8.2}$	**+85.6**$^{+4.2}$	**+41.3**$^{+5.3}$	**+35.8**$^{+6.3}$	**47.3**$^{+4.9}$	**+77.9**$^{+2.1}$	**+34.5**$^{+1.7}$	**55.1**$^{+2.1}$	**59.2**$^{+6.4}$	**54.6**$^{+4.1}$

Table 2: **MuGI Pipeline Results (nDCG@10) on TREC and BEIR.** Best performing are marked bold. MuGI pipeline suggests application of MuGI on both sparse retrieval and dense retrieval as shown in fig. 2. ‡RankLLaMA rerank based on top 200 references from RepLLaMA.

with sufficient contextual information. With the MuGI application, it consistently outperforms the larger 7B E5-Mistral BM25 re-rank baseline and approaches the performance of the cross-encoder monoT5-3B. However, only modest improvements are noted on the Signal dataset, likely due to the ambiguous queries like *"A stadium for Hughes"* and *"Revilla wants 34 defense witnesses"*, which may not provide sufficient context for LLMs.

Notably, the combination of sparse and dense retrievers in the MuGI pipeline proves to be significantly faster and more cost-effective for retrieval inference compared to using dense retrievers with FAISS directly on whole database. This efficiency arises from the fact that MuGI eliminates the need for pre-indexing and avoids performing inner product searches across the entire database. Additionally, the pipeline shows promising performance, particularly in out-of-domain evaluations, where it exceeds the performance of direct retrieval with dense retrievers. We attribute this strong performance to the advantages of sparse retrievers in zero-shot scenarios, where dense retrievers may be less effective. The results from Table 1 suggest that proposed MuGI pipeline not only enhances speed and reduces computational costs but also delivers superior retrieval results, while maintaining the flexibility to integrate seamlessly with more recent and advanced models.

4.4 Explore Best Practice of Query Expansion

Explore Reweight for BM25. Reweighting queries and pseudo references is crucial for optimizing the sensitive BM25 algorithm. To explore reweighting strategy, we experiment with pseudo-references generated by GPT-4 under the MuGI framework in two settings: 1) constant repetition of query for t times and 2) our proposed adaptive reweighting with reweight factor β.

Fig. 3 displays the average scores for the TREC DL+BEIR dataset. It shows that constant repetition, overlooking variations in pseudo-reference lengths, is not a general solution. For instance, repeating a query five times, as suggested in (Wang et al., 2023b), is only optimal with a single pseudo-reference. When dealing with multiple pseudo-references, a higher repetition t is needed to achieve better performance. The main drawback of this method is its failure to adjust to the different lengths of references, resulting in inconsistent performances when using a fixed repetition rate t. Moreover, finding an effective t for multiple references requires considerable effort. Our adaptive reweighting strategy dynamically adjusts to query and reference lengths, optimizing word frequency in the enhanced query which is important

Figure 3: **BM25 + MuGI** Reweighting Strategy Results (nDCG@10) on average scores of TREC DL + BEIR. The left panel illustrates the constant repetition of the query, while the right panel displays our adaptive reweighting strategy with various β values. The Y-axis represents the number of pseudo-references used.

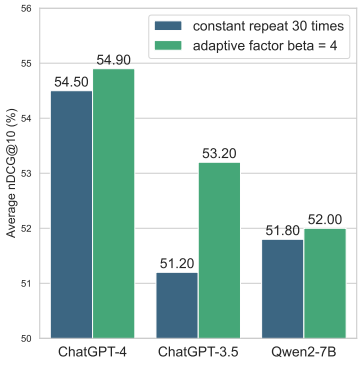

Figure 4: **BM25 + MuGI over various LLMs with different reweight strategy** with 5 References Results (nDCG@10) .

Model	Params	Method	TREC DL	BEIR
MiniLM-L6-v2	23M	concat	71.6	48.1
		contex-pool	**73.6** (+2.0)	**49.5** (+1.4)
AMB-v2	110M	concat	75.7	50.3
		contex-pool	**76.1** (+0.4)	**50.9** (+0.6)
Ember-v1	335M	concat	74.0	52.0
		contex-pool	**74.8** (+0.8)	**52.5** (+0.5)
BGE-Large-EN-v1.5	335M	concat	74.3	51.2
		contex-pool	**74.8** (+0.5)	**51.6** (+0.4)
E5-Mistral-instruct	7B	concat	74.0	53.0
		contex-pool	**75.8** (+1.8)	**53.6** (+0.6)

Table 3: Evaluation Results (nDCG@10 %) of different sized Models with distinct integration approach.

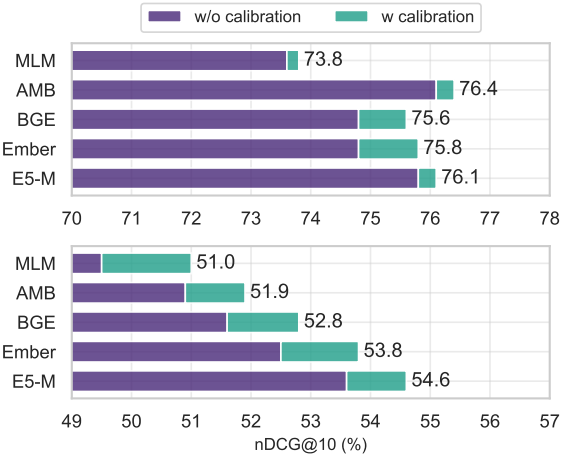

Figure 5: Calibration Ablation $\alpha = 0.2$ (nDCG@10). (Top) In domain TREC DL evaluation; (Bottom) BEIR OOD evaluation. E5-M is E5-Mistral-instruct, BGE is BGE-Large-EN-v1.5, MLM is all-MiniLM-L6-v2, Ember is Ember-v1.

for lexical-based retrievers. This method effectively manages various numbers of references without needing repeated trials to find the optimal repetition ratio. For instance, setting $\beta = 4$, as shown in fig. 3 (right), consistently yields strong performance across different numbers of reference passages, eliminating the need for a grid search to adjust repetition times t as the number of passages varies.

The adaptive reweighting also maintains robust performance across diverse LLMs, accommodating their varied output lengths. We applied the optimal configuration from fig. 3, which includes a constant repetition of $t = 30$ and $\beta = 4$ across 5 passages from various LLMs. As demonstrated in fig. 4, this adaptive approach consistently outperforms constant repetition. For instance, against ChatGPT-3.5, known for its shorter responses, our method effectively compensates for the inadequate query weighting of constant repetition, ensuring strong performance across different models.

Explore Integration Approach for Dense Retriever. We explored two methods for integrating information from pseudo-references with a query: ***concatenation*** in the input space and ***feature pooling*** in the feature space. Our experiments with MUGI pipeline, conducted **without calibration**, indicate that feature pooling consistently outperforms simple concatenation, as detailed in Table 3. The primary drawback of concatenation is truncation; given most models' input length limit of 512 tokens, only 1-2 pseudo-references can be accommodated, limiting the utilization of multiple references.

Additionally, concatenation increases computational costs due to its quadratic complexity. For example, with n references each of average length d, the complexity of concatenation is $O(n^2d^2)$, compared to $O(nd^2)$ for feature pooling.

Effect of Calibration The new calibration method leverages hard negative reference feedback to refine query embeddings in the feature space.

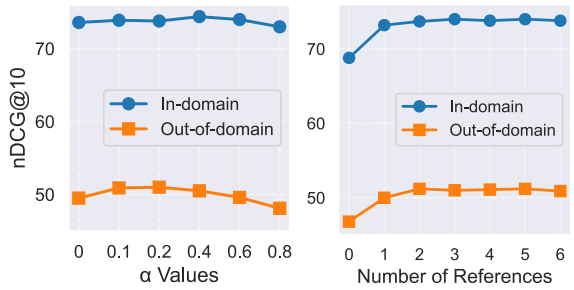

Figure 6: (Left) Ablation on α in calibration. (Right) Ablation on the number of generated pseudo references for MiniLM-L6. Zero references represent directly rerank on top100 BM25.

As shown in fig. 5, it consistently boosts performance in both in-domain and OOD evaluations, with notable gains in OOD scenarios.

5 Analysis

5.1 Ablation Study

In this section, we analyze and conduct ablation studies on MUGI using the all-MiniLM-L6-v2 and references generated by GPT-4 within the full MUGI pipeline settings described in §4.1.

Impact of Number of Pseudo References The fundamental premise of MUGI is that multiple references generated by LLMs can provide contextual information and key words or relevant patterns that enhance queries. Consequently, the critical factor in our framework is the number of pseudo references used. Both sparse retrieval (fig. 3) and dense retrieval (fig. 6) show performance improvements with an increasing number of references; however, gains plateau at five. This suggests that the language model's capability to generate key terms reaches its limit at this point.

Impact of α in Calibration Figure 6 indicates that the calibration parameter α offers benefits, with performance peaking at 0.2 and then declining. The impact of calibration is more pronounced in OOD evaluations, in line with observations from fig. 5.

Impact of Large Language Models We assessed the MUGI framework across various LLMs, specifically examining BM25 (fig. 4) and dense retrievers (Table 4). Our findings highlight that: 1) stronger LLMs consistently produce better results; 2) GPT models outperform others, likely due to their expansive knowledge bases. Although GPT-3.5 generally underperforms relative to Qwen2-7B on most open leaderboards (Zheng et al., 2023; Park, 2023), it demonstrated superior performance in our tests, possibly due to its more robust generalization capabilities.

	Qwen2-7B	Qwen2-72B	GPT3.5	GPT4	GPT4o
In-domain	68.3	73.1	73.1	73.8	73.4
OOD	47.8	50.3	50.5	51.0	50.2

Table 4: nDCG@10 Results Using MiniLM with References from Diverse LLMs.

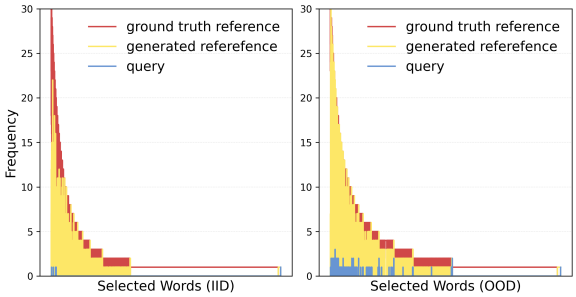

Figure 7: Key Words Overlap Distribution.

5.2 Mechanism behind Query Expansion

We investigate how generated pseudo references improve IR through key vocabulary overlap. We compile "Ground Truth References (GT)" from passages rated relevance level 3 and aggregate "pseudo references (PSE)" for each query from TREC DL. By identifying the top 10 highest Inverse Document Frequency (IDF) words in these references, we compare the frequency of key vocabularies in the query, GT, and PSE. As illustrated in fig. 7, there is a substantial overlap in the frequency of key vocabularies between GT (red region) and PSE (yellow region) in both IID and OOD scenarios, surpassing that between the original query (blue region) and GT (red region). This demonstrates that pseudo references significantly enhance retrieval by incorporating crucial key vocabularies or patterns that target specific passages.

6 Conclusion

This paper explores best practices for query expansion methods in information retrieval with LLMs. We present the Multi-Text Generation Integration (MUGI) framework, a technique that markedly improves information retrieval by integrating the query with multiple generated passages through an adaptive reweighting strategy, feature pooling, and query calibration. Our empirical findings show that MUGI significantly enhances the performance of both sparse and dense retrieval models. We offer a comprehensive discussion of best practices, informed by thorough experimentation.

Method	GPT4 generation	Index search
BM25	-	14 ms
+MuGI	>2500ms	230 ms

Table 5: Latency of vairous methods. Measured by taking average of top 100 results for MS-MARCO dev queries with a single thread. Note that LLM generation latency is highly related to specific models and depends on server load. With the help of accelerating framework such as vLLM, the generation speed could be much faster.

Limitation

The proposed method offers substantial improvements in information retrieval performance. Nonetheless, a shared limitation across query expansion approaches, ours included, is the increased inference time due to the generation of extra passages with LLMs. We provide approxiamte latency using MuGI in table 5.

References

Nasreen Abdul-Jaleel, James Allan, W Bruce Croft, Fernando Diaz, Leah Larkey, Xiaoyan Li, Mark D Smucker, and Courtney Wade. 2004. Umass at trec 2004: Novelty and hard. *Computer Science Department Faculty Publication Series*, page 189.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Data augmentation for information retrieval using large language models. *arXiv preprint arXiv:2202.05144*.

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, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff 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. *ArXiv*, abs/2005.14165.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2021. Overview of the trec 2020 deep learning track.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the trec 2019 deep learning track. *arXiv preprint arXiv:2003.07820*.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*.

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

Wenqi Fan, Zihuai Zhao, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Jiliang Tang, and Qing Li. 2023. Recommender systems in the era of large language models (llms). *ArXiv*, abs/2307.02046.

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

Jiafeng Guo, Yinqiong Cai, Yixing Fan, Fei Sun, Ruqing Zhang, and Xueqi Cheng. 2022. Semantic models for the first-stage retrieval: A comprehensive review. *ACM Transactions on Information Systems*, 40(4):1–42.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338.

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*.

Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*.

Vitor Jeronymo, Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, Roberto Lotufo, Jakub Zavrel, and Rodrigo Nogueira. 2023. Inpars-v2: Large language models as efficient dataset generators for information retrieval. *arXiv preprint arXiv:2301.01820*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering.

Victor Lavrenko and W Bruce Croft. 2017. Relevance-based language models. In *ACM SIGIR Forum*, volume 51, pages 260–267. ACM New York, NY, USA.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks.

Hang Li, Ahmed Mourad, Shengyao Zhuang, Bevan Koopman, and Guido Zuccon. 2022. Pseudo relevance feedback with deep language models and dense retrievers: Successes and pitfalls.

Minghan Li, Honglei Zhuang, Kai Hui, Zhen Qin, Jimmy Lin, Rolf Jagerman, Xuanhui Wang, and Michael Bendersky. 2024. Can query expansion improve generalization of strong cross-encoder rankers?

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

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.

Jerry Liu. 2022. LlamaIndex.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2023a. Fine-tuning llama for multi-stage text retrieval. *arXiv preprint arXiv:2310.08319*.

Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. 2023b. Zero-shot listwise document reranking with a large language model. *arXiv preprint arXiv:2305.02156*.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

Rodrigo Nogueira and Kyunghyun Cho. 2020. Passage re-ranking with bert.

Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. *arXiv preprint arXiv:2003.06713*.

Daniel Park. 2023. Open-llm-leaderboard-report.

Hongjin Qian, Zhicheng Dou, Yutao Zhu, Yueyuan Ma, and Ji rong Wen. 2021. Learning implicit user profile for personalized retrieval-based chatbot. *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2020. Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. In *North American Chapter of the Association for Computational Linguistics*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Stephen E. Robertson and Karen Spärck Jones. 1976. Relevance weighting of search terms. *J. Am. Soc. Inf. Sci.*, 27:129–146.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at trec-3. In *Text Retrieval Conference*.

Devendra Singh Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2022. Improving passage retrieval with zero-shot question generation. *arXiv preprint arXiv:2204.07496*.

Gerard Salton. 1971. The smart retrieval system—experiments in automatic document processing.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge.

Tao Shen, Guodong Long, Xiubo Geng, Chongyang Tao, Tianyi Zhou, and Daxin Jiang. 2023. Large language models are strong zero-shot retriever. *arXiv preprint arXiv:2304.14233*.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. 2023. Is chatgpt good at search? investigating large language models as re-ranking agent. *arXiv preprint arXiv:2304.09542*.

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

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

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2023a. Improving text embeddings with large language models. *arXiv preprint arXiv:2401.00368*.

Liang Wang, Nan Yang, and Furu Wei. 2023b. Query2doc: Query expansion with large language models. *arXiv preprint arXiv:2303.07678*.

Orion Weller, Kyle Lo, David Wadden, Dawn J Lawrie, Benjamin Van Durme, Arman Cohan, and Luca Soldaini. 2023. When do generative query and document expansions fail? a comprehensive study across methods, retrievers, and datasets. *ArXiv*, abs/2309.08541.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*.

Qian Yang, Qian Chen, Wen Wang, Baotian Hu, and Min Zhang. 2023. Enhancing multi-modal multi-hop question answering via structured knowledge and unified retrieval-generation. In *Proceedings of the 31st ACM International Conference on Multimedia*, MM '23, page 5223–5234, New York, NY, USA. Association for Computing Machinery.

Chunyuan Yuan, Wen jie Zhou, Mingming Li, Shangwen Lv, Fuqing Zhu, Jizhong Han, and Songlin Hu. 2019. Multi-hop selector network for multi-turn response selection in retrieval-based chatbots. In *Conference on Empirical Methods in Natural Language Processing*.

Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji rong Wen. 2023a. Recommendation as instruction following: A large language model empowered recommendation approach. *ArXiv*, abs/2305.07001.

Le Zhang, Yihong Wu, Fengran Mo, Jian-Yun Nie, and Aishwarya Agrawal. 2023b. MoqaGPT : Zero-shot multi-modal open-domain question answering with large language model. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1195–1210, Singapore. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Xiang Gao, Yuwei Fang, Chris Brockett, Michel Galley, Jianfeng Gao, and Bill Dolan. 2022. Retgen: A joint framework for retrieval and grounded text generation modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11739–11747.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Zhicheng Dou, and Ji-Rong Wen. 2023. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107*.