A Multi-Task Embedder For Retrieval Augmented LLMs

Peitian Zhang[2]*, **Shitao Xiao**[1]*, **Zheng Liu**[1]*†, **Zhicheng Dou**[2], **Jian-Yun Nie**[1,3],
[1]Beijing Academy of Artificial Intelligence, [2]Renmin University of China, [3]University of Montreal
{namespace.pt, zhengliu1026}@gmail.com

Abstract

LLMs confront inherent limitations in terms of its knowledge, memory, and action. The retrieval augmentation stands as a vital mechanism to address these limitations, which brings in useful information from external sources to augment the LLM. However, existing retrieval methods encounter two pressing issues. On one hand, the general retrievers are not properly optimized for retrieval augmentation hence exhibit limited effectiveness; on the other hand, the task-specific retrievers excel in the targeted retrieval augmentation scenario, while lack the versatility to handle diverse scenarios. In this work, we propose **LLM-Embedder** for the unified support of diverse retrieval augmentation scenarios. Our method presents three technical contributions. Firstly, we introduce a new *reward formulation*, namely rank-aware reward. It exploits the ranking position of the desired output among N sampled outputs from the LLM, which leads to fine-grained and robust computation of reward from the LLM's feedback. Secondly, we design a novel *distillation objective*, called graded distillation. It incorporates both the absolute value and the relative order of the reward for more sufficient utilization of the LLM's feedback. Thirdly, we systematically optimize the *multi-task learning*, which effectively unifies the multiple retrieval functionalities into one model. In our experiment, LLM-Embedder notably improves the LLM's performances in various downstream tasks, and outperforms both general and task-specific retrievers with a substantial advantage. Our data, code, and model have been released at https://github.com/FlagOpen/FlagEmbedding.

1 Introduction

Large language models (LLMs) present a unified foundation to support general artificial intelligence applications (Brown et al., 2020a; Chowdhery et al., 2022; Touvron et al., 2023). Despite the substantial improvement over the last-gen methods, LLMs still face many severe problems, such as hallucination (Ji et al., 2023; Bang et al., 2023), limited memory (Bai et al., 2023b; An et al., 2023), misfollowing of instructions (Ouyang et al., 2022; Bai et al., 2022). Many of the challenges can be traced back to the inherent limitations of LLMs in terms of *knowledge*, *memory*, and *action*. Specifically, LLMs cannot internalize the vast and constantly changed world knowledge due to their finite and static parameters. LLMs are incapable of memorizing and utilizing long-term information because of the limited context length. Finally, LLMs require manually in-context examples and tools to accomplish complex real-world tasks.

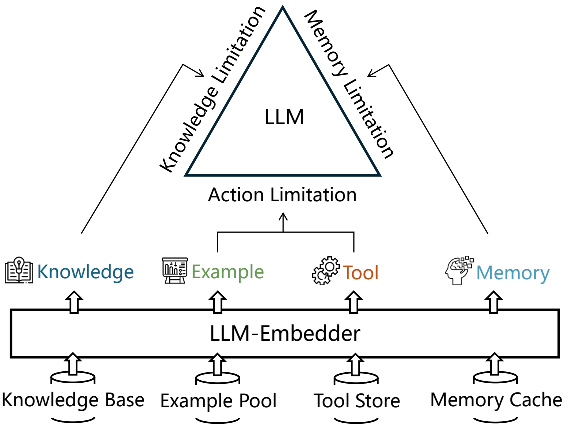

Figure 1: LLM-Embedder presents a unified embedding model for the diverse retrieval augmentation scenarios.

Retrieval augmentation stands as a vital mechanism to address these inherent limitations of the LLM. It brings in useful information from external sources, such as knowledge, memory pieces, in-context examples, and tools, which substantially enhances the LLM for the generation of desired outputs (Gao et al., 2023). The embedding model (a.k.a. *embedder*) is a critical part of retrieval augmentation, which bridges the LLM's information needs with external sources. The existing embed-

[*] Co-first authors. This work was done when Peitian Zhang is in an internship at BAAI.

[†] Corresponding author.

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 3537–3553
August 11-16, 2024 ©2024 Association for Computational Linguistics

ding models can be briefly partitioned into two categories. One is the general-purpose embedders, which aim to be universally applicable for various retrieval tasks (Izacard et al., 2021; Wang et al., 2022b; Xiao and Liu, 2023). Despite their popularity, they are not properly optimized for retrieval augmentation, and are thus prone to an inferior effectiveness in the corresponding task. The other one is the task-specific embedders, which are tailored for one specific retrieval augmentation scenario, e.g., knowledge retrieval (Yu et al., 2023) and example retrieval (Wang et al., 2023a). However, these methods lack versatility across different scenarios. As the LLMs require assistance from diverse external sources in solving real-world problems, it becomes imperative to develop an effective and versatile embedding model to support the diverse retrieval augmentation needs.

In this paper, we present LLM-Embedder, a unified embedding model to support a broad range of retrieval augmentation scenarios, including knowledge retrieval, memory retrieval, example retrieval, and tool retrieval. Training such a versatile embedding model presents multiple challenges in terms of 1) how to learn from the LLM, and 2) how to harmonize different retrieval tasks. In LLM-Embedder, the following technical contributions are presented.

• **Reward Formulation.** For each retrieval augmentation scenario, the embedder is learned from the LLM's feedback, i.e. the retrieval candidate needs to be promoted if it contributes to the generation of the desired output. Conventional methods rely on the generation likelihood (Shi et al., 2023; Izacard et al., 2023). However, the absolute generation likelihood tends to fluctuate dramatically, which may lead to inaccurate estimation of the contribution of each retrieval candidate. In LLM-Embedder, we propose a new reward formulation called *rank-aware reward*. Essentially, a retrieval candidate will receive a higher reward if it can better promote the desired output's ranking among N sampled outputs from the LLM. Thus, it is free from dealing with the absolute generation likelihood, which facilitates a fine-grained and more robust computation of the reward.

• **Distillation Objective**. Based on the LLM's reward, the embedding model is learned by knowledge distillation. Typically, this is accomplished by minimizing the KL-divergence between the reward distributions and the relevance distribution estimated by the embedder (Shi et al., 2023; Yu et al., 2023). In many cases, the reward distribution are either polarized (extremely high rewards for one candidate while low rewards for others) or flat (even rewards for every candidate), which makes it difficult to distill fine-grained knowledge with KL-Divergence. To address this problem, we design the *graded distillation*. It integrates both the absolute values of rewards and their relative orders for knowledge distillation, which leads to a more sufficient exploitation of the LLM's feedback.

• **Multi-task Learning.** LLM-Embedder is trained to support diverse retrieval augmentation scenarios through multi-task learning. However, different scenarios need to capture distinct semantic relationships, hence the multiple training tasks may conflict with each other. To harmonize the learning process, we perform systematic optimization with three techniques: 1) *self-paced learning scheduling*, where lossy tasks can be automatically compensated by higher learning rates; 2) *homogeneous batching*, where training samples from one common task are gathered in the same batch to optimize the impact of in-batch negative sampling; 3) *diversified prompting*, which presents different tasks with unique prefixes such that the embedding model can better distinguish each of them.

To summarize, LLM-Embedder stands as a pioneering work for the uniform support of the diverse retrieval augmentation scenarios of LLMs. It makes threefold technical contributions, and brings valuable inspirations on how to learn from LLM's feedback and how to harmonize different retrieval tasks. In our experiment, LLM-Embedder achieves a superior performance, where it notably improves the LLM's performance in a variety of downstream tasks. Meanwhile, its retrieval augmentation's effect is superior to both general and task-specific retrieval methods. Our model and code will be publicly available to facilitate future research.

2 Related Works

• **Embedding Model** maps the input text into dense vector (i.e. *embedding*) in the semantic space, where the relevance between texts is measured by the similarity between embeddings. It has become the de-facto choice for modern information retrieval systems. There are mainly three research threads for improving the performance of embedding models. The first one is leveraging advanced backbone models, including the retrieval oriented models (Liu and Shao, 2022; Wang et al., 2022a) and large language models (Ma et al., 2023; Li et al., 2023). Another thread is enhancing the learning methodology, such as upgrading the negative sampling strategy (Karpukhin et al., 2020; Izacard et al., 2021; Xiong et al., 2020) and incorporating knowledge distillation from a more precise ranking model (Qu et al., 2020; Hofstätter et al., 2021; Xiao et al., 2022). Last but not least, many recent works dedicate to train a universal retriever across a wide array of tasks (Wang et al., 2021; Lewis et al., 2021; Karouzos et al., 2021; Yu et al., 2022; Su et al., 2022; Asai et al., 2022). LLM-Embedder inherits successful practices for training high-quality dense retriever, while innovating novel techniques to tailor for the multi-task learning of diverse retrieval augmentation scenarios.

- **Retrieval Augmentation** is a vital mechanism to address the inherent limitations of the LLM in terms of knowledge, memory, and action. Concretely, the LLM can 1) generate factoid answers with retrieved knowledge (Gao et al., 2024; Jiang et al., 2023); 2) utilize long-context information with retrieved memory pieces (Rubin and Berant, 2023; Wang et al., 2023b; Xu et al., 2023); 3) better follow human instruction with retrieved in-context examples (Brown et al., 2020b; Cheng et al., 2023); 4) execute complex tasks with retrieved tools (Qin et al., 2023). In practice, there are two common options of retrievers: the general retrievers (Robertson et al., 2009; Izacard et al., 2021; Xiao and Liu, 2023; Neelakantan et al., 2022) and the task-specific retrievers (Yu et al., 2023; Wang et al., 2023b; Qin et al., 2023). The general retrievers exhibit superior versatility, but may suffer from an inferior retrieval quality in retrieval augmentation tasks. In contrast, task-specific retrievers are more specialized, achieving better performance in the targeted scenario, while falling short when handling other scenarios. Compared with the existing works, LLM-Embedder unifies the generality and specialty: it comprehensively supports all major retrieval augmentation needs of the LLM, meanwhile achieving the leading performance in every retrieval augmentation scenario.

3 LLM-Embedder

In this section, we will present the retrieval augmentation scenarios with LLM-Embedder (§3.1), and introduce its training methodology (§3.2).

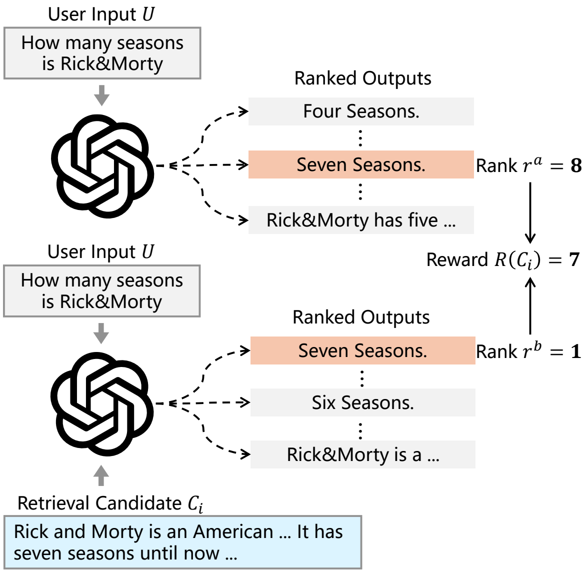

Figure 2: The rank-aware reward for each retrieval candidate. It measures the improvement of the rank of the desired output among multiple sampled outputs.

3.1 Retrieval Augmentation

LLM-Embedder targets on the unified support for the major retrieval augmentation needs of the LLMs, including knowledge retrieval, memory retrieval, example retrieval, and tool retrieval. It transforms each retrieval candidate $C_i \in \mathcal{C}$ into its embedding $\boldsymbol{C}_i \in \mathbb{R}^D$ and stores all embeddings in a vector DB. It also embeds the user input U into $\boldsymbol{U} \in \mathbb{R}^D$, then retrieves the top-K relevant candidates based on cosine similarity:

$$\mathrm{Ret}(U) \leftarrow \underset{C_i}{\text{top-}K}\{\cos(\boldsymbol{U}, \boldsymbol{C}_i)\}. \tag{1}$$

The retrieval result and the user input are synthesized with template ψ to prompt the LLM Θ:

$$O \leftarrow \Theta(\psi(U, \mathrm{Ret}(\boldsymbol{U}))). \tag{2}$$

Each retrieval augmentation scenario has its unique formulation of retrieval candidate, user input, and prompt template, which are elaborated as follows.

- **Knowledge Retrieval**. The LLM can generate factoid answers with retrieved knowledge. Each retrieval candidate is a passage from an external knowledge corpus. The user input is usually an explicit question. It can also be a conversation context with a context-dependent question. In this case, we concatenate the entire context as the user input. The retrieved passages and the user input are synthesized according to Template A.1.
- **Memory Retrieval**. The LLM can remember and utilize long context memory with memory retrieval (Xu et al., 2023). Specifically, the long

context split into equal-size chunks $\{v_1, \dots, v_n\}$. When processing the v_j, each previous chunk concatenated with its subsequent chunk is treated as a retrieval candidate, i.e. $C_i \leftarrow v_i + v_{i+1},\ \ i < j$. The user input is v_i itself. Denote the LLM's context window size as L^*. We maintain the recent L tokens in the context window, while the rest $L^* - L$ are populated with retrieved chunks.

- **Example Retrieval**. In-context examples help the LLM to better follow human instruction. Instead of relying on manual specification, in-context examples can be retrieved automatically to improve the performance. Each example contains an optional task description, an input, and an output, which are all concatenated to form a retrieval candidate. The user input is the concatenation of the task description and the task-specific input. The retrieved examples and the user input are synthesized with Template A.5 to feed into the LLM.
- **Tool Retrieval**. The LLM leverages tools to execute complex real-world tasks (Qin et al., 2023; Yao et al., 2023). Tool retrieval efficiently provides useful tools for the LLM. The tool's description and its API are concatenated as the retrieval candidate. The user's request is treated as the user input.

3.2 Training Methodology

3.2.1 Reward Formulation

A retrieval candidate is useful if it can facilitate the generation of the desired output (denoted as O^*). The absolute value of generation likelihood is not an appropriate measurement because it is prone to dramatic numerical fluctuations. Alternatively, as shown in Figure 2, we argue that a retrieval candidate is useful if it can lead to a better ranking position of the desired output among N sampled outputs from the LLM's ($\{O_i\}_{i=1}^{N}$). Based on this argument, we descendingly sort the sampled outputs based their generation likelihoods and compute the rank of the desired output among them when retrieval is disabled:

$$r^a \leftarrow \underset{O^*}{\mathrm{rank}}(\{O_1, \dots, O_N :\ O_i \sim \Theta(U)\}).$$

We then compute the rank of the desired output with the same operation except that the retrieval augmentation is enabled:

$$r^b \leftarrow \underset{O^*}{\mathrm{rank}}(\{O_1, \dots, O_N :\ O_i \sim \Theta(\psi(U, C_i))\})$$

Finally, the reward for the retrieval candidate C_i is computed as its improvement of the rank:

$$R(C_i) \leftarrow r^a - r^b. \tag{3}$$

This reward formulation is free of dealing with absolute likelihood values, but focuses on the retrieval candidate's real impact on facilitating the generation of the desired output.

3.2.2 Distillation Objective

Based on the LLM's rewards, the embedding model is learned through knowledge distillation, so that the relevance estimated by the embedder becomes consistent with the retrieval candidate's actual usefulness. Minimizing KL-Divergence between the relevance distribution and the reward distribution is the most typical approach (Shi et al., 2023; Izacard et al., 2023; Yu et al., 2023). However, the reward distribution sometimes exhibits polarized (substantially high reward for one candidate while low for others) or flat (even reward for each candidate) patterns. The KL-Divergence cannot effectively distill fine-grained knowledge from these distributions. To address this problem, we innovate a *graded distillation* objective, which integrates both the absolute reward values and the relative reward orders for learning. It consists of a series of contrastive losses, where the negatives of each loss include the lower-rewarded candidates and the in-batch candidates. All contrastive losses are aggregated with normalized rewards as weights. Formally, given the retrieval candidates $\{C_i\}_{i=1}^{M}$, their normalized rewards $w(C_i) \leftarrow \mathrm{softmax}(R(C_*))[i]$, the objective is formulated as:

$$\mathcal{N}(C_i) \leftarrow \{C :\ R(C) < R(C_i)\} \cup \mathrm{InBatch}(C_i),$$
$$\min \sum_{C_i} -w(C_i) \log \frac{e^{\cos(\boldsymbol{U}, \boldsymbol{C_i})}}{\sum_{C' \in \mathcal{N}(C_i)} e^{\cos(\boldsymbol{U}, \boldsymbol{C'})}}. \tag{4}$$

The graded distillation objective enjoys two advantages. On one hand, it can robustly optimize the embedder from various reward distributions. For the polarized rewards, it will become the one-hot contrastive learning. For the flat rewards, it will always supervise the embedder to prioritize the more useful candidates against the less useful ones, regardless of the absolute value of the reward. On the other hand, it incorporates in-batch negatives in the training process, which further improves the discrimination capability of the embedder.

3.2.3 Multi-Task Learning

LLM-Embedder learns to support the four retrieval augmentation needs with a single model through multi-task learning. Different retrieval tasks call for distinct semantic relationships, which may conflict with each other. Therefore, it's important to distinguish these tasks and harmonize the their learning process. In this place, we tailor the multi-task learning framework with three techniques.

- **Self-Paced Learning Scheduling.** The intrinsic learning difficulty of each task may vary, potentially leading to differences in the model's learning pace for each task. This may result in the over-optimization of simpler tasks and the under-optimization of more challenging tasks. Inspired by (Liu et al., 2019), we propose to dynamically adjust the learning pace of each retrieval task to address this problem. Specifically, we deem the loss of each retrieval task as a proxy to the learning condition of that task. Based on it, we amplify the learning rate for lossy tasks and reduce the learning rate for already learned tasks. To achieve this goal, we periodically checkpoint the loss of retrieval task T during training, denoted as L_0^T. Given the basic learning rate α, and the current loss of the retrieval task T, the learning rate of the current optimization step is set to $\alpha \times \sqrt{\frac{L^T}{L_0^T}}$.
- **Homogeneous Batching.** The embedding model's discrimination capability benefits from the quality and quantity of negative samples (Izacard et al., 2021; Wang et al., 2022b), which consist of hard negatives and in-batch negatives. The vanilla batching strategy often packs training samples from different tasks in the same batch. These samples are irrelevant to each other and hence adversely influence the quality of in-batch negatives. Instead, we gather the training samples from the same retrieval task to form every batch. In this way, LLM-Embedder should discriminate the positive sample against $B \times M \times Z - 1$ negatives from the same retrieval task, where B is the batch size, M the candidate number, and Z the GPU number.
- **Diversified Prompting.** For retrieval task T, two unique instructions I_U^T, I_C^T are assigned, which are prefixed to the user input and the retrieval candidate, respectively. The concatenated sequence is encoded into its embedding by LLM-Embedder:

$$\boldsymbol{U}^T \leftarrow \text{encode}(I_U^T{+}U),\ \boldsymbol{C}_i^T \leftarrow \text{encode}(I_C^T{+}C_i).$$

The resulting embedding $\boldsymbol{U}^T$ and $\boldsymbol{C}_i^T$ are differentiated across tasks, which helps LLM-Embedder to distinguish each task.

4 Experiment

The experimental studies aim to investigate three research questions. *RQ 1.* Can LLM-Embedder support the LLM's diverse retrieval augmentation need? (§4.2) *RQ 2.* What is LLM-Embedder's impact on each retrieval augmentation scenario? (§4.3) *RQ 3.* What is the individual contribution of each technique in LLM-Embedder? (§4.4)

4.1 Settings

4.1.1 Training & Evaluation

We introduce the details of training and evaluation on the four retrieval augmentation scenarios. Statistics of all training datasets are reported in Table 7.

- **Knowledge Retrieval.** We train LLM-Embedder with three datasets for knowledge retrieval, including MSMARCO (Nguyen et al., 2016), Natural Questions (Kwiatkowski et al., 2019), and QReCC (Anantha et al., 2020). Note that QReCC does not have well-formed answers for generating rewards, thus, we use the annotated relevance for contrastive learning. We include three datasets to evaluate the impact of knowledge retrieval. 1) MMLU (Hendrycks et al., 2020), a multiple-choice questions dataset that covers a wide range of knowledge. We retrieve 3 passages from the MSMARCO Passage corpus (Nguyen et al., 2016), which are integrated as a prompt with the official Template A.2. The metric is accuracy. 2) PopQA (Mallen et al., 2022), a question answering dataset that focuses on long-tail entities. We retrieve 3 passages from Wikipedia (Karpukhin et al., 2020), which are integrated with the official Template A.3. The metric is exact match. 3) QReCC (Anantha et al., 2020), a conversational search dataset that requires the retriever to find the relevant passage according to a conversation context. It already provides the ground-truth passage, we directly evaluate the ranking metric, i.e. NDCG@3 following previous works (Mao et al., 2023).
- **Memory Retrieval.** We consider two tasks for memory retrieval. 1) Long-context conversation with MSC (Xu et al., 2021), where the LLM should generate the ground-truth response. We retrieve 1 historical dialogue turn as additional context, which is synthesized with Template A.4. We use its training set to fine-tune LLM-Embedder. 2) Long-range language modeling with Books3 (Gao et al., 2020), ArXiv (Gao et al., 2020), CodeParrot (Tunstall et al., 2022), and PG19 (Rae et al., 2019), where PG19 is held-out from training. We set the chunk size to 128, and maintain a recent context length of 2048. We retrieve 8 chunks and their continuation chunk to prepend to the recent

context. Perplexity is the metric for both tasks.

- **Example Retrieval.** We follow LLM-R (Wang et al., 2023a) to use in-context learning tasks from FLAN (Chung et al., 2022) for training and evaluating the impact of example retrieval. It consists of 9 distinct categories with 30 datasets: Closed-Book QA (CQA), Commonsense (Comm), Coreference (Coref), Paraphrase (Para), Natural Language Inference (NLI), Reading Comprehension (RC), Sentiment Analysis (Sent), Data2Text (D2T), Summarization (Summ). We retrieve 8 examples from the union of the training set examples, which are synthesized with Template A.5. The evaluation metric is specified in Table 6.
- **Tool Retrieval.** We use the ToolBench (Qin et al., 2023) for training and evaluating the tool retrieval performance. Akin to QReCC, this dataset does not include desired output from the LLM, hence we train LLM-Embedder with contrastive loss and directly evaluate NDCG@5.

4.1.2 Baselines

Firstly, we measure the performance of the LLM without retrieval augmentation, denoted as *None*. Secondly, we compare with two types of retrievers. 1) *General retrievers*, which aim to support a wide range of text retrieval and representation tasks, such as question answering, entity retrieval, and duplication detection. We include the following widely-recognized baselines: BM25 (Robertson et al., 2009), Contriever (Izacard et al., 2021), Instructor (Su et al., 2022), RetroMAE-BEIR (Liu and Shao, 2022), and BGE (Xiao and Liu, 2023). These methods are empirically competitive according to BEIR (Thakur et al., 2021) and MTEB (Muennighoff et al., 2022) benchmarks. 2) *Task-specific embedding models*. These models are optimized for one specific retrieval augmentation scenario. We include the following baselines that excel in their respective scenario: ARR (Yu et al., 2023) for knowledge retrieval, LLM-R (Wang et al., 2023a) for example retrieval, and API-Retriever (Qin et al., 2023) for tool retrieval. Since retrieval augmentation introduces additional context to the LLM, we add a simple yet strong baseline called *Recency* for memory retrieval. It directly extends the context window by the length of retrieved context.

4.1.3 Implementation

We use Llama-2-7B-Chat (Touvron et al., 2023) as the backbone LLM. Besides, we utilize BGE base (Xiao and Liu, 2023) to initialize LLM-Embedder and fine-tune it as described in §3.2. The hyper parameters during fine-tuning are shown in Table 9. Although using rewards from Llama-2 7B Chat, LLM-Embedder is also applicable to other LLMs and its advantage remains. The experimental results are shown in Appedix D. We use Flat index from faiss (Johnson et al., 2019) for searching.

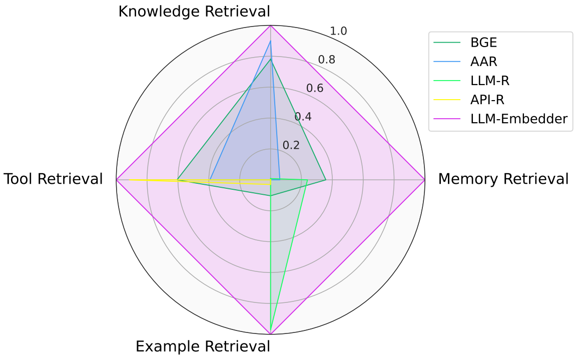

Figure 3: Impact of retrieval augmentation from different retrievers (metric values are min-max normalized).

4.2 Overall Analysis

The evaluation results of the four retrieval augmentation scenarios are presented in Table 1-3.

Firstly, compared with the results without retrieval augmentation, i.e. None, LLM-Embedder delivers more precise answers with the retrieved knowledge (Table 1), improved quality of long-sequence generation with the retrieved memory (Table 2), better instruction following effect with the retrieved examples (Table 3), and more accurate tool retrieval (Table 3). Besides, though the LLM's performance can also by improved by other baseline retrievers, LLM-Embedder always leads to the most amplified retrieval augmentation effect across all scenarios. It outperforms all general retrievers and is competitive against task-specific retrievers, i.e. AAR for knowledge enhancement, LLM-R for example retrieval, and API-Retriever for tool retrieval. This observation validates that *the LLM benefits from the retrieved information; meanwhile, LLM-Embedder can provide a strong and unified foundation to support diverse retrieval augmentation needs of the LLM*.

We can also observe that the task-specific embedders optimized for one scenario result in limited performances in others, suggesting that the semantic relationships required by different retrieval scenarios are not transferable. To better illustrate this point, we visualize the retrieval augmentation's impact from five representative methods in Figure 3. Notably, although task-specific embed-

	MMLU					PopQA	QReCC
Method	STEM	Social	Human	Other	All Avg.	PopQA	QReCC
None	0.347	0.533	0.509	0.497	0.460	0.206	–
BM25	0.376	0.538	0.505	0.509	0.472	0.349	0.434
Instructor	0.370	0.541	0.511	0.508	0.472	0.353	0.286
Contriever	0.368	0.538	0.508	0.501	0.468	0.328	0.356
RetroMAE-BEIR	**0.386**	0.546	0.522	0.528	0.485	0.436	0.404
BGE*	0.385	0.556	0.519	**0.539**	**0.490**	0.449	0.386
AAR†	0.380	0.550	0.513	0.529	0.483	0.479	0.288
API-Retriever	0.354	0.534	0.500	0.507	0.463	0.249	0.114
LLM-R	0.363	0.528	0.502	0.498	0.463	0.251	0.023
LLM-Embedder (Ours)	0.385	**0.557**	**0.523**	0.536	**0.490**	**0.505**	**0.505**

Table 1: The impact of knowledge retrieval. "$*$" and "$\dagger$" indicate the SOTA general embedder and the task-specific embedder, respectively. The best metrics are in bold, and the second-best metrics are underlined.

	Conversation	Language Modeling			
Method	MSC	Books3	Arxiv	CodeParrot	PG19 (o.d.)
None	19.350	8.819	3.765	2.766	10.251
Recency	13.957	8.739	3.416	2.599	10.222
BM25	14.651	8.658	3.311	2.459	10.196
Instructor	14.880	8.662	3.355	2.476	10.201
Contriever	14.213	8.646	3.271	2.444	10.162
RetroMAE-BEIR	14.399	8.638	3.290	2.459	10.173
BGE*	14.294	8.631	3.291	2.458	10.154
AAR	14.700	8.638	3.326	2.467	10.181
API-Retriever	14.783	8.672	3.386	2.492	10.183
LLM-R	14.475	8.662	3.364	2.472	10.202
LLM-Embedder (Ours)	**13.483**	**8.608**	**3.232**	**2.430**	**10.118**

Table 2: The impact of memory retrieval. Recency is to directly extend the context without retrieval.

ders exhibit competitive performance for their targeted scenario, their impacts are severely weakened when applied on other scenarios. In contrast, LLM-Embedder demonstrates a steady and competitive performance across all scenarios. To summarize, *the irrelevant or even adverse retrieval patterns can be reconciled by one unified embedding model on top of our optimized training methodology*.

4.3 Individualized Analysis

• **Knowledge Retrieval**. The evaluation results of knowledge retrieval are reported in Table 1. We make the following observations. 1) Benefit of external knowledge. On both MMLU and PopQA, we can observe significant empirical advantages of the retrieval augmentation methods compared with the plain LLM, i.e. None. Among all retrieval methods, LLM-Embedder is able to return the most accuracy knowledge, leading to the best retrieval augmentation effect on both datasets. 2) Distinction among datasets. The impact of knowledge retrieval is more noticeable on PopQA than MMLU. This is because PopQA is more knowledge-intensive, with a focus on questions about long-tail entities. Moreover, the baseline embedding models fail to handle conversational search queries, resulting in their inferior NDCG compared with BM25 on QReCC. In contrast, LLM-Embedder significantly outperfoms all baselines on QReCC, again verifying its versatility.

• **Memory Retrieval**. The evaluation results of memory retrieval are reported in Table 2. On one hand, baseline retrievers underperform the Recency baseline on MSC, which translates to the negative impact of the retrieved conversation compared with the recent one. This observation underscores the challenges in effective memory retrieval. On the other hand, the LLM-Embedder retains its superior performance, reducing the perplexity against the all baseline methods on all datasets.

• **Example Retrieval**. The evaluation results of example retrieval are reported in Table 3. We have the following observations. 1) Compared with random examples, using retrieved examples yields improved performances in most cases. This finding underscores the effect of example retrieval for helping the LLM to properly follow instructions. 2) BM25's performance is substantially weaker than its performance in other scenarios. This discrepancy can be attributed to the specific nature of in-context learning, where useful examples may

	In-Context Learning										Tool
Method	CQA	Comm	Coref	Para	NLI	RC	Sent	D2T	Summ	Avg	ToolBench
None	0.292	0.721	**0.658**	0.524	0.448	0.489	0.708	0.198	0.145	0.465	–
Random	0.359	0.719	0.589	0.520	0.477	0.553	0.916	0.350	0.357	0.545	0
BM25	0.360	0.702	0.603	0.506	0.458	0.540	0.728	0.302	0.156	0.484	0.512
Instructor	0.500	0.777	0.574	0.631	0.536	0.622	0.915	0.460	0.457	0.604	0.388
Contriever	0.491	0.772	0.562	0.636	0.547	0.630	0.914	0.438	0.444	0.601	0.490
RetroMAE-BEIR	0.459	0.774	0.584	0.576	0.541	0.603	**0.929**	0.466	0.447	0.594	0.521
BGE*	0.472	0.777	0.555	0.617	0.541	0.599	0.928	0.472	0.452	0.597	0.576
AAR	0.481	0.780	0.585	0.589	0.535	0.604	0.921	0.445	0.441	0.594	0.420
API-Retriever†	0.477	0.762	0.547	0.627	0.520	0.610	0.924	**0.487**	0.442	0.595	0.802
LLM-R†	**0.517**	0.780	0.583	**0.657**	**0.615**	0.622	0.906	0.478	**0.488**	0.626	0.132
LLM-Embedder	0.516	**0.784**	0.593	0.656	0.604	**0.632**	0.922	0.473	0.474	**0.627**	**0.865**

Table 3: The impact of example retrieval and tool retrieval.

Method	**Knwl.**	**Mem.**	**Expl.**	**Tool**
LLM-Embedder	**0.505**	**13.483**	**0.627**	**0.865**
w.o. Rank-Aware Reward	0.485	14.253	0.622	0.861
w.o. Graded Distillation	0.492	13.547	0.610	0.854
w.o. Self-Paced Scheduling	0.492	13.883	0.619	0.809
w.o. Homogeneous Batching	0.447	14.183	0.605	0.836
w.o. Diversified Instruction	0.503	13.942	0.619	0.828

Table 4: Ablation studies of LLM-Embedder.

have low lexical similarity with the user input.

- **Tool Retrieval**. The evaluation results of example retrieval are reported in Table 3. We observe that the task-specific method, i.e. the API retriever, beats other baseline methods by a large margin. This is because these baselines are unfamiliar with tools and hence fail to properly estimate the relevance. However, LLM-Embedder continues to maintain the leading position, highlighting its unfied support for diverse retrieval tasks.

4.4 Ablation Studies

The ablation studies are performed to to evaluate the impact from each technical factor. The evaluation results are reported in Table 4.

For "*w.o. Rank-Aware Reward*", we switch to the typical likelihood-based reward formulation (Shi et al., 2023). Notably, the performance on knowledge retrieval and memory retrieval substantially decreases. We conjecture that in both scenarios, the generation likelihood of the desired output drastically fluctuate, resulting in the inaccurate measurement of the retrieval candidate's usefulness.

For "*w.o. Graded Distillation*", the graded distillation objective is replaced by the typical KL-divergence (Izacard et al., 2023). As introduced, graded distillation can stay robust to the polarized or flat rewards, which leads to more effective usage of the LLM's feedback. In this place, we can observed that LLM-Embedder's performance is reduced when graded distillation is disabled, especially for example retrieval.

For "*w.o. Self-Paced Scheduling*", the learning rate is the kept static for all retrieval tasks during fine-tuning. We can observe that the performance of tool retrieval drops significantly. This is because the learning for this scenario does not proceed at the same pace as other scenarios, necessitating the dynamic control over learning speed for different retrieval tasks.

For "*w.o. Homogeneous Negatives*", the homogeneous in-batch negatives are disabled. This change reduces the discrimination capability of the embedder, because a great portion of the in-batch negative samples will come from different tasks, which are irrelevant to the target one. As we can observe, LLM-Embedder's performance is decreased due to such a change, especially for knowledge retrieval, where LLM-Embedder should discriminate the relevant passage from a massive corpus.

For "*w.o. Diversified Instruction*", we remove the task-specific instructions in fine-tuning and evaluation. Without this technique, it becomes harder for the embedding model to distinguish different retrieval tasks. This intuition is consistent with the observed result, as LLM-Embedder's performance decreases across all tasks.

5 Conclusion

In this work, we present LLM-Embedder, a unified embedding model to support the LLM's diverse retrieval augmentation needs, including knowledge retrieval, memory retrieval, example retrieval, and tool retrieval. We propose three key techniques to facilitate the training of LLM-Embedder, spanning from reward formulation, distillation objective, and multi-task learning recipe. Our experiments show LLM-Embedder's empirical advantages over both general and task-specific embedding models across all evaluation scenarios. This highlights its effectiveness as a foundational building block to support the retrieval augmentation of the LLM.

6 Limitations

A few recent studies incorporate large language models as the embedding backbone and achieve new state-of-the-art performance. However, LLM-Embedder is a BERT-base scale model. Its scaling effect remains unexplored. Besides, LLM-Embedder is specifically tailored for the four retrieval scenarios. For tasks that fall outside its scope of coverage, such as documentation retrieval, the effectiveness of the LLM-Embedder may not be as robust as that of a strong general embedding model like BGE.

7 Ethical Considerations

LLM-Embedder is an embedding model that maps the text into high-dimensional vectors and relies on vector similarity to determine relevance between texts. Therefore, it inherits the potential risks of the embedding model family. Specifically, LLM-Embedder may process a large amount of personal or sensitive data, which must be handled with consent. There is also the security concern as recent works have proven it possible to decrypt the original textual information from embedded vectors. Lastly, it may perpetuate and amplify biases present in the training data, leading to unfair or discriminatory outcomes.

Acknowledgement

This research is supported by National Science and Technology Major Project(2023ZD0121504).

References

2023. AquilaChat-7B. https://huggingface.co/BAAI/AquilaChat-7B/.

Chenxin An, Shansan Gong, Ming Zhong, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. 2023. L-eval: Instituting standardized evaluation for long context language models.

Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. 2020. Open-domain question answering goes conversational via question rewriting.

Akari Asai, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen-tau Yih. 2022. Task-aware retrieval with instructions.

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. 2023a. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional ai: Harmlessness from ai feedback.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. 2023b. Longbench: A bilingual, multitask benchmark for long context understanding.

Baichuan. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity.

Luisa Bentivogli, Bernardo Magnini, Ido Dagan, Hoa Trang Dang, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In *Proceedings of the Second Text Analysis Conference, TAC 2009, Gaithersburg, Maryland, USA, November 16-17, 2009*. NIST.

Sumithra Bhakthavatsalam, Daniel Khashabi, Tushar Khot, Bhavana Dalvi Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, and Peter Clark. 2021. Think you have solved direct-answer question answering? try arc-da, the direct-answer AI2 reasoning challenge. *CoRR*, abs/2102.03315.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21,* *2015*, pages 632–642. The Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020a. Language models are few-shot learners.

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. 2020b. 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*.

Daixuan Cheng, Shaohan Huang, Junyu Bi, Yuefeng Zhan, Jianfeng Liu, Yujing Wang, Hao Sun, Furu Wei, Denvy Deng, and Qi Zhang. 2023. Uprise: Universal prompt retrieval for improving zero-shot evaluation.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Y. Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *CoRR*, abs/2210.11416.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. 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 2924–2936. Association for Computational Linguistics.

DataCanary, hilfialkaff, Lili Jiang, Meg Risdal, Nikhil Dandekar, and tomtung. 2017. Quora question pairs.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*. Asian Federation of Natural Language Processing.

Ondrej Dusek, David M. Howcroft, and Verena Rieser. 2019. Semantic noise matters for neural natural language generation. In *Proceedings of the 12th International Conference on Natural Language Generation, INLG 2019, Tokyo, Japan, October 29 - November 1, 2019*, pages 421–426. Association for Computational Linguistics.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding.

Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. Efficiently teaching an effective dense retriever with balanced topic aware sampling.

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

Gautier Izacard, Patrick S. H. Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *J. Mach. Learn. Res.*, 24:251:1–251:43.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation.

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation.

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

Constantinos Karouzos, Georgios Paraskevopoulos, and Alexandros Potamianos. 2021. Udalm: Unsupervised domain adaptation through language modeling.

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.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 252–262. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research.

Hector J. Levesque. 2011. The winograd schema challenge. In *Logical Formalizations of Commonsense Reasoning, Papers from the 2011 AAAI Spring Symposium, Technical Report SS-11-06, Stanford, California, USA, March 21-23, 2011*. AAAI.

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. Paq: 65 million probably-asked questions and what you can do with them.

Chaofan Li, Zheng Liu, Shitao Xiao, and Yingxia Shao. 2023. Making large language models A better foundation for dense retrieval. *CoRR*, abs/2312.15503.

Bill Yuchen Lin, Ming Shen, Wangchunshu Zhou, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. Commongen: A constrained text generation challenge for generative commonsense reasoning. In *Conference on Automated Knowledge Base Construction, AKBC 2020, Virtual, June 22-24, 2020*.

Shengchao Liu, Yingyu Liang, and Anthony Gitter. 2019. Loss-balanced task weighting to reduce negative transfer in multi-task learning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 9977–9978. AAAI Press.

Zheng Liu and Yingxia Shao. 2022. Retromae: Pre-training retrieval-oriented transformers via masked auto-encoder.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2023. Fine-tuning llama for multi-stage text retrieval. *CoRR*, abs/2310.08319.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Hannaneh Hajishirzi, and Daniel Khashabi. 2022. When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories.

Kelong Mao, Hongjin Qian, Fengran Mo, Zhicheng Dou, Bang Liu, Xiaohua Cheng, and Zhao Cao. 2023. Learning denoised and interpretable session representation for conversational search. In *Proceedings of the ACM Web Conference 2023*, WWW '23, page 3193–3202, New York, NY, USA. Association for Computing Machinery.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2381–2391. Association for Computational Linguistics.

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

Linyong Nan, Dragomir R. Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Mutethia Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, Yi Chern Tan, Xi Victoria Lin, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. 2021. DART: open-domain structured data record to text generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 432–447. Association for Computational Linguistics.

Courtney Napoles, Matthew R. Gormley, and Benjamin Van Durme. 2012. Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction, AKBC-WEKEX@NAACL-HLT 2012, Montrèal, Canada, June 7-8, 2012*, pages 95–100. Association for Computational Linguistics.

Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, et al. 2022. Text and code embeddings by contrastive pre-training.

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.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. 2023. Toolllm: Facilitating large language models to master 16000+ real-world apis.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2020. Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, Chloe Hillier, and Timothy P Lillicrap. 2019. Compressive transformers for long-range sequence modelling.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 784–789. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *Logical Formalizations of Commonsense Reasoning, Papers from the 2011 AAAI Spring Symposium, Technical Report SS-11-06, Stanford, California, USA, March 21-23, 2011*. AAAI.

Ohad Rubin and Jonathan Berant. 2023. Long-range language modeling with self-retrieval.

Tapan Sahni, Chinmay Chandak, Naveen Reddy Chedeti, and Manish Singh. 2017. Efficient twitter sentiment classification using subjective distant supervision. In *9th International Conference on Communication Systems and Networks, COMSNETS 2017, Bengaluru, India, January 4-8, 2017*, pages 548–553. IEEE.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2023. Replug: Retrieval-augmented black-box language models.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Hongjin Su, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A Smith, Luke Zettlemoyer, Tao Yu, et al. 2022. One embedder, any task: Instruction-finetuned text embeddings.

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, et al. 2023. Llama: Open and efficient foundation language models.

Lewis Tunstall, Leandro Von Werra, and Thomas Wolf. 2022. Natural language processing with transformers.

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

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.

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.

Liang Wang, Nan Yang, and Furu Wei. 2023a. Learning to retrieve in-context examples for large language models.

Tianshi Wang, Li Liu, Huaxiang Zhang, Long Zhang, and Xiuxiu Chen. 2020. Joint character-level convolutional and generative adversarial networks for text classification. *Complex.*, 2020:8516216:1–8516216:11.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. 2023b. Augmenting language models with long-term memory.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers), pages 1112–1122. Association for Computational Linguistics.

Shitao Xiao and Zheng Liu. 2023. Baai general embedding.

Shitao Xiao, Zheng Liu, Weihao Han, Jianjin Zhang, Defu Lian, Yeyun Gong, Qi Chen, Fan Yang, Hao Sun, Yingxia Shao, et al. 2022. Distill-vq: Learning retrieval oriented vector quantization by distilling knowledge from dense embeddings.

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.

Jing Xu, Arthur Szlam, and Jason Weston. 2021. Beyond goldfish memory: Long-term open-domain conversation.

Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. 2023. Retrieval meets long context large language models. *CoRR*, abs/2310.03025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Yue Yu, Chenyan Xiong, Si Sun, Chao Zhang, and Arnold Overwijk. 2022. Coco-dr: Combating distribution shifts in zero-shot dense retrieval with contrastive and distributionally robust learning.

Zichun Yu, Chenyan Xiong, Shi Yu, and Zhiyuan Liu. 2023. Augmentation-adapted retriever improves generalization of language models as generic plug-in.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? 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 4791–4800. Association for Computational Linguistics.

Rui Zhang and Joel R. Tetreault. 2019. This email could save your life: Introducing the task of email subject line generation. 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 446–456. Association for Computational Linguistics.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: paraphrase adversaries from word scrambling. 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 1298–1308. Association for Computational Linguistics.

A Prompt Templates

Prompt A.1: Rank-Aware Reward (Knowledge)

Knowledge:
<Passage>

Q: <Question> A:

Prompt A.2: MMLU

Knowledge:
<Passage 1>
<Passage 2>
<Passage 3>

The following are multiple-choice questions (with answers) about <subject>.

<Question>
A. <Option 1>
B. <Option 2>
C. <Option 3>
D. <Option 4>
Answer:

Prompt A.3: PopQA

Knowledge:
<Passage 1>
<Passage 2>
<Passage 3>

Q: <Question 1> A: <Answer 1>
Q: <Question 2> A: <Answer 2>
. . .
Q: <Question 15> A: <Answer 15>
Q: <Question> A:

Prompt A.4: Multi-Session Chat

Speaker 1: <Retrieved/Recent Utterance 1>
Speaker 2: <Retrieved/Recent Utterance 2>
Speaker 1: <Utterance 1>
Speaker 2:

Dataset	Average Length
Books3	101010
Arxiv	26735
CodeParrot	217364
PG19	90447

Table 5: Average lengths of long-range language modeling datasets.

Prompt A.5: In-Context Learning

```
<Example 1 Input><Example 1 Ouptut>

<Example 2 Input><Example 2 Ouptut>
. . .
<Example 8 Input><Example 8 Ouptut>

<Input>
```

B Dataset Details

The detailed information of in-context learning datasets is reported in Table 6. The statistics of all training and evaluation datasets are reported in Table 7. The average lengths of long-range language modeling datasets are reported in Table 5.

C Implementation Details

C.1 Instructions

The instructions used for each retrieval task are shown in Table 8.

C.2 Training Settings

The hyper parameter settings for training LLM-Embedder are reported in Table 9.

D Impact of LLM-Embedder on Different LLMs

We evaluate the impact of LLM-Embedder when augmenting different LLMs to validate its generalization ability. Specifically, we utilize Aquila-7B-Chat (Aqu, 2023), Qwen-7B-Chat (Bai et al., 2023a), Baichuan2-7B-Chat (Baichuan, 2023), and Llama-2-13B-Chat (Touvron et al., 2023). The results are shown in Table 10. We report the average accuracy for MMLU, accuracy for PopQA, the average score for in-context learning, and perplexity for both Multi-Session Chat and Arxiv. Note that we do not replicate the evaluation of tool learning and conversational search because their performances are directly measured by retrieval metrics.

We can observe that our conclusions in Section 4.2 still hold. First of all, retrieval from the external world benefits LLM's performance in all four scenarios, since the performance of the plain LLM (i.e. None) underperforms retrieval-augmented one (BGE and LLM-Embedder). Besides, our proposed LLM-Embedder is able to generalize well to other LLMs and maintain its superiority over BGE on most datasets (PopQA and ICL in particular). This observation highlights the practical effectiveness and versatility of LLM-Embedder.

Dataset name	Category	#Train Sample	#Test Sample	Metric	Evaluation Strategy
ARC Challenge (Bhakthavatsalam et al., 2021)	Close QA	1,117	1,165	Accuracy	Likelihood
ARC Easy (Bhakthavatsalam et al., 2021)	Close QA	2,241	2,365	Accuracy	Likelihood
NQ (Kwiatkowski et al., 2019)	Close QA	87,925	3,610	Exact Match	Generation
COPA (Roemmele et al., 2011)	Commonsense	400	100	Accuracy	Likelihood
HellaSwag (Zellers et al., 2019)	Commonsense	39,905	10,042	Accuracy	Likelihood
PIQA (Bisk et al., 2020)	Commonsense	16,113 (held out)	1,838	Accuracy	Likelihood
Winogrande (Sakaguchi et al., 2021)	Coreference	40,398	1,267	Accuracy	Likelihood
WSC (Levesque, 2011)	Coreference	554	104	Accuracy	Likelihood
WSC273 (Levesque, 2011)	Coreference	0 (held out)	273	Accuracy	Likelihood
CommonGen (Lin et al., 2020)	Data-to-text	67,389	4,018	ROUGE-L	Generation
DART (Nan et al., 2021)	Data-to-text	62,659	2,768	ROUGE-L	Generation
E2E NLG (Dusek et al., 2019)	Data-to-text	33,525	1,847	ROUGE-L	Generation
MNLI (m) (Williams et al., 2018)	NLI	392,702	9,815	Accuracy	Likelihood
MNLI (mm) (Williams et al., 2018)	NLI	392,702	9,832	Accuracy	Likelihood
RTE (Bentivogli et al., 2009)	NLI	2,490	277	Accuracy	Likelihood
SNLI (Bowman et al., 2015)	NLI	549,367	9,824	Accuracy	Likelihood
QNLI (Rajpurkar et al., 2018)	NLI	104,743 (held out)	5,463	Accuracy	Likelihood
MRPC (Dolan and Brockett, 2005)	Paraphrase	3,668	408	Accuracy	Likelihood
PAWS (Zhang et al., 2019)	Paraphrase	49,401	8,000	Accuracy	Likelihood
QQP (DataCanary et al., 2017)	Paraphrase	363,846	40,430	Accuracy	Likelihood
BoolQ (Clark et al., 2019)	Reading Comp.	9,427	3,270	Accuracy	Likelihood
MultiRC (Khashabi et al., 2018)	Reading Comp.	27,243	4,848	F1	Likelihood
OpenBook QA (Mihaylov et al., 2018)	Reading Comp.	4,957	500	Accuracy	Likelihood
SQuAD v1 (Rajpurkar et al., 2016)	Reading Comp.	87,599	10,570	Exact Match	Generation
Sentiment140 (Sahni et al., 2017)	Sentiment	1,600,000	359	Accuracy	Likelihood
SST2 (Socher et al., 2013)	Sentiment	67,349	872	Accuracy	Likelihood
Yelp (Wang et al., 2020)	Sentiment	490,456 (held out)	33,285	Accuracy	Likelihood
AESLC (Zhang and Tetreault, 2019)	Summarize	13,181	1,750	ROUGE-L	Generation
AGNews (Zhang et al., 2015)	Summarize	120,000	7,600	Accuracy	Likelihood
Gigaword (Napoles et al., 2012)	Summarize	2,044,465	730	ROUGE-L	Generation
Total	n.a.	6.3M	177k	n.a.	n.a.
Total (sampled)	n.a.	591k	177k	n.a.	n.a.

Table 6: Detailed information of in-context learning datasets.

Scenario	Dataset	Corpus Size	#Training Samples	#Testing Samples
Knowledge Retrieval	MSMARCO	8841823	400870	–
	NQ	21051324	58622	–
	MMLU	8841823	–	14042
	PopQA	21051324	–	14267
	QReCC	54573064	29596	8209
Memory Retrieval	MSC	–	48925	2763
	Books3	–	10000	1000
	Arxiv	–	10000	757
	CodeParrot	–	10000	1000
	PG19	–	–	1000
Example Retrieval	Misc.	6283120	591359	177230
Tool Retrieval	ToolBench	10439	87322	100
Total	–	–	1333911	–

Table 7: Statistics of all training and evaluation datasets.

Scenario	Task	Input	Instruction
Knowledge Retrieval	Conversational Search	Query	Encode this query and context for searching relevant passages:
		Key	Encode this passage for retrieval:
	Others	Query	Represent this query for retrieving relevant documents:
		Key	Represent this document for retrieval:
Memory Retrieval	Long-Context Conversation	Query	Embed this dialogue to find useful historical dialogues:
		Key	Embed this historical dialogue for retrieval:
	Long-Range Language Modeling	Query	Embed this text chunk for finding useful historical chunks:
		Key	Embed this historical text chunk for retrieval:
Example Retrieval	In-Context Learning	Query	Convert this example into a vector to look for useful examples:
		Key	Convert this example into vector for retrieval:
Tool Retrieval	Tool Retrieval	Query	Transform this user request for fetching helpful tool descriptions:
		Key	Transform this tool description for retrieval:

Table 8: Instructions for each task.

#GPU	8×A100 (40G)
#Hard Negative (M)	7
#Sampled Outputs (N)	10
Batch Size Per GPU (B)	100
Optimizer	AdamW
Learning Rate (α)	5e-5
Learning Rate Checkpoint Step	1000
Weight Decay	0.01
Scheduler	Linear with warm-up of 0.2
Max Steps	10000
Gradient Checkpointing	✓

Table 9: Hyper parameter settings for fine-tuning.

LLM	**Embedder**	**MMLU**	**PopQA**	**ICL**	**MSC**	**Arxiv**
Llama-2-7B-Chat	None	0.460	0.206	0.465	19.350	3.765
	BGE	**0.490**	0.449	0.597	14.294	3.291
	LLM-Embedder	**0.490**	**0.505**	**0.627**	**13.483**	**3.232**
Aquila-7B-Chat	None	0.450	0.203	0.515	16.011	3.120
	BGE	0.483	0.398	0.573	14.184	2.791
	LLM-Embedder	**0.485**	**0.440**	**0.590**	**14.184**	**2.735**
Qwen-7B-Chat	None	0.556	0.239	0.535	21.047	2.789
	BGE	**0.579**	0.445	0.633	16.206	2.517
	LLM-Embedder	0.576	**0.478**	**0.646**	**15.452**	**2.482**
Baichuan2-7B-Chat	None	0.523	0.236	0.491	18.971	2.751
	BGE	**0.553**	0.441	0.596	16.076	2.444
	LLM-Embedder	0.551	**0.485**	**0.618**	**15.589**	**2.413**
Llama-2-13B-Chat	None	0.539	0.289	0.461	14.733	3.236
	BGE	**0.560**	0.460	0.620	11.688	2.904
	LLM-Embedder	0.558	**0.503**	**0.644**	**11.538**	**2.854**

Table 10: The impact of LLM-Embedder on different LLMs.