Efficient Long Context Language Model Retrieval with Compression

Anonymous ACL submission

Abstract

Long Context Language Models (LCLMs) have emerged as a new paradigm to perform Information Retrieval (IR), which enables the direct ingestion and retrieval of information by processing an entire corpus in their single context, showcasing the potential to surpass traditional sparse and dense retrieval methods. However, processing a large number of passages within in-context for retrieval is computationally expensive, and handling their representations during inference further exacerbates the processing time; thus, we aim to make LCLM retrieval more efficient and potentially more effective with passage compression. Specifically, we propose a new compression approach tailored for LCLM retrieval, which is trained to maximize the retrieval performance while minimizing the length of the compressed passages. To accomplish this, we generate the synthetic data, where compressed passages are automatically created and labeled as chosen or rejected according to their retrieval success for a given query, and we train the proposed **Co**mpression model for **Lo**ng context **R**etrieval (CoLoR) with this data via preference optimization while adding the length regularization loss on top of it to enforce brevity. Through extensive experiments on 9 datasets, we show that CoLoR improves the retrieval performance by 6% while compressing the in-context size by a factor of 1.91.

1 Introduction

The context size of Language Models (LMs) refers to the maximum number of tokens that the model can process in a single input, which has rapidly expanded, growing from a few hundred to 128K, and recently reaching 1M tokens in Long Context Language Models (LCLMs) (OpenAI, 2023; Reid et al., 2024; Anthropic, 2024). Notably, this expansion has unlocked new capabilities, enabling models to handle tasks that require extensive context lengths, such as summarization or question answering over long articles (Xu et al., 2023; Kim et al., 2024). In addition to this, LCLMs go beyond these relatively simple tasks to handle more complex tasks, such as Information Retrieval (IR) or Text-to-SQL, which not only demand long-range context understanding but also involve reasoning across multiple documents or structured queries (Lee et al., 2024; An et al., 2023; Liu et al., 2023; Xu et al., 2023). Furthermore, due to their impressive performance, they have established a new paradigm in LM utilization and task solving. For example, in IR tasks that we focus on, LCLMs are capable of processing an entire corpus with a large number of documents along with a user query in their single context, leading to more precise identification of relevant information and further surpassing traditional sparse or dense retrieval approaches in many cases (Lee et al., 2024). Yet, LCLMs face the limitation that the required computational resources scale with the input length, which has been overlooked by existing work.

To tackle this challenge, we propose a more efficient (and potentially more effective) method for LCLM retrieval. To be specific, instead of relying on the original passages, our approach uses compressed passages that retain the core information while filtering our irrelevant details, leading to the substantial reduction in input size. It is worth noting that, while there are existing compression methods (Jiang et al., 2023; Xu et al., 2024) particularly designed for text generation or retrieval-augmented generation, they are largely suboptimal for retrieval since they are not trained to prioritize key elements crucial for precise retrieval, such as relevance to the query or fine-grained document distinctions. In contrast, our compression model is trained to optimize the LCLM retrieval performance, while minimizing the passage length with the regularization term added on top of the optimization objective.

Specifically, to train our compression model, we leverage the strategy of Odds Ratio Preference Optimization (ORPO) (Hong et al., 2024), a method

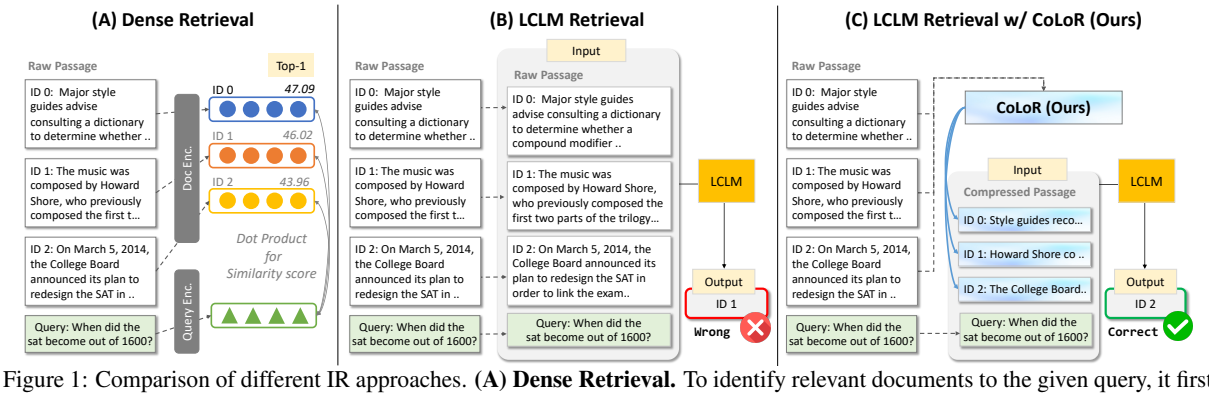

Figure 1: Comparison of different IR approaches. **(A) Dense Retrieval.** To identify relevant documents to the given query, it first embeds them into the vector space and then calculates their semantic similarity. **(B) LCLM Retrieval.** The LCLM takes and processes the raw passages from the corpus along with the query in the input context, and identifies the relevant passages. **(C) CoLoR.** We compress the raw passages, and use the compressed passages alongside the query as the LCLM input for retrieval.

well-suited for maximizing the preference by comparing pairs of samples and learning to prefer one over the other based on the specific objective. We note that this is particularly useful for our scenario since it allows us to rank compressed passages according to their retrieval performance, helping the model distinguish between more and less effective compressions, without the need for generating the ground-truth compression outputs manually. In other words, to generate the training data for it, we automatically create multiple compressed versions of each raw passage in the corpus and evaluate their retrieval success with the LCLM retrieval outcome. Subsequently, compressed samples are labeled as either chosen or rejected based on this evaluation, which can ultimately guide the model toward generating the effective compression for passages for retrieval during preference optimization. However, using ORPO alone might not sufficiently reduce the output length of the compressed passages. Thus, to overcome this, we further introduce a dynamic regularization term that adjusts the odds ratio loss in the training objective based on the length difference between chosen and rejected samples (where we additionally consider the compressed passages with the correct retrieval as rejected if there exist more shorter ones with the correct retrieval), which can encourage the model to prioritize brevity while at the same time optimizing the retrieval accuracy. We refer to our approach as **Co**mpression for **Lo**ng Context Language Model **R**etrieval (**CoLoR**), illustrated in Figure 1 with previous IR methods.

We experimentally validate the effectiveness of CoLoR on 9 datasets for LCLM retrieval, including 5 single-document and 4 multi-document retrieval scenarios. On a battery of tests conducted, we then demonstrate that our approach not only improves the retrieval performance by 6% but also reduces the context length for retrieval by a factor of 1.91 over the baseline that uses the original passages, and it is further superior to the existing text compression (or summarization) methods. Further, we show that our compression method can be generalizable to datasets not seen during its training.

2 Related Work

Information Retrieval The goal of Information Retrieval (IR) is to fetch documents relevant to a query, which has evolved significantly with their application to various tasks, such as web search and question answering. Early approaches used sparse retrieval methods, such as BM25 (Robertson and Zaragoza, 2009), which are based on lexical matching between queries and documents. As the field progressed, dense retrieval techniques have developed, leveraging text embedding models to capture richer semantic relationships between queries and documents. Notable examples of dense retrievers include SentenceBERT (Reimers and Gurevych, 2019), Dense Passage Retrieval (DPR) (Karpukhin et al., 2020), and SentenceT5 (Ni et al., 2021). More recently, researchers have begun transforming Large Language Models (LLMs) into retrieval systems (BehnamGhader et al., 2024), which aim to utilize the vast contextual understanding capability of LLMs in representing documents and queries. Following this line of approaches, our work extends by utilizing LCLMs as the retrieval mechanism.

Long Context Language Models The recent expansion in context length of LLMs, which is called Long Context Language Models (LCLMs), has empowered them to process and comprehend much larger amounts of information. Specifically, models like YaRN (Peng et al., 2023), Longformer (Beltagy et al., 2020), Gemini (Reid et al., 2024), GPT-4 (OpenAI, 2023), and Claude (Anthropic, 2024) exemplify this advancement. In tandem with these developments, new benchmarks have been intro-

duced to assess the capabilities of LCLMs across various tasks. For example, many studies (Bai et al., 2023; Li et al., 2023a; Liu et al., 2023; Yuan et al., 2024; Wang et al., 2024) evaluated their performance in long-context understanding, and there are also other studies that focused on more specialized areas such as code comprehension (Liu et al., 2024) and training-free in-context learning (Bertsch et al., 2024; Li et al., 2024). Moreover, Lee et al. (2024) demonstrated that LCLMs outperform traditional fine-tuned specialized models in several areas (such as IR). However, despite these promising results, Liu et al. (2023) highlighted the persistent challenges LCLMs face in fully grasping complex long contexts with high computational costs. In contrast to existing work that has mainly explored the potential and diverse applications of LCLMs, we take a different direction on improving the efficiency of LCLMs in the context of IR by reducing the context size while maintaining or enhancing performance.

Prompt Compression As LCLMs handle increasingly longer contexts, the corresponding rise in computational costs has sparked research into methods for prompt compression. Extractive compression is one common approach, where only the relevant tokens are retained. This often involves techniques such as token pruning, which require assessing the importance of individual tokens based on specific metrics, for example, utilizing the self-information or perplexity of the model (Jiang et al., 2023; Li et al., 2023b). However, these methods typically require access to the model's internal processes, making them feasible only for white-box models. In contrast, abstractive compression methods generate the condensed prompts without needing to preserve the original token order or structure, and can be applied to both black-box and white-box models as they do not rely on internal model access. For example, Xu et al. (2024) generate compressed content from multiple documents in Retrieval Augmented Generation (RAG) settings, and Wang et al. (2023) use a similar approach to generate distilled documents. Despite these advancements, previous work focusing on context compression in RAG or instruction-following tasks is not well-optimized for retrieval tasks, as it does not cater specifically to the needs of retrieval. To address this gap, we propose an abstractive compression model designed to improve the efficacy of LCLM retrieval.

Preference Optimization Aligning the language models with human preferences has become a key focus in improving response generation (Ouyang et al., 2022; Zhao et al., 2023; Rafailov et al., 2023; Hong et al., 2024). A prominent approach is Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022; Stiennon et al., 2020), which leverages a reinforcement learning framework where a policy model learns to evaluate and choose actions based on the state of an environment, with human feedback acting as a reward signal. Notably, what sets these approaches apart is their ability to train models using only preference selections on outputs, without needing explicit ground truth answers. Additionally, Odds Ratio Preference Optimization (ORPO) (Hong et al., 2024) further simplifies this by removing the requirement for a reference model during training, allowing single-step learning via preference selection of outputs. In our approach, we apply this framework to train the compression model without needing ground truth labels, enabling the model to learn based on a pair of compression outputs and their retrieval results.

3 Methodology

3.1 Problem Statement

We begin with formally explaining IR and LCLMs.

Information Retrieval In a typical IR task, given a query $\boldsymbol{q}$, its objective is to retrieve a ranked list k relevant entries from a corpus $\mathcal{C}$, formulated as follows: $\{\boldsymbol{d}_i\}_{i=1}^{k} = \texttt{Retriever}(\boldsymbol{q}, \mathcal{C})$, where $\boldsymbol{d}_i$ is a document from $\mathcal{C}$. The query $\boldsymbol{q}$ is typically textual, and $\mathcal{C}$ is a collection of documents. Traditionally, `Retriever` is operationalized with the sparse retrieval based on lexical term matching (Robertson and Zaragoza, 2009) or the neural embedding-based dense methods (Karpukhin et al., 2020).

Long Context Language Model Retrieval Recently, Long Context Language Models (LCLMs) have emerged with the ability to process extended contexts, enabling them to handle inputs spanning dozens of documents. This capability has given rise to a new paradigm called LCLM Retrieval, which utilizes LCLMs to solve IR tasks (Lee et al., 2024). To be formal, similar to the typical IR approaches, LCLM retrieval aims to retrieve relevant documents from the corpus $\mathcal{C}$ for the query $\boldsymbol{q}$, which can be represented as follows: $\{\boldsymbol{d}_i\}_{i=1}^{k} = \texttt{LCLM}(\mathcal{T}(\boldsymbol{q}, \mathcal{C}))$, where $\mathcal{T}$ is the prompt template which serves as the structured format that outlines the context for LCLMs (including task descriptions) to direct them in performing retrieval. It is worth noting that, unlike traditional retrieval methods (sparse or dense),

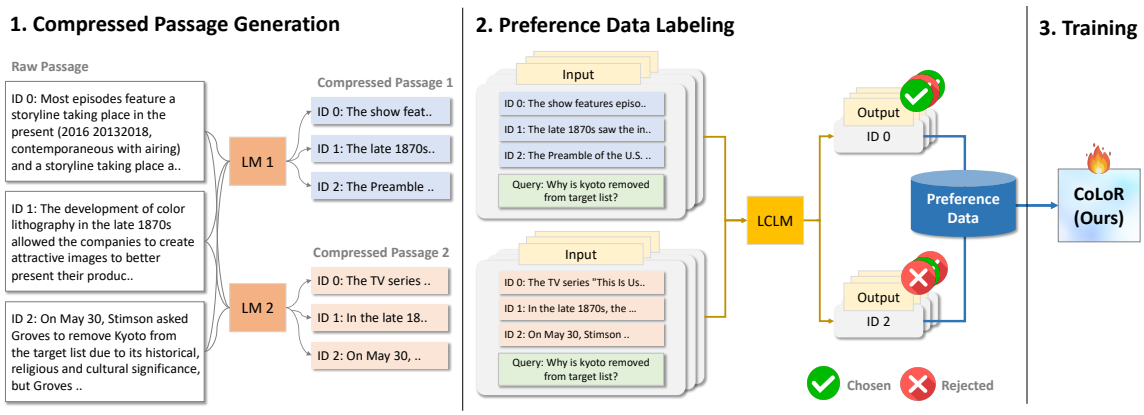

Figure 2: **Overview of Training Processes for CoLoR**. 1. We first create the training data for CoLoR by generating multiple compressed passages from their original passages with multiple LMs. 2. The compressed passages and their associated query are used as input to the LCLM, and their retrieval performance is measured to label them as either chosen or rejected based on retrieval results. 3. CoLoR is trained using the pairs of chosen and rejected compressed passages obtained from previous steps.

which involve pairing each document with a query and calculating similarity scores to rank documents, LCLM retrieval takes both the entire corpus and the query as the single input and directly identifies relevant documents within it. However, a significant challenge with LCLM retrieval lies in the use of raw documents (or passages) as input, as they often contain unnecessary, redundant, and irrelevant context, leading to increased computational costs.

3.2 CoLoR: Compression for Long Context Language Model Retrieval

To tackle the inefficiency of using raw passages in LCLM retrieval, we propose using compressed passages to reduce the computational overhead without compromising retrieval effectiveness. There are two common approaches to compressing passages: using prompt-based methods with LLMs or leveraging off-the-shelf compression models. However, prompt-based methods often fail to achieve optimal compression ratios and are not tailored to enhance IR performance. Likewise, existing compression models, which are typically designed for tasks other than IR (such as RAG or instruction following), are not well-suited for LCLM retrieval.

To address these limitations, we propose CoLoR (Compression for Long Context Retrieval), which is a novel compression model designed for LCLM retrieval. CoLoR generates compressed passages by learning to balance two objectives: maintaining high retrieval accuracy and reducing passage length. To achieve this, we leverage preference optimization using synthetic preference data, where compressed passages are automatically generated, and labeled based on their retrieval success as well as their resulting lengths. This allows the model to distinguish between more and less effective compressions without manually collecting labels.

Formally, let $\boldsymbol{d}_i \in \mathcal{C}$ represent a raw document (or passage) from the corpus. Our goal is to apply the CoLoR model to compress each document $\boldsymbol{d}_i$ into a more concise representation $\boldsymbol{c}_i = \texttt{CoLoR}(\boldsymbol{d}_i)$, where $\boldsymbol{c}_i$ is the compressed version of the raw document. Ideally, the compressed passage $\boldsymbol{c}_i$ retains the most relevant information while filtering out unnecessary details, therefore, reducing the length of the input to the LCLM. After compressing every document in the corpus, during the retrieval process, instead of directly using the original corpus $\mathcal{C}$, the LCLM ingests the compressed corpus $\mathcal{C}^* = \{\boldsymbol{c}_i\}_{i=1}^{|\mathcal{C}|}$, where each element in $\mathcal{C}$ is transformed by CoLoR. In other words, the retrieval process can be redefined as follows: $\{\boldsymbol{d}_i\}_{i=1}^{k} = \texttt{LCLM}(\mathcal{T}(\boldsymbol{q}, \mathcal{C}^*))$ with $|\mathcal{T}(\boldsymbol{q}, \mathcal{C}^*)| \ll |\mathcal{T}(\boldsymbol{q}, \mathcal{C})|$ where $|\cdot|$ measures the number of tokens in the resulting prompt.

3.2.1 Training Recipe for CoLoR

We now turn to explaining the details of how we train our CoLoR to optimize efficiency while improving retrieval accuracy, illustrated in Figure 2.

Data Collection To train CoLoR for LCLM retrieval, we need to create a new dataset as no such datasets are available. Our data creation process begins with leveraging multiple LLMs to generate multiple compressed versions of raw documents (or passages), by prompting them with the prompt template: $\mathcal{T} =$ `Summarize the following content: {passage}`, formalized as follows: $\boldsymbol{c} = \texttt{LLM}(\mathcal{T}(\boldsymbol{d}))$ where $\boldsymbol{c}$ is the compressed passage and $\boldsymbol{d}$ is its original version. After that, the compressed passages are used as inputs for the LCLM retrieval, which are then labeled as either chosen or rejected based on two criteria: 1) whether the compressed passage is correctly retrieved in response to its associated query and 2) whether its length is shorter than any

of the other successfully retrieved compressed passages. For instance, if several compressed versions of a passage are retrieved correctly, the shortest of these is labeled as chosen, while the others are labeled as rejected. Also, if the retrieval with the compressed passage fails, it is labeled as rejected. This allows us to create the dataset with pairs of chosen and reject compression results for training CoLoR with preference optimization, which can ultimately prioritize compressions that improve retrieval accuracy while minimizing passage length.

CoLoR Optimization To optimize our compression model, we leverage the Odds Ratio Preference Optimization (ORPO) (Hong et al., 2024), an approach designed for training models by comparing pairs of chosen and rejected samples without the need for a reference model. ORPO is particularly suited for our task, as it allows us to directly optimize the model to prefer compressed passages that yield better retrieval performance with the shortest length. Formally, the standard ORPO loss function measures the odds ratio between the likelihood of generating the chosen response $\boldsymbol{y}_w$ and the rejected response $\boldsymbol{y}_l$, represented as follows:

$$\mathcal{L}_{\text{ORPO}} = \mathbb{E}_{(\boldsymbol{q},\boldsymbol{y}_w,\boldsymbol{y}_l)} \left[\mathcal{L}_{\text{SFT}} + \lambda \cdot \mathcal{L}_{\text{OR}}\right],$$

where $\mathcal{L}_{\text{SFT}}$ is the supervised fine-tuning loss for the chosen response based on the causal language modeling negative log-likelihood, and $\mathcal{L}_{\text{OR}}$ is the loss for the odds ratio of the chosen response over the rejected one (See Hong et al. (2024) for details).

However, while ORPO enables effective preference learning for making the compression model, it does not inherently reduce the length of the compressed passages as much as desired. To overcome this, we further propose to use a dynamic regularization term that adjusts the odds ratio loss based on the length difference between rejected and chosen samples. Specifically, we redefine the ORPO loss by multiplying it with a specific factor determined as the length difference between $\boldsymbol{y}_l$ and $\boldsymbol{y}_w$ (where $\boldsymbol{y}_l$ is always longer than $\boldsymbol{y}_w$ based on the criteria in our data collection process), as follows:

$$\mathcal{L}_{\text{CoLoR}} = \mathbb{E}_{(\boldsymbol{q},\boldsymbol{y}_w,\boldsymbol{y}_l)} \left[\mathcal{L}_{\text{SFT}} + \lambda \cdot \mathcal{L}_{\text{OR}} \cdot (|\boldsymbol{y}_l| - |\boldsymbol{y}_w|)\right],$$

where $|\cdot|$ measures the length of the compressed passage. This extra regularization term directly allows the model to make larger updates in cases where the chosen sample is significantly shorter than the rejected one, making it to favor concise outputs without sacrificing retrieval accuracy.

4 Experiment Setup

4.1 Datasets

We evaluate the performance of CoLoR on 9 widely used LCLM retrieval benchmark datasets, following the setup from Lee et al. (2024), including 5 single-document retrieval datasets: FEVER, FIQA, MS MARCO, NQ, and SciFact (Thorne et al., 2018; Maia et al., 2018; Campos et al., 2016; Kwiatkowski et al., 2019; Wadden et al., 2020) and 4 multi-document retrieval datasets: HotpotQA, MuSiQue, QAMPARI, and QUEST (Yang et al., 2018; Trivedi et al., 2021; Amouyal et al., 2022; Malaviya et al., 2023). Note that single-document retrieval tasks involve retrieving a single document relevant to a query, whereas multi-document retrieval tasks require retrieving two or more documents. We provide the detailed statistics in Table 7.

4.2 Baselines and Our Model

We evaluate CoLoR against baselines, as follows: 1. **Raw Passage** – which is a standard approach for LCLM retrieval that directly uses raw passages. 2. **Document Title** – which uses only the titles of the passages without the full content. 3. **Zero-Shot Compression** – which uses LLMs to compress passages via prompting, including GPT-4o-mini (OpenAI, 2023) and Phi3 (Abdin et al., 2024). 4. **Selective Context** – which is an extractive compression method that selects tokens based on the self-information of the model (Li et al., 2023b), where we use two compression rates: 0.3 and 0.6. 5. **LLMLingua** – which is an extractive compression method that selects tokens based on the perplexity scores (Jiang et al., 2023). 6. **RECOMP** – which is an abstractive compression method that compresses multiple documents designed for RAG scenarios (Xu et al., 2024). 7. **COMPACT** – which is an abstractive compression method that compresses and refines passages iteratively for question answering (Yoon et al., 2024). 8. **CoLoR (Ours)** – which is our abstractive compression method, trained to maximize the retrieval accuracy and minimize the compressed passage length, with the preference optimization. Additionally, we also include BM25 (Robertson and Zaragoza, 2009) (which is a sparse retriever that scores documents based on term frequency and inverse document frequency) and DPR (Karpukhin et al., 2020) (which is a dense retriever that uses embeddings to match queries and relevant passages) as the reference to the performance of conventional sparse and dense retrievers, which are neither comparable nor our competitors.

Table 1: **Results on LCLM retrieval**. **Type** refers to a compression type: ✗ denotes no compression, Ex. denotes extractive compression, and Ab. denotes abstractive compression. † denotes multi-document retrieval. Comp. is the compression rate.

		FEVER		FIQA		MS MARCO		NQ		SciFact	
Methods	**Type**	R@1	Comp.	R@1	Comp.	R@1	Comp.	R@1	Comp.	R@1	Comp.
Raw Passage	✗	0.95	1.00x	0.63	1.00x	0.90	1.00x	0.97	1.00x	0.57	1.00x
Document Title	✗	0.97	31.65x	N/A	N/A	N/A	N/A	0.86	22.56x	0.68	13.98x
BM25	✗	0.93	1.00x	0.41	1.00x	0.78	1.00x	0.79	1.00x	0.75	1.00x
DPR	✗	0.89	1.00x	0.31	1.00x	0.85	1.00x	0.94	1.00x	0.35	1.00x
Selective Context (0.6)	Ex.	0.96	1.83x	0.35	1.83x	0.52	1.92x	0.95	1.88x	0.71	1.89x
Selective Context (0.3)	Ex.	0.95	4.58x	0.15	4.24x	0.19	4.69x	0.90	4.91x	0.63	5.33x
LLMLingua	Ex.	0.96	1.70x	0.46	1.6x	0.83	1.12x	0.97	1.24x	0.74	1.92x
Comp. w/ GPT	Ab.	0.96	1.74x	0.68	2.09x	0.92	1.39x	0.99	1.37x	0.73	1.71x
Comp. w/ Phi	Ab.	0.95	1.83x	0.68	2.31x	0.92	1.41x	0.99	1.39x	0.71	1.78x
RECOMP	Ab.	0.96	3.02x	0.36	2.00x	0.75	1.77x	0.98	1.97x	0.70	2.31x
COMPACT	Ab.	0.96	2.21x	0.48	2.56x	0.88	1.17x	0.98	1.32x	0.77	2.95x
CoLoR (Ours)	Ab.	0.94	2.15x	0.73	2.82x	0.95	1.63x	0.98	1.62x	0.75	2.12x

		HotPotQA†		MuSiQue†		QAMPARI†		QUEST†		Average	
Methods	**Type**	F1@2	Comp.	F1@5	Comp.	F1@5	Comp.	F1@3	Comp.	Perf.	Comp.
Raw Passage	✗	0.87	1.00x	0.35	1.00x	0.56	1.00x	0.33	1.00x	0.68	1.00x
Document Title	✗	0.65	12.52x	0.38	23.43x	0.25	22.48x	0.11	55.65x	0.60	26.04x
BM25	✗	0.82	1.00x	0.39	1.00x	0.76	1.00x	0.37	1.00x	0.64	1.00x
DPR	✗	0.79	1.00x	0.46	1.00x	0.55	1.00x	0.31	1.00x	0.61	1.00x
Selective Context (0.6)	Ex.	0.79	2.06x	0.37	1.93x	0.42	1.95x	0.19	2.01x	0.58	1.92x
Selective Context (0.3)	Ex.	0.68	5.49x	0.37	5.10x	0.30	5.30x	0.11	5.41x	0.48	5.01x
LLMLingua	Ex.	0.81	1.13x	0.36	1.23x	0.54	1.05x	0.21	1.75x	0.65	1.42x
Comp. w/ GPT	Ab.	0.87	1.20x	0.40	1.32x	0.54	1.28x	0.32	1.95x	0.71	1.56x
Comp. w/ Phi	Ab.	0.85	1.21x	0.41	1.39x	0.55	1.33x	0.31	2.03x	0.71	1.63x
RECOMP	Ab.	N/A	0.76x	0.38	1.00x	0.53	0.97x	0.21	1.50x	0.54	1.70x
COMPACT	Ab.	0.87	1.14x	0.37	1.47x	0.52	1.51x	0.30	3.47x	0.68	1.98x
CoLoR (Ours)	Ab.	0.86	1.37x	0.42	1.55x	0.55	1.50x	0.33	2.39x	0.72	1.91x

4.3 Evaluation Metrics

For single-document retrieval, we evaluate performance with **Recall@1 (R@1)**, which measures the proportion of queries for which the top-ranked document is relevant. For multi-document, we use **F1@k**, a metric combining **Precision@k** (the proportion of correctly retrieved relevant documents in the top k results) and **Recall@k** (the proportion of relevant documents retrieved from up to k total). For compression efficiency, we compute the **compression rate (Comp.)**, defined as the average number of tokens in raw passages divided by the average number of tokens in compressed passages.

4.4 Implementation Details

To ensure a fair comparison across all experiments, we use GPT-4o-mini as the underlying LCLM. We use the Phi-3-mini-4k-instruct model as the base model for our compression method, CoLoR. For the prompt, we structure it as a sequence of the corpus, 5-shot examples, and query, following Lee et al. (2024). Additional details are in Appendix A.

5 Experiment Results

Main Results We report main results in Table 1, demonstrating that the proposed CoLoR approach consistently outperforms all baseline methods on both the single-document and multi-document retrieval tasks while at the same time substantially compressing the input context size of LCLMs for retrieval. Specifically, CoLoR achieves a compression rate that reduces the input size by a factor of 1.91, while also improving retrieval performance by 6%, compared to the standard approach with raw passages. Also, our CoLoR provides the superior quality compressed passages for retrieval, compared to extractive and abstractive compression models. For instance, when compared with extractive methods (Selective Context and LLMLingua), CoLoR consistently demonstrates better retrieval performance. In addition to this, even against the

Table 3: **Results on out-of-domain datasets**, where the target datasets within each category is excluded for training CoLoR*.

	Raw Passage		Comp w/ Phi		CoLoR*	
	Perf.	Comp.	Perf.	Comp.	Perf.	Comp.
Fact-checking						
FEVER	0.95	1.00x	0.95	1.83x	0.95	2.14x
SciFact	0.71	1.00x	0.71	1.78x	0.73	2.13x
Average	0.83	1.00x	0.83	1.81x	0.84	2.14x
Multi-document						
HotpotQA	0.85	1.00x	0.85	1.21x	0.87	1.36x
MuSiQue	0.35	1.00x	0.41	1.39x	0.40	1.54x
QAMPARI	0.56	1.00x	0.55	1.33x	0.56	1.48x
QUEST	0.33	1.00x	0.31	2.03x	0.32	2.33x
Average	0.52	1.00x	0.53	1.49x	0.54	1.68x
Argument						
ArguAna	0.28	1.00x	0.27	2.26x	0.34	2.73x
Touché-2020	0.76	1.00x	0.79	3.79x	0.79	4.66x
Average	0.52	1.00x	0.53	3.03x	0.57	3.70x

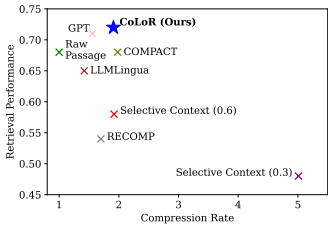

Figure 3: **The trade-off visualization of different methods**, showing their compression rate (x-axis) and retrieval performance (y-axis).

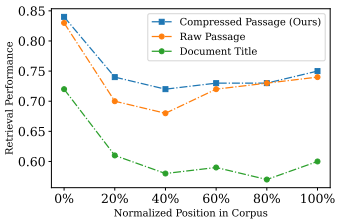

Figure 4: **Results with varying the position of (compressed) passages** associated with the query within the corpus (0% represents beginning).

	FIQA		SciFact		MuSiQue		Average	
	R@1	Comp.	R@1	Comp.	F1@5	Comp.	Perf.	Comp.
Phi3 mini	0.68	2.31x	0.71	1.78x	0.41	1.39x	0.68	1.63x
+ CoLoR	0.67	2.82x	0.75	2.12x	0.42	1.55x	0.72	1.91x
Mistral 7B	0.60	1.55x	0.73	1.46x	0.39	1.09x	0.69	1.21x
+ CoLoR	0.63	3.07x	0.80	3.03x	0.40	1.71x	0.71	2.25x
Llama 3.2	0.58	2.19x	0.70	2.17x	0.39	1.63x	0.69	1.84x
+ CoLoR	0.61	2.83x	0.72	3.03x	0.39	1.71x	0.69	2.15x

Table 2: **Results with varying the base LM for CoLoR**, such as Phi3, Mistral, and Llama. Note that Average indicates the average across all 9 datasets, and we report results with other datasets in Table 9.

Table 4: **Manual evaluation results** on three sampled passages per dataset. We report the average number of total facts (Facts), query-supportive facts (Sup. Facts), the proportion of supportive facts to total facts (Ratio), and the token count.

Methods	Facts	Sup. Facts	Ratio	Tokens
Raw Passages	14.13	1.91	13.52	210.93
CoLoR (Ours)	9.26	1.74	18.80	93.41

strong proprietary and open-source models (such as GPT and Phi3), CoLoR excels, particularly in terms of the compression rate, highlighting the limitations of relying on prompting techniques to generate compressed passages for retrieval. Lastly, when compared with multi-document compression methods such as RECOMP and COMPACT, our CoLoR significantly outperforms them in the retrieval performance with the similar compression rate, which further confirms the necessity of task-specific training for passage compression for LCLM retrieval.

Results on Out-of-Domain Datasets To assess the generalizability of our compression approach (CoLoR) on datasets not seen for training, we evaluate its performance in out-of-domain settings by excluding a set of datasets from each retrieval category (such as fact-checking, multi-document, and argument) from the training process and testing on them. As shown in Table 3, we observe that CoLoR consistently enhances retrieval performance while significantly reducing the input context size, which demonstrates the ability of our CoLoR to generalize across diverse retrieval tasks and datasets.

Qualitative Analysis with Manual Evaluation To see whether query-relevant information is preserved after passage compression, we manually compare the atomic facts in the compressed passages to the ones in the raw passages over randomly sampled three examples from each of all datasets with two individual persons. As shown in Table 4, while the total number of facts in the compressed passages decreases, the number of query-relevant facts is only slightly reduced (from 1.91 to 1.74 per passage on average). Also, when we look at the proportion of relevant facts to total facts (Ratio), this proportion increases, indicating that the compressed passages contain a higher density of query-related atomic facts (while the proportion of noisy, query-irrelevant information is reduced), which may support the performance improvement of our CoLoR. Additionally, we provide the case study on the compressed passages in Figure 11.

Trade Off Between Compression Rate and Retrieval Performance To examine the trade-off between the compression rate and retrieval performance, we visualize and analyze them in Figure 3. First of all, we observe that, while extremely high compression rates (such as those achieved by using the compression ratio of 0.3 with the Selective Context baseline) drastically reduce the input size, they also lead to significant information loss (potentially due to the removal of crucial information for retrieval), resulting in the retrieval performance drop of 20 on average compared to using the raw passages. This observation highlights the critical trade-off between compression and performance: simply maximizing compression for efficiency compromises accuracy. In contrast, the proposed CoLoR effectively balances this trade-off, ensuring that the reduction in context size does not sacrifice critical information, thanks to our training strategy that guides the model to prefer the compressed passages of successful retrieval over the ones with unsuccessful retrieval (while enforcing brevity as well).

Analysis on Passage Position In Figure 5, we analyze the position of passages within the input context (associated with the query in the same context), to see the potential lost-in-the-middle problem (Liu et al., 2023) in the context of LCLM retrieval: the retrieval performance can be decreased if the relevant passages to the query are placed in the middle of the input sequence. To measure this, we place documents at intervals of 0%, 20%, 40%, 60%, 80%, and 100% across the input cor-

Table 5: **Results of an ablation study**, where SFT refers to supervised fine-tuning and ORPO w/ Reg refers to our full CoLoR model with the dynamic regularization term. Average indicates the average performance across all 9 datasets.

	MS MARCO		MuSiQue		QUEST		Average	
	R@1	Comp.	F1@5	Comp.	F1@3	Comp.	Perf.	Comp.
Base Model	0.92	1.41x	0.41	1.39x	0.31	2.03x	0.71	1.63x
+ SFT	0.88	1.42x	0.39	1.39x	0.32	2.18x	0.72	1.65x
+ ORPO	0.94	1.59x	0.40	1.53x	0.31	2.31x	0.71	1.86x
+ ORPO w/ Reg.	0.95	1.41x	0.42	1.55x	0.33	2.39x	0.72	1.91x

Table 6: **Results with different retrieval approaches with raw and compressed passages**. In the first couple of columns, Types refers to retrieval types, and Formats refers to corpus formats. Average indicates the performance over all 9 datasets.

		FEVER		MS MARCO		HotpotQA		Average	
Types	**Formats**	R@1	Comp.	R@1	Comp.	F1@2	Comp.	Perf.	Comp.
BM25	Raw Passage	0.93	1.00x	0.78	1.00x	0.82	1.00x	0.67	1.00x
	Comp. w/ GPT	0.91	1.74x	0.78	1.39x	0.80	1.20x	0.64	1.56x
	CoLoR (Ours)	0.91	2.15x	0.70	1.63x	0.77	1.37x	0.62	1.91x
DPR	Raw Passage	0.89	1.00x	0.85	1.00x	0.79	1.00x	0.61	1.00x
	Comp. w/ GPT	0.89	1.74x	0.85	1.39x	0.77	1.20x	0.62	1.56x
	CoLoR (Ours)	0.91	2.15x	0.88	1.63x	0.81	1.37x	0.63	1.91x
LCLM	CoLoR (Ours)	0.94	2.15x	0.95	1.63x	0.86	1.37x	0.72	1.91x

pus, and compare three different approaches: Raw Passage, Document Title, and our CoLoR. Then, similar to the finding in Liu et al. (2023), placing documents towards the middle leads to a certain level of performance degradation across all methods. Yet, interestingly, our proposed compression method (CoLoR) mitigates the lost-in-the-middle issue, since it not only filters out irrelevant information within passages during compression but also allows for a more compact use of the input context.

CoLoR with Different LMs To see whether the proposed CoLoR is versatile across different underlying LMs in generating compressed passages, we vary them with three different LMs: Phi-3-mini-4k-instruct, Mistral-7B-Instruct-v0.3, and Llama-3.2-3B-Instruct. Then, as shown in Table 2, we observe the consistent improvements of our CoLoR in both compression rate and retrieval performance across all models. This demonstrates that CoLoR and its training methodology is not limited to a specific model but can be effectively generalized to others. We provide results with all datasets in Table 9.

Ablation Study To see the effectiveness of each component of our CoLoR, we perform an ablation study and present the results in Table 5. First of all, we observe that, while Supervised Fine-Tuning (SFT) yields strong retrieval performance, the compression rate remains comparable to the untrained method. On the other hand, by utilizing preference optimization with Odds Ratio Preference Optimization (ORPO), we observe an improved compression rate, though this comes with a slight performance degradation. However, the proposed dynamic regularization term (for compressed passage length) mitigates this trade-off, further improving both the compression ratio and retrieval performance, reaffirming the overall efficacy of our proposed CoLoR approach in both the efficiency and effectiveness. More detailed results are in Table 10 of Appendix.

Adaption of CoLoR to Conventional Retrieval In Table 6, we investigate how using compressed passages impacts the performance of conventional sparse and dense retrievers as they can bring an additional benefit of faster indexing thanks to the reduced passage size. First, for the sparse retriever (BM25), performance tends to decrease when using compressed passages, likely due to the loss of lexical information that BM25 relies on to match documents based on exact lexical similarities. In contrast, the dense retriever (DPR) shows performance improvements with compressed passages. We conjecture that this may be because the underlying LM for dense retrieval already contains much of the passage's information within its parameters, and, as a result, compressing the passage still retains essential details in making valuable representations for it while additionally filtering our irrelevant content (that might lead to noise in embedding). However, despite these gains in dense retrieval with the proposed CoLoR, the performance of LCLM retrieval coupled with CoLoR is substantially better than conventional retrieval methods.

6 Conclusion

In this work, we introduced **Co**mpression for **Lo**ng Context Langauge Model **R**etrieval (**CoLoR**), a method specifically designed to improve efficiency and effectiveness of LCLM retrieval by transitioning from raw to compressed passages. Specifically, the proposed CoLoR, trained with the synthesized preference data (based on retrieval outcomes of the compressed passages) and regularization loss for their lengths, optimizes both brevity and retrieval performance. Through our extensive experiments conducted across 9 datasets spanning single- and multi-document retrieval tasks, we demonstrated that CoLoR not only achieves a 6% improvement in retrieval performance but also reduces context size by a factor of 1.91 over the standard LCLM retrieval, which further surpasses existing text compression methods. These highlight the significant advantage of compressed passages to enhance efficiency for LCLM retrieval by reducing the computational load and its associated costs, all while even improving retrieval accuracy, making it more scalable and practical for real-world applications.

Limitations

While our proposed CoLoR approach demonstrates significant advantages in LCLM retrieval, there are still areas that future work may explore. First, following the LCLM retrieval benchmark setup (Lee et al., 2024), our experiments are conducted with a maximum context length of 128K tokens, and, while this context length is indeed very large and it has been increasingly extended further, in real-world applications, the size of the corpus can be much larger (even after utilizing our compression method), which may necessitate further modifications of the overall LCLM retrieval framework. Yet, developing the new process for LCLM retrieval is beyond the scope of our work and we leave it as future work. Another consideration is the compression process: it introduces an additional step before retrieval; however, this is not a big deal as it only needs to be performed once as like the indexing process of sparse and dense retrieval approaches.

Ethics Statement

It is worth noting that, similar to any other retrieval approaches, the retrieval corpus may contain harmful or offensive content, and the compressed passages could potentially reflect these biases. Also, additional biases may be induced during the training process of LCLMs. Although addressing these concerns are obviously beyond the scope of our work, we acknowledge the importance of implementing the safeguards in future research to ensure that the retrieval process remains safe and fair.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Hassan Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Singh Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio C'esar Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, Allison Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Dan Iter, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Young Jin Kim, Mahoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Chen Liang, Weishung Liu, Eric Lin, Zeqi Lin, Piyush Madan, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Xianmin Song, Olatunji Ruwase, Xin Wang, Rachel Ward, Guanhua Wang, Philipp Witte, Michael Wyatt, Can Xu, Jiahang Xu, Sonali Yadav, Fan Yang, Ziyi Yang, Donghan Yu, Cheng-Yuan Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yunan Zhang, and Xiren Zhou. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *ArXiv*, abs/2404.14219.

Samuel Joseph Amouyal, Tomer Wolfson, Ohad Rubin, Ori Yoran, Jonathan Herzig, and Jonathan Berant. 2022. Qampari: A benchmark for open-domain questions with many answers. In *IEEE Games Entertainment Media Conference*.

Chen 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. *ArXiv*, abs/2307.11088.

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku.

Yushi Bai, Xin Lv, Jiajie Zhang, Hong Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *ArXiv*, abs/2308.14508.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. Llm2vec: Large language models are secretly powerful text encoders. *ArXiv*, abs/2404.05961.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *ArXiv*, abs/2004.05150.

Amanda Bertsch, Maor Ivgi, Uri Alon, Jonathan Berant, Matthew R. Gormley, and Graham Neubig. 2024. In-context learning with long-context models: An in-depth exploration. *ArXiv*, abs/2405.00200.

Daniel Fernando Campos, Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, Li Deng, and Bhaskar Mitra. 2016. Ms marco: A human generated machine reading comprehension dataset. *ArXiv*, abs/1611.09268.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Orpo: Monolithic preference optimization without reference model. *ArXiv*, abs/2403.07691.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. Llmlingua: Compressing prompts for accelerated inference of large language models. In *Conference on Empirical Methods in Natural Language Processing*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Yu Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering. *ArXiv*, abs/2004.04906.

Yekyung Kim, Yapei Chang, Marzena Karpinska, Aparna Garimella, Varun Manjunatha, Kyle Lo, Tanya Goyal, and Mohit Iyyer. 2024. Fables: Evaluating faithfulness and content selection in book-length summarization. *ArXiv*, abs/2404.01261.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc V. Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Jinhyuk Lee, Anthony Chen, Zhuyun Dai, Dheeru Dua, Devendra Singh Sachan, Michael Boratko, Yi Luan, S'ebastien M. R. Arnold, Vincent Perot, Sid Dalmia, Hexiang Hu, Xudong Lin, Panupong Pasupat, Aida Amini, Jeremy R. Cole, Sebastian Riedel, Iftekhar Naim, Ming-Wei Chang, and Kelvin Guu. 2024. Can long-context language models subsume retrieval, rag, sql, and more? *ArXiv*, abs/2406.13121.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. 2023a. Loogle: Can long-context language models understand long contexts? *ArXiv*, abs/2311.04939.

Tianle Li, Ge Zhang, Quy Duc Do, Xiang Yue, and Wenhu Chen. 2024. Long-context llms struggle with long in-context learning. *ArXiv*, abs/2404.02060.

Yucheng Li, Bo Dong, Chenghua Lin, and Frank Guerin. 2023b. Compressing context to enhance inference efficiency of large language models. In *Conference on Empirical Methods in Natural Language Processing*.

Jiawei Liu, Jia Le Tian, Vijay Daita, Yuxiang Wei, Yifeng Ding, Yuhan Katherine Wang, Jun Yang, and Lingming Zhang. 2024. Repoqa: Evaluating long context code understanding. *ArXiv*, abs/2406.06025.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. Www'18 open challenge: Financial opinion mining and question answering. *Companion Proceedings of the The Web Conference 2018*.

Chaitanya Malaviya, Peter Shaw, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2023. Quest: A retrieval dataset of entity-seeking queries with implicit set operations. *ArXiv*, abs/2305.11694.

Jianmo Ni, Gustavo Hernández Abrego, Noah Constant, Ji Ma, Keith B. Hall, Daniel Matthew Cer, and Yinfei Yang. 2021. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. *ArXiv*, abs/2108.08877.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. 2022. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *ArXiv*, abs/2309.00071.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *ArXiv*, abs/2305.18290.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy P. Lillicrap, Jean-Baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, Ioannis Antonoglou, Rohan Anil, Sebastian Borgeaud, Andrew M. Dai, Katie Millican, Ethan Dyer, Mia Glaese, Thibault Sottiaux, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, James Molloy, Jilin Chen, Michael Isard, Paul Barham, Tom Hennigan, Ross McIlroy, Melvin Johnson, Johan Schalkwyk, Eli Collins, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, Clemens Meyer, Gregory Thornton, Zhen Yang, Henryk Michalewski, Zaheer Abbas, Nathan Schucher, Ankesh Anand, Richard Ives, James Keeling, Karel Lenc, Salem Haykal, Siamak Shakeri, Pranav Shyam, Aakanksha Chowdhery, Roman Ring, Stephen Spencer, Eren Sezener, and et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *CoRR*, abs/2403.05530.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Conference on Empirical Methods in Natural Language Processing*.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3:333–389.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan J. Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2020. Learning to summarize from human feedback. *ArXiv*, abs/2009.01325.

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

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *ArXiv*, abs/1803.05355.

H. Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2021. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

David Wadden, Kyle Lo, Lucy Lu Wang, Shanchuan Lin, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Conference on Empirical Methods in Natural Language Processing*.

Chonghua Wang, Haodong Duan, Songyang Zhang, Dahua Lin, and Kai Chen. 2024. Ada-leval: Evaluating long-context llms with length-adaptable benchmarks. In *North American Chapter of the Association for Computational Linguistics*.

Zhiruo Wang, Jun Araki, Zhengbao Jiang, Md. Rizwan Parvez, and Graham Neubig. 2023. Learning to filter context for retrieval-augmented generation. *ArXiv*, abs/2311.08377.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 38–45. Association for Computational Linguistics.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2024. Recomp: Improving retrieval-augmented lms with context compression and selective augmentation. In *International Conference on Learning Representations*.

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

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing*.

Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. 2024. Compact: Compressing retrieved documents actively for question answering. *ArXiv*, abs/2407.09014.

Tao Yuan, Xuefei Ning, Dong Zhou, Zhijie Yang, Shiyao Li, Minghui Zhuang, Zheyue Tan, Zhuyu Yao, Dahua Lin, Boxun Li, Guohao Dai, Shengen Yan, and Yu Wang. 2024. Lv-eval: A balanced long-context benchmark with 5 length levels up to 256k. *ArXiv*, abs/2402.05136.

Yao Zhao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J. Liu. 2023. Slic-hf: Sequence likelihood calibration with human feedback. *ArXiv*, abs/2305.10425.

Dawei Zhu, Liang Wang, Nan Yang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. 2024. Longembed: Extending embedding models for long context retrieval. *ArXiv*, abs/2404.12096.

Table 7: Statistics of the generated dataset for preference optimization for training CoLoR, which includes the number of samples per dataset and the average length of reject and chosen tokens. † denotes multi-document retrieval datasets.

Dataset	# of Samples	Avg. Rejected Token	Avg. Chosen Token
FEVER	483	236.19	152.63
FIQA	455	148.15	88.22
MS MARCO	1061	76.06	51.3
NQ	635	158.16	111.67
SciFact	198	229.84	156.26
HotPotQA†	544	93.52	71.11
MuSiQue†	12	107.58	80.5
QAMPARI†	30	107.23	82.77
Total	3418	135.61	91.36

A Additional Experimental Setups

Fine-tuning Details For Supervised Fine-Tuning (SFT), we use a learning rate of 5e-6. Similarly, for ORPO with Phi and Llama, we use λ of 2.5 and a learning rate of 1e-6, while, for Mistral, we change the learning rate to 5e-6. Also, all models are trained for 10 epochs with a batch size of 8, and the best epoch is selected based on the validation set. Lastly, we use the TRL library[1] for training.

Computational Resources We train and inference all baselines and our model by using one of the NVIDIA RTX A6000 and NVIDIA RTX A5000 GPUs, depending on their availability. With those GPUs, the time required to train our model over 10 epochs ranges from 3 to 5 hours.

Deep Learning Libraries In our experiments, we utilize the following deep learning libraries: PyTorch (Paszke et al., 2019), Transformers (Wolf et al., 2020), SentenceTransformers (Reimers and Gurevych, 2019), and BEIR (Thakur et al., 2021). Also, BM25 is implemented using a python library rank_bm25[2], while, for DPR, we use a BEIR framework[3]. Other baselines are sourced from publicly available checkpoints on their repositories[4][5].

Datasets Details In table 7, we provide the statistics of the dataset that we create for training our compression model (CoLoR). Note that, among all the samples, we use 3,077 samples for training and 341 samples for validation, with random selection.

B Additional Experimental Results

Results on Long Context Retrieval Benchmark We further evaluate our CoLoR on the long context retrieval scenario, including two long context question-answering datasets from the LongEmbed benchmark (Zhu et al., 2024) as well as the original corpus for the Natural Questions (NQ) dataset[6]. To enable comparisons between different methods, the raw passages are truncated (as the context size with original raw passages exceeds its limit) and the compressed passages are generated using GPT-4o-mini (prompted to create summaries under 200 words)[7]. Then, as shown in Table 8, the compression model reduces passage size by 216.2×, while increasing Recall@1 by 42%, compared to using (truncated) raw passages, which further strengths the effectiveness of our compression paradigm, particularly in handling lengthy passages.

Table 8: Results on the long context retrieval benchmark.

	NQ		2WikimQA		NarrativeQA		Average	
	R@1	Comp.	R@1	Comp.	R@1	Comp.	R@1	Comp.
Raw Passage	0.92	1.00x	0.42	1.00x	0.22	1.00x	0.52	1.00x
Compressed Passage	0.95	63.02x	0.72	29.64x	0.56	555.93x	0.74	216.2x

Full Results on Analyses In Table 9 and Table 10, we provide the full results of varying the base LM for CoLoR and the ablation study with all datasets, respectively. Also, we provide the results of analysis on passage position with all datasets in Figure 5.

Case Study We provide the case study on the compressed passages generated by different approaches in Table 11, which shows that the compressed passages from our CoLoR tend to leading to the retrieval success and tend to be shorter.

Prompt Details For the prompt construction, we follow the Corpus-in-Context prompting approach from prior work (Lee et al., 2024). An example prompt for the NQ dataset is provided in Table 12, and, for more examples and details on the prompt, please refer to Lee et al. (2024).

[1] https://github.com/huggingface/trl
[2] https://github.com/dorianbrown/rank_bm25
[3] https://github.com/beir-cellar/beir
[4] https://huggingface.co/cwyoon99/CompAct-7b
[5] https://github.com/liyucheng09/Selective_Context
[6] https://github.com/google-research-datasets/natural-questions
[7] Due to the excessive length of passages for these datasets, training CoLoR on them is not feasible within our computational resources, and we leave this as a future work.

Table 9: **Full results with all datasets by varying the base LM for CoLoR**. † indicates multi-document retrieval datasets, and * denotes out-of-domain datasets (that are not used for training CoLoR).

	FEVER		FIQA		MS MARCO		NQ		SciFact	
Methods	R@1	Comp.	R@1	Comp.	R@1	Comp.	R@1	Comp.	R@1	Comp.
Phi3 mini	0.95	1.83x	0.68	2.31x	0.92	1.41x	0.99	1.39x	0.71	1.78x
+ CoLoR	0.94	2.15x	0.73	2.82x	0.95	1.63x	0.98	1.62x	0.75	2.12x
Mistral 7B	0.96	1.36x	0.58	1.55x	0.90	0.98x	0.98	1.01x	0.70	1.46x
+ CoLoR	0.96	2.51x	0.63	3.07x	0.91	1.66x	0.98	1.78x	0.80	3.03x
Llama 3.2	0.96	2.03x	0.35	2.19x	0.52	1.47x	0.95	1.68x	0.71	2.17x
+ CoLoR	0.95	2.30x	0.61	2.83x	0.86	2.76x	0.99	1.91x	0.72	2.64x

	HotPotQA†		MuSiQue†		QAMPARI†		QUEST†*		Average	
Methods	F1@2	Comp.	F1@5	Comp.	F1@5	Comp.	F1@3	Comp.	Perf.	Comp.
Phi3 mini	0.85	1.21x	0.41	1.39x	0.55	1.33x	0.31	2.03x	0.68	1.63x
+ CoLoR	0.86	1.37x	0.42	1.55x	0.55	1.50x	0.33	2.39x	0.72	1.91x
Mistral 7B	0.83	0.92x	0.39	1.09x	0.55	1.07x	0.32	1.46x	0.69	1.21x
+ CoLoR	0.85	1.43x	0.40	1.71x	0.56	1.68x	0.33	3.35x	0.71	2.25x
Llama 3.2	0.84	1.42x	0.39	1.63x	0.52	1.64x	0.31	2.33x	0.69	1.84x
+ CoLoR	0.85	1.55x	0.40	1.84x	0.54	1.83x	0.30	2.68x	0.69	2.15x

Table 10: **Full results of the ablation study with all datasets.** † denotes multi-document retrieval datasets, and * indicates out-of-domain datasets (not used for training CoLoR). SFT refers to supervised fine-tuning, and ORPO w/ Reg denotes CoLoR.

	FEVER		FIQA		MS MARCO		NQ		SciFact	
Methods	R@1	Comp.	R@1	Comp.	R@1	Comp.	R@1	Comp.	R@1	Comp.
Base Model	0.95	1.83x	0.68	2.31x	0.92	1.41x	0.99	1.39x	0.71	1.78x
+ SFT	0.95	1.82x	0.67	2.28x	0.88	1.42x	0.99	1.42x	0.80	1.83x
+ ORPO	0.94	2.11x	0.64	2.75x	0.94	1.59x	0.99	1.59x	0.74	2.05x
+ ORPO w/ Reg	0.94	2.15x	0.73	2.82x	0.95	1.63x	0.98	1.62x	0.75	2.12x

	HotPotQA†		MuSiQue†		QAMPARI†		QUEST†*		Average	
Methods	F1@2	Comp.	F1@5	Comp.	F1@5	Comp.	F1@3	Comp.	Perf.	Comp.
Base Model	0.85	1.21x	0.41	1.39x	0.55	1.33x	0.23	2.03x	0.71	1.63x
+ SFT	0.88	1.22x	0.39	1.39x	0.57	1.33x	0.32	2.18x	0.72	1.65x
+ ORPO	0.85	1.35x	0.40	1.53x	0.54	1.48x	0.31	2.31x	0.71	1.86x
+ ORPO w/ Reg	0.86	1.37x	0.42	1.55x	0.55	1.50x	0.33	2.39x	0.72	1.91x

Table 11: **Case study** on the retrieval sample from the FIQA dataset.

Methods (# of Tokens)	**Passage**	**Prediction**
Query	If an index goes up because an underlying company issues more shares, what happens to the ETF	
Raw Passage (315)	If a stock that makes up a big part of the Dow Jones Industrial Average decided to issue a huge number of additional shares, that will make the index go up. At least this is what should happen, since an index is basically a sum of the market cap of the contributing companies. No, indices can have various weightings. The DJIA is a price-weighted index not market-cap weighted. An alternative weighting besides market-cap and price is equal weighting. From Dow Jones: Dow Jones Industrial Average2̆122. Introduced in May 1896, the index, also referred to as The Dow0̆0ae, is a price-weighted measure of 30 U.S. blue-chip companies. Thus, I can wonder what in the new shares makes the index go up? If a stock is split, the Dow divisor is adjusted as one could easily see how the current Dow value isn't equal to the sum or the share prices of the members of the index. In other cases, there may be a dilution of earnings but that doesn't necessarily affect the stock price directly as there may be options exercised or secondary offerings made. SO if the index, goes up, will the ETF DIA also go up automatically although no additional buying has happened in the ETF itself? If the index rises and the ETF doesn't proportionally, then there is an arbitrage opportunity for someone to buy the DIA shares that can be redeemed for the underlying stocks that are worth more in this case. Look at the Creation and Redemption Unit process that exists for ETFs.	325818 ✗
Comp. w/ GPT (185)	The Dow Jones Industrial Average (DJIA) is a price-weighted index of 30 U.S. blue-chip companies, meaning its value is based on the share prices of its components rather than their market capitalizations. If a major stock in the DJIA issues a large number of new shares, the index should theoretically rise, but adjustments like the Dow divisor may affect this relationship. When a stock splits, for example, the divisor is adjusted to maintain the index's integrity. If the index increases, it raises the question of whether the corresponding ETF (DIA) will also rise automatically, despite no new buying in the ETF. If the ETF's price doesn't align proportionally with the index's increase, it may create an arbitrage opportunity, allowing traders to profit by redeeming DIA shares for the underlying, more valuable stocks. Understanding the creation and redemption unit process is essential for grasping this dynamic.	325818 ✗
Comp. w/ Phi (138)	The content discusses how the Dow Jones Industrial Average (DJIA) index, a price-weighted measure of 30 U.S. blue-chip companies, can be affected by the issuance of additional shares by a major company. It explains that the index value may not directly reflect the sum of the share prices due to adjustments in the Dow divisor. The content also addresses the relationship between the DJIA index and the ETF DIA, stating that if the index rises but the ETF doesn't proportionally, there may be an arbitrage opportunity for someone to buy DIA shares and redeem them for the underlying stocks that are worth more. The creation and redemption process of ETFs is briefly mentioned."	325818 ✗
LLMLingua (189)	a stock that the IndustrialAverage to a of that will . least what, since is a market the., can have various The index. Aning. From: Average™. in index also as Dowed ., can what the new shares is the Dow is adjusted as one could see the value't sum the the members of the index. In other cases, there may be a dilution of earnings but that doesn't necessarily affect the stock price directly as there may be options exercised or secondary offerings made. SO if the index, goes up, will the ETF DIA also go up automatically although no additional buying has happened in the ETF itself? If the index rises and the ETF doesn't proportionally, then there is an arbitrage opportunity for someone to buy the DIA shares that can be redeemed for the underlying stocks that are worth more in this case. Look at the Creation and Redemption Unit process that exists for ETFs.	325818 ✗
Selective Context (0.3) (69)	of the Dow Jones Industrial Average to of, up. is what happen of of....: Dow Jones Industrial Average in, the to as is of., what up is, isn't to of of., be ofn't necessarily the stock price may be. up, up?n't then is for to buy be for are more. at.	482415 ✗
Selective Context (0.6) (175)	up a big part of the Dow Jones Industrial Average to a huge number of additional shares, that make the index go up. is what happen, is a sum of the market cap of., can. is a price-weighted index weighted. and is equal weighting.: Dow Jones Industrial Average in May, the also referred to as is a price-weighted measure of 30 U.S. blue-chip companies., I can what go up? is, is as see the current Dow valuen't equal to the sum the share prices of the members of the index., there may be a dilution of that doesn't necessarily affect the stock price as there may be or made. up, will go up has happened in the ETF itself? andn't then there is an arbitrage opportunity for to buy be redeemed for the underlying stocks that are more in this case. at for	418150 ✓
COMPACT (116)	The Dow Jones Industrial Average (DJIA) or "Dow 30" is a price-weighted average of 30 significant stocks traded on the New York Stock Exchange (NYSE) and the Nasdaq. The DJIA is one of the most widely followed equity indices. The DJIA is calculated by summing the individual prices of the 30 stocks in the index, then dividing that total by a divisor. If a stock that makes up a big part of the Dow Jones Industrial Average decided to issue a huge number of additional shares, that will make the index go up.	325818 ✗
CoLoR (Ours) (103)	A stock issuing additional shares can cause the Dow Jones Industrial Average (DJIA) to rise due to its price-weighted nature. However, the DJIA's value isn't directly equal to the sum of its member stocks' prices. In cases of stock splits or dilution of earnings, the Dow divisor is adjusted. If the DJIA rises and the ETF DIA doesn't proportionally increase, there's an arbitrage opportunity for someone to buy DIA shares and redeem them for the underlying stocks worth more.	418150 ✓

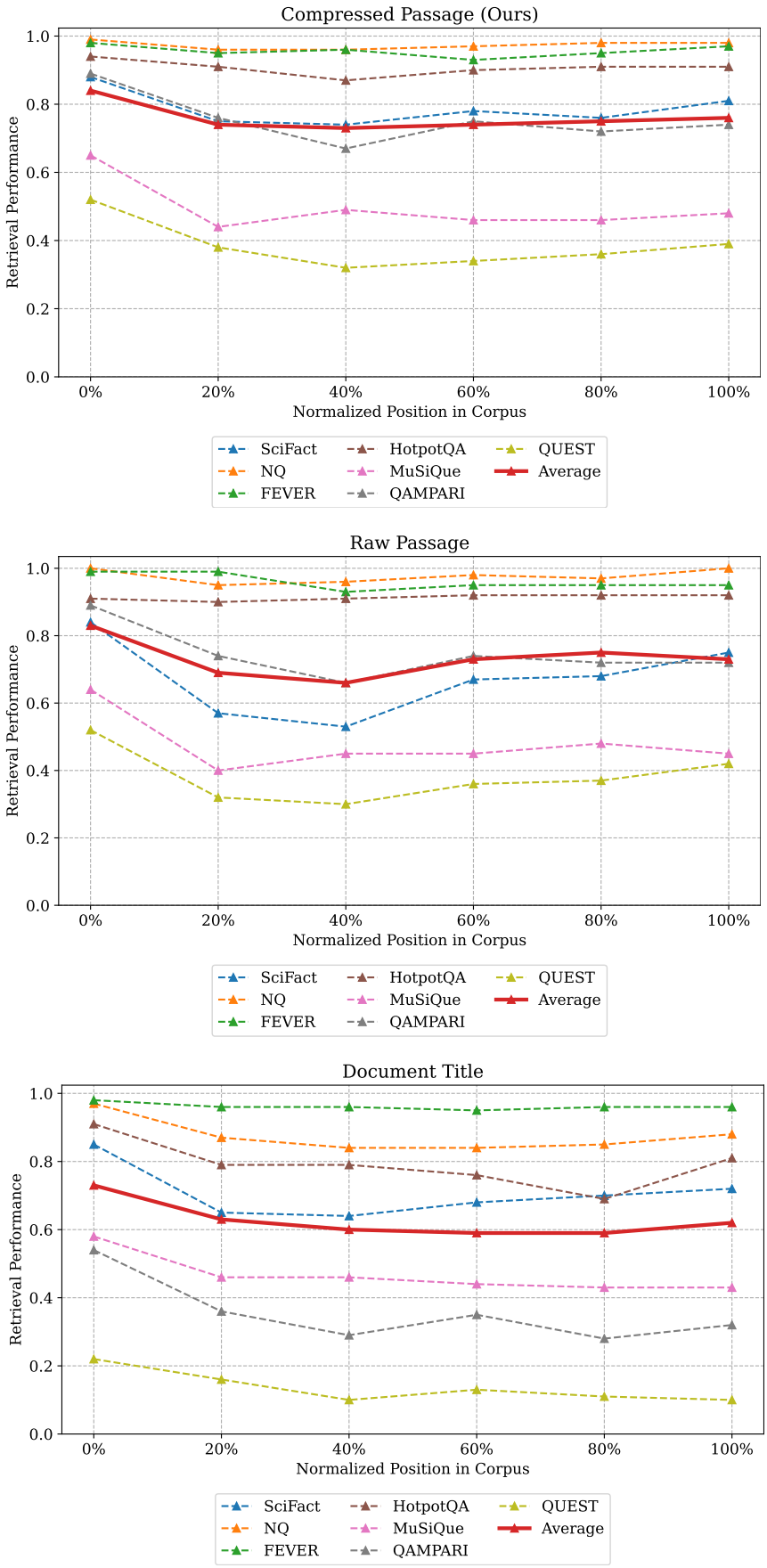

Figure 5: **Results with varying the position of (compressed) passages for all datasets**. Specifically, we arbitrarily adjust the positions of the gold and few-shot passages within the corpus relative to the query (0% represents the beginning). The figures at the top, middle, and bottom represent the results with CoLoR, raw passage, and document title, respectively.

Table 12: **Example of corpus-in-context prompting** for the NQ dataset, following Lee et al. (2024). The input is categorized by type, with all types being provided as input to the LCLM for retrieval.

Types	
Instruction	You will be given a list of documents. You need to read carefully and understand all of them. Then you will be given a query, and your goal is to find all documents from the list that can help answer the query. Print out the ID and TITLE of each document. Your final answer should be a list of IDs, in the following format: Final Answer: [id1, id2, ...] If there is only one ID, it should be in the format: Final Answer: [id1] If there is no perfect answer output the closest one. Do not give an empty final answer.
Corpus Formatting	ID: 0 | TITLE: English compound | CONTENT: Major style guides advise consulting a dictionary to determine whether . . . | END ID: 0 ID: 1 | TITLE: The Lord of the Rings: The Return of the King | CONTENT: The music was composed by Howard Shore . . . | END ID: 1 . . . ID: 881 | TITLE: Dexter (season 3) | CONTENT: While stalking a murderous drug dealer . . . | END ID: 881 ID: 882 | TITLE: Interstellar medium | CONTENT: In the series of investigations . . . | END ID: 882
Few-shot Examples	====== Example 1 ====== Which document is most relevant to answer the query? Print out the TITLE and ID of the document. Then format the IDs into a list. If there is no perfect answer output the closest one. Do not give an empty final answer. query: where did the dewey decimal system come from The following documents can help answer the query: TITLE: Dewey Decimal Classification | ID: 199 Final Answer: ['199'] . . .
Query Formatting	====== Now let's start! ====== Which document is most relevant to answer the query? Print out the TITLE and ID of the document. Then format the IDs into a list. If there is no perfect answer output the closest one. Do not give an empty final answer. query: when does monday night raw come on hulu The following documents can help answer the query: