Self-Retrieval: End-to-End Information Retrieval with One Large Language Model

Qiaoyu Tang[1,2]*, **Jiawei Chen**[1,2]*, **Zhuoqun Li**[1,2], **Bowen Yu**[3], **Yaojie Lu**[1], **Cheng Fu**[3], **Haiyang Yu**[3], **Hongyu Lin**[1]†, **Fei Huang**[3], **Ben He**[1,2], **Xianpei Han**[1]†, **Le Sun**[1], **Yongbin Li**[3]†
[1]Chinese Information Processing Laboratory, Institute of Software, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
[3]Alibaba Group
{tangqiaoyu2020,jiawei2020,lizhuoqun2021}@iscas.ac.cn
{luyaojie,hongyu,xianpei,sunle}@iscas.ac.cn
{yubowen.ybw,fucheng.fuc,yifei.yhy,f.huang,shuide.lyb}@alibaba-inc.com
benhe@ucas.ac.cn

Abstract

The rise of large language models (LLMs) has significantly transformed both the construction and application of information retrieval (IR) systems. However, current interactions between IR systems and LLMs remain limited, with LLMs merely serving as part of components within IR systems, and IR systems being constructed independently of LLMs. This separated architecture restricts knowledge sharing and deep collaboration between them. In this paper, we introduce *Self-Retrieval*, a novel end-to-end LLM-driven information retrieval architecture. Self-Retrieval unifies all essential IR functions within a single LLM, leveraging the inherent capabilities of LLMs throughout the IR process. Specifically, Self-Retrieval internalizes the retrieval corpus through self-supervised learning, transforms the retrieval process into sequential passage generation, and performs relevance assessment for reranking. Experimental results demonstrate that Self-Retrieval not only outperforms existing retrieval approaches by a significant margin, but also substantially enhances the performance of LLM-driven downstream applications like retrieval-augmented generation. [3]

1 Introduction

Recently, information retrieval (IR) systems and large language models (LLMs) have witnessed a growing synergy, with advancements in one field driving progress in the other [13, 56]. On one hand, IR systems have proven effective in augmenting LLMs and mitigating challenges such as hallucinations and outdated knowledge [22, 16]. By providing accurate, up-to-date external knowledge, IR systems significantly enhance the reliability and performance of LLMs. On the other hand, the powerful language understanding and generation capabilities of LLMs have been leveraged to enhance almost all components of traditional IR systems–indexing, retrieval [42, 9, 26], and reranking [58, 27, 40]. Through the integration of LLMs into the IR pipeline, these systems achieve substantially improved retrieval accuracy [57, 1].

However, current IR systems typically adopt a pipeline architecture where different components operate in isolation, limiting LLMs' role to specific components rather than leveraging their full

* Equally Contribution.
† Corresponding authors.
[3]The code of this work is available at https://github.com/icip-cas/SelfRetrieval.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

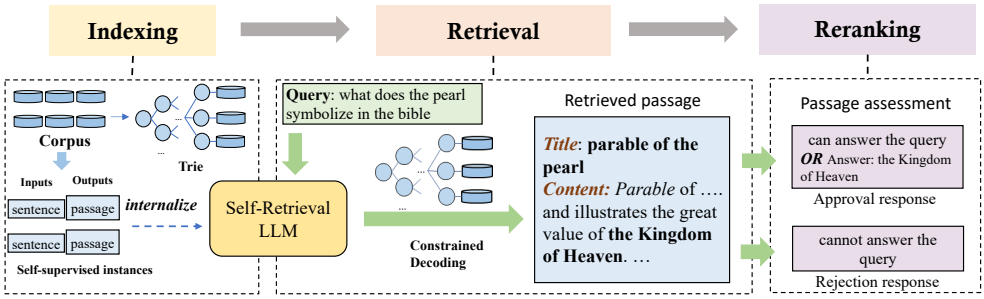

Figure 1: The Self-Retrieval framework consists of three key components: (1) corpus indexing through self-supervised learning, (2) passage generation via constrained decoding, (3) passage ranking using self-assessment scoring.

potential across the entire system. This fragmented approach creates several challenges: it hinders knowledge sharing between components, prevents deep integration of LLMs' diverse capabilities, and results in complex implementations with potentially sub-optimal performance. These limitations underscore the need for a more unified approach that fully integrates LLMs across all components of the IR system. Such an approach would not only maximize the utility of LLMs' capabilities but also simplify system implementation while potentially achieving better performance through enhanced component synergy.

In this paper, we introduce Self-Retrieval, an end-to-end information retrieval architecture driven entirely by one large language model. This integration is not trivial due to the inherent mismatch between information retrieval tasks and text generation, particularly in ensuring accurate document generation using language models. As illustrated in Figure 1, Self-Retrieval consolidates the separate components of an IR system - indexing, retrieval, and reranking - into the parameters of a single LLM. For indexing, the corpus is internalized into the LLM's parameters through self-supervised learning, enabling the model to encode and store corpus information within its internal representations. During retrieval, Self-Retrieval leverages its encoded knowledge of the corpus to semantically match the input query and directly generates the relevant documents as outputs. To ensure the generated documents exactly match those in the original corpus, we employ the constrained decoding algorithm [10, 8, 24] based on the trie of the corpus. For reranking, Self-Retrieval performs self-assessment on the retrieved documents to evaluate their relevance. The output score is used to rerank the retrieved passages. Moreover, for downstream tasks such as retrieval-augmented generation (RAG), Self-Retrieval integrates the reader component into the model, enabling direct answer generation following retrieval. Through this end-to-end approach, Self-Retrieval fully leverages LLMs' powerful capabilities in language understanding, matching, assessment, and generation to achieve unified information retrieval.

We evaluate Self-Retrieval on three representative retrieval benchmarks: NQ, TriviaQA, and MS MARCO. Experimental results demonstrate that Self-Retrieval substantially outperforms existing sparse retrieval, dense retrieval, and generative retrieval methods on both document-level and passage-level retrieval tasks. Furthermore, our experiments on retrieval-augmented generation tasks reveal that Self-Retrieval considerably enhances downstream performance. Additionally, larger LLMs lead to progressively better performance in Self-Retrieval, showing clear scaling benefits. These results demonstrate the effectiveness of Self-Retrieval across different retrieval tasks and application scenarios.

The potential impacts of this paper may include the following aspects. First, we introduce Self-Retrieval, an end-to-end architecture that consolidates the entire information retrieval system within a single large language model. This unified approach demonstrates substantial performance improvements over existing IR methods. Second, the corpus internalization and indexing mechanism of Self-Retrieval establishes a new paradigm to memorize, organize and retrieve the learned documents (at least part of them) during the pre-training phase, paving the way for more transparent and trustworthy text generation from LLMs. Third, as a LLM-driven retrieval system, Self-Retrieval offers inherent advantages in terms of compatibility, consistency, and interaction with LLMs' internal knowledge. Through experiments on RAG, we demonstrate how this natural compatibility leads to superior performance, suggesting broader potential for enhancing various LLM-based applications.

2 Related Work

LLM for IR Recent studies have explored leveraging LLMs to enhance various components of IR systems, including query rewriting, retrieval, and reranking. For query rewriting, LLMs have been employed to generate pseudo-documents for query expansion [46] and to rewrite queries based on conversational context [15]. In the retrieval stage, researchers have explored augmenting data by generating pseudo-queries [6, 17] or relevance labels [25] using LLMs, as well as employing LLMs directly as generative retrievers [42, 5]. Regarding reranking, LLMs have been utilized in two ways: serving as rerankers directly [27, 40] and augmenting the reranking dataset [12]. While these methods have advanced specific components within the IR pipeline, Self-Retrieval distinguishes itself by presenting an end-to-end architecture driven entirely by a single LLM, eliminating the need for external components.

Dense retrieval Dense retrieval models retrieve information by matching dense vector representations of queries and documents [19]. In this paradigm, an encoder transforms both queries and documents into dense vectors, with relevance determined by their vector distance. Various strategies have been proposed to enhance dense retrievers, including designing loss functions [45], multi-vector [38], training with synthetic queries [33, 47], and leveraging large-scale query-document pairs [30, 50]. Recent work has also explored using large language models to generate dense vectors for both queries and documents [29]. However, the fundamental limitation of dense retrieval lies in its limited interaction with LLMs, as the compression of natural language into dense vectors inherently constrains the utilization of LLMs' sophisticated language understanding and semantic inference capabilities.

Generative retrieval Generative retrieval methods leverage sequence-to-sequence language models to generate document identifiers for a given query [8, 42]. This paradigm is pioneered by GENRE [7], which introduces the concept of entity retrieval through constrained beam search generation of entity names. DSI [42] extends it to document retrieval by training T5 models to generate document-specific identifiers. The field has since evolved through various innovations, including query generation techniques [11, 59], sophisticated identifier design [48, 51], architectural improvements [5, 36], and continual learning strategies [20, 14].

Most relevant to our work, Yu et al.[52] proposed a "generate-then-read" approach, advocating for the use of LLMs to directly generate documents instead of relying on a retriever. UniGen [23] proposed a unified framework that integrates generative retrieval and question answering through a dual-decoder architecture. Compared to them, Self-Retrieval ensures accurate document generation through constrained decoding and accomplishes both retrieval and answer generation in one turn.

The main distinctions between Self-Retrieval and existing generative retrieval methods can be summarized as follows: (1) Self-Retrieval enables LLMs to directly generate document content rather than relying on other text or numeric identifiers. This approach aligns naturally with LLMs' pre-training objectives, preserves their inherent knowledge, and eliminates the need for complex identifier construction schemes. (2) Self-Retrieval further integrates components such as reranking and answer generation into the framework, further expanding its scope and enhancing the retrieval performance. These distinctions highlight that Self-Retrieval represents a more natural and effective approach for leveraging the capabilities of LLMs in information retrieval.

3 Self-Retrieval

In this section, we introduce our proposed Self-Retrieval. The overall architecture is illustrated in Figure 1. Different from traditional information retrieval systems that separate indexing, retrieval, and reranking components, Self-Retrieval integrates these functionalities directly into the parameters of a single large language model:

- **Indexing**: Self-Retrieval internalizes the entire corpus into its parameters through self-supervised learning, enabling the model to process passages internally without relying on external indices.
- **Retrieval**: Given an input query q, Self-Retrieval generates relevant passage p using the knowledge embedded within its parameters, which is different from dense retrieval or generative retrieval that rely on embedding or document identifiers as proxies of passage.

- **Reranking**: After generating passage p, Self-Retrieval assesses its relevance to the query q through self-assessment. The output logits provide the basis for reranking candidate passages.

Through this unified approach, Self-Retrieval enables a streamlined, end-to-end process that enhances the overall effectiveness of information retrieval. In the following sections, we detail each component of our method.

3.1 Indexing: Internalize the Corpus

Self-Retrieval integrates indexing into the LLM's parameters through self-supervised learning, enabling the model to internalize the entire corpus. Unlike generative retrieval methods that rely on complex document identifiers and identifier matching, Self-Retrieval employs a straightforward sentence-to-passage task to construct the index. Specifically, given a passage $p = \{s_1, s_2, ..., s_L\}$ consisting of L sentences, each sentence s_i is provided as input to the LLM with parameters θ. The training objective is to generate the source passage p in an auto-regressive way, represented as $P(p|s_i, \theta)$. This self-supervised indexing approach offers several advantages. First, it provides a simple yet effective method for corpus indexing. Second, it naturally frames the indexing process as a retrieval-like task, enabling the model to simultaneously internalize the corpus and develop retrieval capabilities using a consistent data format. Furthermore, this indexing technique closely aligns with the pre-training processes of language models, suggesting that our method could be considered as continued pre-training on the corpus. Through this process, the LLM learns to efficiently memorize and organize corpus information within its parameters.

3.2 Retrieval: Generate Relevant Passage through Constrained Decoding

Retrieval serves as a first-pass filter to collect passages related to the input query. In Self-Retrieval, we train the LLM to directly generate relevant passages in response to queries, eliminating the need for intermediaries such as embedding in dense retrieval or document identifier in generative retrieval. Specifically, given the query q and corpus $\mathcal{D}$, Self-Retrieval first generates a potential document title $\hat{t}$ as global information, formulated as $P(\hat{t}|q; \theta)$. The model then generates a relevant passage, denoted as $P(\hat{p}|q, \hat{t}; \theta)$.

However, since LLMs are general-purpose pre-trained models rather than statistical frequency models, the generated passage $\hat{p}$ may not exactly match any passage in $\mathcal{D}$, making it challenging to locate the corresponding passages in the corpus. To address this challenge, we employ a trie-based constrained decoding algorithm [10, 8, 24]. This approach restricts generated tokens to a dynamically constrained vocabulary. We construct a prefix tree $\mathcal{T}$ from corpus $\mathcal{D}$, where each path from the root to a leaf node represents a unique passage in the corpus, and each node stores valid tokens for the next generation step. During inference, the vocabulary at each generation step is constrained by the valid continuations in the prefix tree. Due to the relatively short common prefixes among documents, the LLM terminates generation once it has produced sufficient tokens to uniquely identify the current document and concatenates the full document to the context. This results in document title and passage generation processes represented as $P(\hat{t}|q; \theta; \mathcal{T})$ and $P(\hat{p}|q, \hat{t}; \theta; \mathcal{T})$. This mechanism ensures that generated passages align with existing corpus content.

3.3 Reranking: Assess the Relevance

Reranking serves as a second-pass filter to precisely sort the retrieved passages based on the relevance to the query. We implement a self-assessment mechanism that leverages the Self-Retrieval model itself to evaluate the relevance of generated passages. Specifically, Self-Retrieval assesses the passage relevance by generating responses such as "can answer the query" for relevant passages and "cannot answer the query" for irrelevant ones. This self-assessment mechanism allows the model to generate passages and evaluate their relevance within a single inference turn.

During training, we utilize the gold passage from the supervision data as the positive instance, while sampling negative instances from both the same and different documents. This training strategy conditions the LLM to accurately discern and verify the relevance of its outputs, thereby enhancing its autonomous relevance assessment capabilities and improving the overall precision of the retrieval process.

During inference, the overall relevance score $\mathcal{S}$ is composed of the document title score $\mathcal{S}^T$ and the self-assessment score $\mathcal{S}^P$. Specifically, the document title score is derived from the title generation probability, while the self-assessment score is calculated based on the probability of the language model rejecting the passage. Formally, for a set of generated titles and passages $\{(t_1, p_1), (t_2, p_2), \ldots, (t_n, p_n)\}$, the title score for each (t_i, p_i) is given by:

$$\mathcal{S}_i^T = \text{Softmax}(P(t_i|q;\theta)/\tau) \tag{1}$$

and the assessment score is:

$$\mathcal{S}_i^P = \text{Softmax}((1 - P(\text{rejection response}|q, t_i, p_i;\theta))/\delta) \tag{2}$$

where τ and δ are temperature parameters used to scale the logits. Based on preliminary experiments on the development set, we simply set $\tau = \delta = 0.4$ for the main passage retrieval experiments.

The final relevance score is computed as the product of these two components:

$$\mathcal{S} = \mathcal{S}^T \cdot \mathcal{S}^P \tag{3}$$

This combined score is then used to rerank the passage set, producing a more refined ordering based on relevance.

3.4 Training & Inference

Training Self-Retrieval unifies the three distinct tasks of information retrieval – indexing, retrieval, and reranking – into text generation tasks, trained using cross-entropy loss in an auto-regressive manner. Specifically, Self-Retrieval first internalizes the corpus into its parameters through self-supervised learning as introduced in Section 3.1. Subsequently, in addition to a portion of self-supervised instances, it incorporates two different types of data to build retrieval and reranking abilities:

- **Retrieval data:** Utilizes supervised query-passage pairs from the dataset, where the model learns to generate both document titles and passage content in response to input queries.
- **Reranking data:** Employs positive and negative examples to train the model in relevance assessment between queries and passages.

This auto-regressive training approach enables Self-Retrieval to integrate traditionally separate IR components into a unified language model, establishing an end-to-end IR system.

Furthermore, leveraging the universal language generation capabilities of LLMs, we can seamlessly integrate downstream task components, such as readers in RAG, into Self-Retrieval. This integration can be achieved by simply appending the golden answer after the assessment in Self-Retrieval. Consequently, the LLM can function as a comprehensive RAG system, effectively reducing the knowledge gap between IR system and reader modules.

Inference During inference, given an input query, Self-Retrieval aims to obtain the relevant passages that are sorted based on the relevance to query. Firstly, the model generates i document titles through constrained beam search. Secondly, for each title, it generates j passages using beam search. Finally, the resulting $i \times j$ passages are scored using the self-assessment mechanism and reranked to produce the final output.

4 Experimental Results

4.1 Experimental Setup

Datasets and metrics We conduct main experiments on Natural Questions (NQ) [21] and TriviaQA [18] datasets, both of which are widely used retrieval benchmarks based on Wikipedia. We use their versions from the KILT benchmark [34], which consolidates these datasets into a single pre-processed Wikipedia dump, facilitating easier evaluation. Since the KILT test set is not publicly accessible, we use the development set for testing and randomly sample 2,000 instances from the training set as our development set. For our experiments, we sample approximately 40K documents

Model	Params	NQ			TriviaQA		
		H@1	H@5	M@5	H@1	H@5	M@5
Sparse Retrieval							
BM25 [37]	-	14.54	32.71	21.13	20.09	42.73	28.35
Dense Retrieval							
DPR [19]	110M	40.41	61.79	48.80	35.57	57.39	43.93
DPR-FT [19]	110M	42.21	60.45	49.33	36.58	53.05	42.91
BGE [50]	335M	36.30	66.95	48.05	46.97	70.14	55.95
BGE-FT [50]	335M	53.42	80.15	63.99	52.70	75.22	61.65
BGE-FT + BGE-Reranker-FT	770M	52.15	76.15	61.37	44.87	67.39	53.39
GTR-XL [32]	1.24B	37.64	66.84	48.94	35.97	63.75	46.67
GTR-XL + BGE-Reranker-FT	1.57B	57.50	78.92	66.06	58.56	77.65	66.22
GTR-XXL [32]	4.86B	39.21	69.72	50.88	35.97	64.15	46.83
text-embedding-ada-002	-	34.28	62.28	44.64	35.09	62.00	45.15
GritLM [29]	7.24B	44.67	76.00	57.03	39.91	69.34	51.14
GritLM + BGE-Reranker-FT	7.57B	57.57	**81.35**	66.98	58.60	80.54	67.21
Generative retrieval							
DSI-XL [42]	2.85B	43.03	60.26	49.47	29.64	46.74	36.12
DSI-XXL [42]	11.3B	43.81	60.45	50.20	30.55	46.67	36.56
SEAL [5]	406M	36.79	61.35	45.88	36.88	61.66	46.29
DSI-QG [59]	2.85B	34.88	56.60	43.33	29.15	45.53	35.20
NCI + BGE-Reranker-FT	1.07B	50.86	70.27	58.53	28.42	42.18	33.62
Self-Retrieval (StableLM)	2.8B	62.16*	79.28	69.45*	58.69*	78.39*	66.72*
Self-Retrieval (Llama 2)	6.74B	**63.44***	79.29	**70.00***	**59.94***	**81.06***	**68.74***

Table 1: The experimental results of passage retrieval on NQ and TriviaQA test set. * indicates statistically significant improvements ($p < 0.01$) over state-of-the-art retrieval baselines.

from Wikipedia for each dataset. Each document is segmented into passages of maximum 200 words, yielding approximately 1 million passages in total. The detailed statistics of the datasets are presented in Appendix A. We use passage-level Hits@{1, 5} and Mean Reciprocal Rank (MRR)@5 as evaluation metrics.

To comprehensively compare with other generative information retrieval methods, we also conduct experiments on document retrieval. Following NCI [49], we conduct experiments on NQ320K and utilize Recall@{1, 10} and MRR@100 as the evaluation metrics. To evaluate the model's robustness in non-Wikipedia scenarios where high-quality text and titles are not available, we conduct experiments on a subset of MS MARCO [3] following the experimental setup of Ultron [55]. The performance was measured using Recall@{1,5} and MRR@10.

Implementation details In this study, we employ StableLM-3B [44] and Llama2-7B [43] as passage retrieval backbones. For document retrieval, we employ StableLM-1.6B [4] for NQ320K and StableLM-3B for MS MARCO. We train the models using ZeRO stage-2 optimization on 8 NVIDIA A100 (80 GB) GPUs with the AdamW optimizer, a batch size of 16 per GPU, and BFloat16 precision. The models are trained for 3 epochs with a learning rate of 2e-5. During inference, we use beam search to generate 5 titles and 10 passages for each title, with hyperparameters τ and δ set to 0.4 across all models and datasets.

Baselines We evaluate Self-Retrieval models for both passage retrieval and document retrieval, comparing them with sparse, dense, and generative retrieval baselines. The **sparse retrieval** baselines are:BM25 [37] and DocT5Query [28]. The **dense retrieval** baselines include: DPR [19], Sentence-T5 [31], GTR [32], BGE [50], text-embedding-ada-002 [30], GritLM [29], and their fine-tuned variants, DPR-FT and BGE-FT. The **generative retrieval** baselines comprise: DSI [42], DSI-QG [59], NCI [49], Ultron [55], DynamicRetriever [54], GenRet [39], and SEAL [5]. Additionally, to ensure a comprehensive comparison, we also evaluate combinations of strong retrieval baselines with various rerankers, including BGE-Reranker, BGE-Reranker-FT, and RankGPT [41]. In the passage retrieval task, we use the official pre-trained models for all non-fine-tuned dense retrieval baselines. For fine-tuned dense models and generative models, we use their official implementations to replicate the experiments on our dataset. In the document retrieval task, we report the baseline

performances from their original paper. For comprehensive details about these baselines, please refer to Appendix B.

4.2 Main Results

Passage retrieval In Table 1, we compare the performance of Self-Retrieval with various baselines on the NQ and TriviaQA datasets. Self-Retrieval 3B outperforms both strong pre-trained dense retrieval models, such as BGE and GritLM 7B, and other generative retrieval methods. Specifically, Self-Retrieval 3B achieves improvements of 5.46 and 5.07 in MRR@5 over the fine-tuned BGE on NQ and TriviaQA datasets, respectively.

Our results indicate that other generative retrieval baselines exhibit suboptimal performance on passage retrieval. Even the largest DSI-XXL model only achieves an MRR@5 of 50.20 on NQ, significantly lagging behind dense retrieval methods such as GritLM, which achieves an MRR@5 of 57.03. In contrast, our Self-Retrieval model demonstrates strong performance in passage retrieval, achieving an MRR@5 of 69.45, significantly outperforming all other generative methods.

We further compare Self-Retrieval with conventional 2-stage retriever-reranker pipeline. Representative results are shown in Table 1, while the complete experimental results are provided in Appendix D. Notably, even strong retrieval baselines (BGE-FT, GTR-XL, GritLM, and DSI-XL) enhanced with powerful rerankers (such as BGE-Reranker-FT) still fall short of Self-Retrieval's performance, highlighting the advantages of unifying multiple retrieval processes into a single framework rather than treating them as separate components.

These findings underscore the efficacy of Self-Retrieval in harnessing the memory, generation, and ranking capabilities of LLMs, thereby excelling in passage retrieval tasks where other generative baselines struggle.

Method	R@1	R@10	M@100
Sparse Retrieval			
BM25 [37]	29.7	60.3	40.2
DocT5Query [28]	38.0	69.3	48.9
Dense Retrieval			
DPR [19]	50.2	77.7	59.9
Sentence-T5 [31]	53.6	83.0	64.1
GTR-Base [32]	56.0	84.4	66.2
Generative Retrieval			
DSI [42]	55.2	67.4	59.6
SEAL [5]	59.9	81.2	67.7
DSI-QG [59]	63.1	80.7	69.5
NCI [49]	66.4	85.7	73.6
GenRet [39]	<u>68.1</u>	<u>88.8</u>	<u>75.9</u>
Self-Retrieval	**73.3**	**92.6**	**80.7**

Table 2: The experimental result of document retrieval on NQ320K.

Method	R@1	R@5	M@10
Sparse Retrieval			
BM25 [37]	18.9	42.8	29.2
DocT5Query [28]	23.3	49.4	34.8
Dense Retrieval			
DPR [19]	29.1	62.8	43.4
Sentence-T5 [31]	27.3	58.9	40.7
Generative Retrieval			
DSI-Atomic [42]	32.5	63.0	44.3
DynamicRetriever [54]	29.0	64.2	42.5
Ultron-URL [55]	29.6	56.4	40.0
Ultron-PQ [55]	31.6	64.0	45.3
Ultron-Atomic [55]	32.8	64.9	46.9
GenRet [39]	**47.9**	-	**58.1**
Self-Retrieval	<u>47.8</u>	69.9	<u>57.2</u>

Table 3: The experimental result of document retrieval on MS MARCO.

Document retrieval We present the document retrieval results on NQ320K dataset in Table 2. Self-Retrieval outperforms all other generative retrieval methods and dense retrieval baselines across all three metrics. Compared to GenRet, the previously strongest generative retrieval method, Self-Retrieval improves Hits@1 by 5.2, Hits@10 by 3.8, and MRR@100 by 4.8 points. Notably, while other methods commonly employ query generation to augment their training data, Self-Retrieval achieves these results using only the original training set.

To evaluate the effectiveness of Self-Retrieval in non-Wikipedia scenarios, we extend our experiments to MS MARCO. To address the absence of document titles in MS MARCO, we employ Llama2 to automatically generate titles. As shown in Table 3, Self-Retrieval achieves comparable performance to the SOTA model GenRet, while significantly outperforming other baselines. These results demonstrate its adaptability and robustness in non-Wikipedia and title-lacking contexts.

Ablation study To study the effect of each component, we conduct ablation study on both NQ and TriviaQA. Results are presented in Table 4. All components prove crucial for Self-Retrieval's

Method	NQ			TriviaQA		
	H@1	H@5	M@5	H@1	H@5	M@5
Self-Retrieval (base)	62.16	79.28	69.45	58.69	78.39	66.72
w/o indexing	53.05	67.16	58.95	54.45	70.64	60.98
w/o title	47.81	60.90	52.81	52.32	67.91	58.48
w/o self-assessment	54.80	75.21	62.77	46.67	70.79	55.92

Table 4: Ablation study on NQ and TriviaQA.

performance, with each ablation resulting in substantial performance degradation. Specifically, removing the indexing mechanism restricts the model to internalizing only the documents encountered during training, leading to poor performance on unseen passages. Without titles, we directly generate passages with constrained decoding. The absence of document titles significantly degrades performance, as titles provide critical global information that guides the LLM in generating relevant content. Furthermore, removing the self-assessment mechanism leads to a significant decrease in both datasets. Without self-assessment, the model cannot effectively evaluate and refine its initial retrieved passages, leading to less accurate document rankings. This degradation directly impacts downstream applications such as RAG, where precise passage ranking is crucial for generating high-quality responses. These ablation results show that each component of Self-Retrieval addresses a specific challenge in the retrieval process, contributing to its overall effectiveness.

4.3 Performance on Retrieval-Augmented Generation

	NQ		TriviaQA	
	10K	40K	10K	40K
BGE-FT + StableLM-FT	43.18	41.24	56.79	58.15
Self-Retrieval 3B	44.62	46.11	64.03	62.69
BGE-FT + Llama2-FT	49.10	49.24	61.79	61.72
Self-Retrieval 7B	53.26	52.98	72.14	70.40

Table 5: The performance on retrieval-augmented generation. For baseline, we use BGE-FT as the retriever and a fine-tuned LLM as reader. Results are reported using Exact Match (EM) scores.

The end-to-end architecture of Self-Retrieval seamlessly integrates retrieval and answer generation into a single inference process. To evaluate its effectiveness in RAG, we compare Self-Retrieval models with a strong baseline that combines BGE-FT for retrieval and fine-tuned versions of StableLM-3B and LLaMA2-7B as readers. We conduct experiments on subsets of NQ and TriviaQA using 10K and 40K documents for each dataset. We utilize the top-1 retrieved passage as the context and measure performance using the Exact Match (EM) metric. As shown in Table 5, Self-Retrieval significantly outperforms the baseline on both datasets across different model scales. Unlike other RAG pipelines that separate retrieval and generation, Self-Retrieval integrates the entire process within the LLM framework, enabling more accurate and coherent responses through end-to-end modeling.

4.4 Detailed Analysis

Scaling model capacity To explore the impact of model scale on retrieval performance, we evaluate Self-Retrieval with various backbone models of different sizes, including StableLM (1.6B, 3B) [4, 44], Llama2 (7B, 13B) [43], and Qwen-1.5 (4B, 7B, 14B) [2]. Figure 2 presents the results on NQ, showing that Self-Retrieval's retrieval performance benefits from the general capabilities of larger language models. For models within the same series, as the model size increases, we observe consistent improvements in both Hits@1 and Hits@5, indicating strong scaling properties of the Self-Retrieval architecture.

Scaling corpus size Recent studies [35, 53] have demonstrated that generative retrieval methods such as DSI or NCI experience more significant performance degradation compared to dense retrieval methods when scaled to larger corpora. To explore the impact of corpus size on Self-Retrieval,

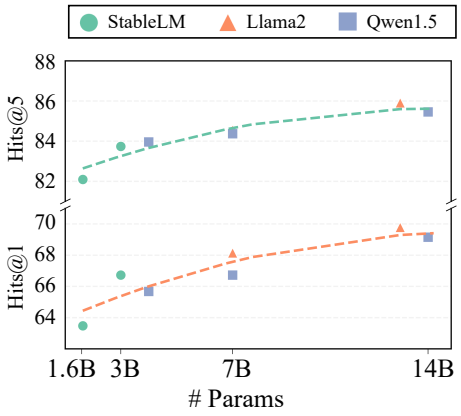

Figure 2: Impact of model capacity on Self-Retrieval performance.

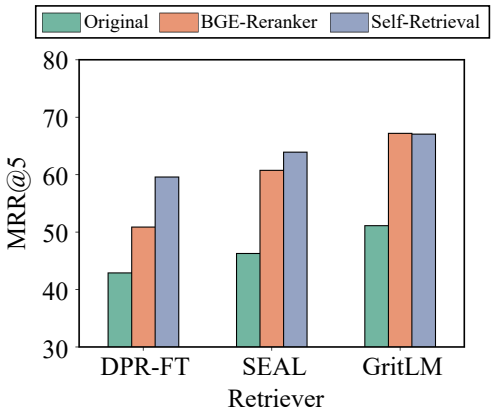

Figure 3: Reranking performance comparison when processing top-100 passages.

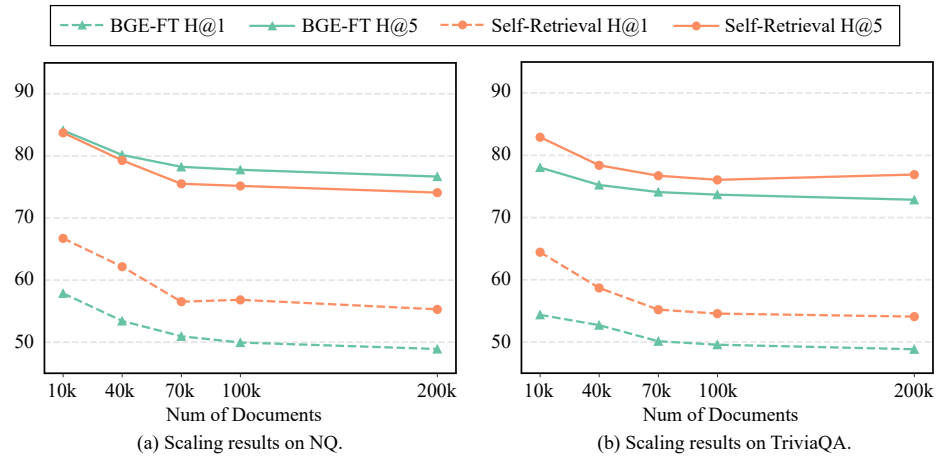

Figure 4: Scalability analysis of retrieval performance for Self-Retrieval and BGE-FT across varying corpus sizes.

we expand our experiments from 10K to 200K documents, scaling the number of passages from 290K to 3M. Figure 4 illustrates the performance trends of BGE-FT and our Self-Retrieval 3B model on the NQ and TriviaQA datasets with increasing corpus sizes. While both models show performance decrease with larger corpus sizes, Self-Retrieval maintains a degradation rate comparable to BGE-FT. As the number of documents continues to increase, the degradation rate gradually diminishes, demonstrating Self-Retrieval's potential scalability to larger document collections. This observation indicates that Self-Retrieval effectively addresses some of the inherent limitations of generative retrieval approaches in large-scale scenarios.

Analysis on reranking In this part, we conduct an in-depth analysis of the reranking performance of Self-Retrieval reranker module in comparison with the fine-tuned BGE-Reranker. We employ DPR-FT, SEAL and GritLM to retrieve 100 passages on TriviaQA, followed by reranking the retrieved results using both approaches. We evaluate performance using MRR@5 as the metric. The experimental results are presented in Figure 3. The results reveal two key findings: (1) Reranking plays a crucial role in information retrieval systems, significantly enhancing the ranking performance across all models. (2) The Self-Retrieval reranker consistently outperforms the fine-tuned BGE Reranker in most scenarios, demonstrating its robustness and effectiveness. These findings demonstrate that Self-Retrieval performs effectively both as a complete IR system and as a reranker component.

In Appendix C, we conduct additional experiments with a chunk size of 100 words, demonstrating Self-Retrieval's adaptability to different text segmentation strategies. In Appendix E, we further discuss Self-Retrieval's computational efficiency.

5 Conclusion

In this paper, we propose Self-Retrieval, an end-to-end LLM-driven information retrieval architecture that unifies indexing, retrieval, and reranking in a single LLM. This approach enables the LLM to internalize the corpus, generate relevant content, and perform self-assessment within a unified framework. Unlike previous works that incorporate LLMs into individual IR components, Self-Retrieval provides a unified framework for the entire IR procedure, facilitating knowledge sharing and deep collaboration among different components. Experimental results demonstrate that Self-Retrieval achieves strong performance across various retrieval benchmarks and application scenarios. In future work, we plan to extend our method to further enhance the reliability and trustworthiness of LLM generation.

Limitations

While our experiments demonstrate the effectiveness of Self-Retrieval, several limitations need to be addressed in future work. Our current evaluation is limited to 200K Wikipedia documents and 3M passages, and testing on larger and noisier text collections is needed. As an LLM-driven system, Self-Retrieval has lower retrieval efficiency compared to sparse or dense retrieval methods, which may limit its applications to specialized knowledge systems. Furthermore, enabling incremental learning and dynamic corpus expansion remains an important direction for future research.

Acknowledge

We sincerely thank the reviewers for their insightful comments and valuable suggestions. We are grateful to Le Yu and Xinyu Lu for their helpful feedback on the paper writing. This work was supported by the Natural Science Foundation of China (No. 62122077, 62272439), Beijing Municipal Science and Technology Project (Nos. Z231100010323002), the Basic Research Program of ISCAS (ISCAS-JCZD-202303), and CAS Project for Young Scientists in Basic Research (Grant No.YSBR-040).

References

[1] Qingyao Ai, Ting Bai, Zhao Cao, Yi Chang, Jiawei Chen, Zhumin Chen, Zhiyong Cheng, Shoubin Dong, Zhicheng Dou, Fuli Feng, Shengling Gao, J. Guo, Xiangnan He, Yanyan Lan, Chenliang Li, Yiqun Liu, Ziyu Lyu, Weizhi Ma, Jun Ma, Zhaochun Ren, Pengjie Ren, Zhiqiang Wang, Min Wang, Jirong Wen, Lei Wu, Xin Xin, Jun Xu, Dawei Yin, Peng Zhang, Fan Zhang, Wei na Zhang, M. Zhang, and Xiaofei Zhu. Information retrieval meets large language models: A strategic report from chinese ir community. *ArXiv*, abs/2307.09751, 2023.

[2] 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. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[3] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*, 2016.

[4] Marco Bellagente, Jonathan Tow, Dakota Mahan, Duy Phung, Maksym Zhuravinskyi, Reshinth Adithyan, James Baicoianu, Ben Brooks, Nathan Cooper, Ashish Datta, Meng Lee,

Emad Mostaque, Michael Pieler, Nikhil Pinnaparju, Paulo Rocha, Harry Saini, Hannah Teufel, Niccolo Zanichelli, and Carlos Riquelme. Stable lm 2 1.6b technical report, 2024.

[5] Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. Autoregressive search engines: Generating substrings as document identifiers. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 31668–31683. Curran Associates, Inc., 2022.

[6] Luiz Henrique Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. Inpars: Data augmentation for information retrieval using large language models. *ArXiv*, abs/2202.05144, 2022.

[7] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. Autoregressive entity retrieval. *ArXiv*, abs/2010.00904, 2020.

[8] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. Autoregressive entity retrieval. In *International Conference on Learning Representations*, 2021.

[9] Jiangui Chen, Ruqing Zhang, J. Guo, Y. Liu, Yixing Fan, and Xueqi Cheng. Corpusbrain: Pretrain a generative retrieval model for knowledge-intensive language tasks. *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022.

[10] Pinzhen Chen, Nikolay Bogoychev, Kenneth Heafield, and Faheem Kirefu. Parallel sentence mining by constrained decoding. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1672–1678, Online, July 2020. Association for Computational Linguistics.

[11] David R. Cheriton. From doc2query to doctttttquery. 2019.

[12] Fernando Ferraretto, Thiago Laitz, Roberto de Alencar Lotufo, and Rodrigo Nogueira. Exaranker: Synthetic explanations improve neural rankers. *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023.

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

[14] Jiafeng Guo, Changjiang Zhou, Ruqing Zhang, Jiangui Chen, Maarten de Rijke, Yixing Fan, and Xueqi Cheng. Corpusbrain++: A continual generative pre-training framework for knowledge-intensive language tasks. *ArXiv*, abs/2402.16767, 2024.

[15] Chao-Wei Huang, Chen-Yu Hsu, Tsung-Yuan Hsu, Chen-An Li, and Yun-Nung (Vivian) Chen. Converser: Few-shot conversational dense retrieval with synthetic data generation. *ArXiv*, abs/2309.06748, 2023.

[16] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43, 2023.

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

[18] Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada, 2017. Association for Computational Linguistics.

[19] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online, November 2020. Association for Computational Linguistics.

[20] Varsha Kishore, Chao gang Wan, Justin Lovelace, Yoav Artzi, and Kilian Q. Weinberger. Incdsi: Incrementally updatable document retrieval. *ArXiv*, abs/2307.10323, 2023.

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

[22] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc., 2020.

[23] Xiaoxi Li, Yujia Zhou, and Zhicheng Dou. Unigen: A unified generative framework for retrieval and question answering with large language models. *ArXiv*, abs/2312.11036, 2023.

[24] Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2795–2806, Online, August 2021. Association for Computational Linguistics.

[25] Guangyuan Ma, Xing Wu, Peng Wang, Zijia Lin, and Songlin Hu. Pre-training with large language model-based document expansion for dense passage retrieval. *ArXiv*, abs/2308.08285, 2023.

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

[27] Xueguang Ma, Xinyu Crystina Zhang, Ronak Pradeep, and Jimmy J. Lin. Zero-shot listwise document reranking with a large language model. *ArXiv*, abs/2305.02156, 2023.

[28] Antonio Mallia, O. Khattab, Nicola Tonellotto, and Torsten Suel. Learning passage impacts for inverted indexes. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.

[29] Niklas Muennighoff, Hongjin Su, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. Generative representational instruction tuning, 2024.

[30] Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, et al. Text and code embeddings by contrastive pre-training. *arXiv preprint arXiv:2201.10005*, 2022.

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

[32] Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, and Yinfei Yang. Large dual encoders are generalizable retrievers. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9844–9855, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[33] Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. From doc2query to doctttttquery. *Online preprint*, 6:2, 2019.

[34] Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online, June 2021. Association for Computational Linguistics.

[35] Ronak Pradeep, Kai Hui, Jai Gupta, Adam Lelkes, Honglei Zhuang, Jimmy Lin, Donald Metzler, and Vinh Tran. How does generative retrieval scale to millions of passages? In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1305–1321, Singapore, December 2023. Association for Computational Linguistics.

[36] Shanbao Qiao, Xuebing Liu, and Seung-Hoon Na. Diffusionret: Diffusion-enhanced generative retriever using constrained decoding. In *Conference on Empirical Methods in Natural Language Processing*, 2023.

[37] Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

[38] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3715–3734, Seattle, United States, July 2022. Association for Computational Linguistics.

[39] Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten de Rijke, and Zhaochun Ren. Learning to tokenize for generative retrieval. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[40] Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. Is chatgpt good at search? investigating large language models as re-ranking agent. *ArXiv*, abs/2304.09542, 2023.

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

[42] Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, Tal Schuster, William W Cohen, and Donald Metzler. Transformer memory as a differentiable search index. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 21831–21843. Curran Associates, Inc., 2022.

[43] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[44] Jonathan Tow, Marco Bellagente, Dakota Mahan, and Carlos Riquelme. Stablelm 3b 4e1t.

[45] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. SimLM: Pre-training with representation bottleneck for dense passage retrieval. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2244–2258, Toronto, Canada, July 2023. Association for Computational Linguistics.

[46] Liang Wang, Nan Yang, and Furu Wei. Query2doc: Query expansion with large language models. In *Conference on Empirical Methods in Natural Language Processing*, 2023.

[47] Liang Wang, Nan Yang, and Furu Wei. Query2doc: Query expansion with large language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9423, Singapore, December 2023. Association for Computational Linguistics.

[48] Yujing Wang, Ying Hou, Hong Wang, Ziming Miao, Shibin Wu, Hao Sun, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, Xing Xie, Hao Sun, Weiwei Deng, Qi Zhang, and Mao Yang. A neural corpus indexer for document retrieval. *ArXiv*, abs/2206.02743, 2022.

[49] Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, Xing Xie, Hao Sun, Weiwei Deng, Qi Zhang, and Mao Yang. A neural corpus indexer for document retrieval. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 25600–25614. Curran Associates, Inc., 2022.

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

[51] Tianchi Yang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, and Qi Zhang. Auto search indexer for end-to-end document retrieval. In *Conference on Empirical Methods in Natural Language Processing*, 2023.

[52] Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. Generate rather than retrieve: Large language models are strong context generators. In *International Conference for Learning Representation (ICLR)*, 2023.

[53] Peiwen Yuan, Xinglin Wang, Shaoxiong Feng, Boyuan Pan, Yiwei Li, Heda Wang, Xupeng Miao, and Kan Li. Generative dense retrieval: Memory can be a burden. *arXiv preprint arXiv:2401.10487*, 2024.

[54] Yujia Zhou, Jing Yao, Zhicheng Dou, Ledell Wu, and Ji-Rong Wen. Dynamicretriever: A pre-training model-based ir system with neither sparse nor dense index. *arXiv preprint arXiv:2203.00537*, 2022.

[55] Yujia Zhou, Jing Yao, Zhicheng Dou, Ledell Wu, Peitian Zhang, and Ji-Rong Wen. Ultron: An ultimate retriever on corpus with a model-based indexer. *arXiv preprint arXiv:2208.09257*, 2022.

[56] Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zhicheng Dou, and Ji-Rong Wen. Large language models for information retrieval: A survey, 2024.

[57] Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Zhicheng Dou, and Ji rong Wen. Large language models for information retrieval: A survey. *ArXiv*, abs/2308.07107, 2023.

[58] Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. Rankt5: Fine-tuning t5 for text ranking with ranking losses. *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2022.

[59] Shengyao Zhuang, Houxing Ren, Linjun Shou, Jian Pei, Ming Gong, Guido Zuccon, and Daxin Jiang. Bridging the gap between indexing and retrieval for differentiable search index with query generation. *arXiv preprint arXiv:2206.10128*, 2022.

A Dataset Statistics

Table 6 presents the statistics of the NQ and TriviaQA datasets used in our experiments.

Dataset	Natural Questions		TriviaQA	
	10K	40K	10K	40K
# doc	10,000	37,202	10,000	38,399
# psg	291,506	979,804	390,586	1,193,047
# train	32,163	72.716	29,038	51,166
# dev	2,000	2,000	2,000	2,000
# test	2,837	2,837	5,355	5,355

Table 6: Statistics of the experimental datasets. #doc/#psg denotes number of documents/passages; #train/#dev/#test denotes size of training/development/test set. Training instances without query-document pairs are removed.

B Baselines

The sparse retrieval baselines are as follows:

- **BM25** [37] is a classical sparse retrieval algorithm based on probabilistic relevance framework and term frequency statistics.
- **DocT5Query** [28] expands documents by generating potential queries using a fine-tuned T5 model.

The dense retrieval baselines are as follows:

- **DPR** [19] is a dual-encoder model trained with in-batch negative sampling. We fine-tune DPR on our training datasets to obtain **DPR-FT**, following the official implementation and hyperparameter settings.
- **BGE** [50] is a state-of-the-art universal embedding model trained on approximately 200 million text pairs using contrastive learning. We employ the bge-large-en-v1.5 variant and fine-tune it on our training datasets to obtain **BGE-FT**. The fine-tuning process uses a learning rate of 1e-5, batch size of 128, and runs for 10 epochs.
- **Sentence-T5** [31] employs a dual-encoder T5 architecture to generate semantic embeddings through contrastive learning for efficient retrieval.
- **GTR-XL** [32] is a dense retrieval model based on Sentence-T5, pre-trained on billions of question-answer pairs.
- **Text-embedding-ada-002** is a powerful embedding model developed by OpenAI, accessible through their API service.
- **GritLM** [29] is built upon the Mistral 7B language model and optimized using both embedding and generation objectives.

The generative retrieval baselines are as follows:

- **DSI** [42] is a sequence-to-sequence model that directly maps queries to document identifiers.
- **DSI-QG** [59] enhances the DSI framework by incorporating a doc2query model for dataset augmentation.
- **SEAL** [5] utilizes n-gram as the document identifiers and constrains the generation process using FM-index.
- **NCI+BGE-Reranker-FT**. NCI [49] employs a sequence-to-sequence architecture with a prefix-aware weight-adaptive decoder. We train the model using T5-Large for document-level retrieval following the official implementation. To obtain passage-level results, we further incorporate a fine-tuned BGE reranker (bge-reranker-large).

- **Ultron** [55] represents documents using three types of identifiers (URL, PQ, Atomic) and trains the model through a progressive three-stage pipeline.
- **DynamicRetriever** [54] parameterizes traditional static indices by embedding both token-level and document-level corpus information into a pre-trained model for dynamic document identifier generation.
- **GenRet** [39] employs discrete auto-encoding with progressive training and clustering techniques to learn semantic document identifiers for generative retrieval.

C Ablation on Chunk Size

To investigate the potential impact of chunk size, we conduct additional experiments comparing Self-Retrieval with strong baselines on the NQ dataset using a chunk size of 100 words, complementing our main experiments where chunk size is set to 200. The experimental results are presented in Table 7. It demonstrates that Self-Retrieval significantly outperforms the baselines with both chunk sizes settings, further validating the effectiveness of our proposed method.

Model	Params	Hits@1	Hits@5	MRR@5
BGE-FT	335M	40.79	58.92	47.76
GritLM	7B	30.95	51.36	38.77
Self-Retrieval (StableLM)	3B	**58.43**	**77.76**	**66.18**

Table 7: Retrieval performance with chunk length of 100 words.

D Full Comparison with Retriever-Reranker Pipeline

	NQ			TriviaQA		
	H@1	H@5	M@5	H@1	H@5	M@5
BGE-FT	53.42	80.15	63.99	52.70	75.22	61.65
BGE-FT + BGE-Reranker	21.91	54.58	33.33	45.36	72.16	55.78
BGE-FT + BGE-Reranker-FT	52.15	76.15	61.37	44.87	67.39	53.39
BGE-FT + RankGPT	44.21	73.68	55.51	48.00	72.00	57.33
GTR-XL	37.64	66.84	48.94	35.97	63.75	46.67
GTR-XL + BGE-Reranker	26.39	59.96	38.50	42.41	68.42	52.51
GTR-XL + BGE-Reranker-FT	57.50	78.92	66.06	58.56	77.65	66.22
GTR-XL + RankGPT	42.11	68.42	52.30	47.00	66.00	54.95
GritLM	44.67	76.00	57.03	39.91	69.34	51.14
GritLM + BGE-Reranker	30.06	65.87	43.20	43.64	70.87	54.23
GritLM + BGE-Reranker-FT	57.57	**81.35**	66.98	58.60	80.54	67.21
GritLM + RankGPT	37.89	70.53	51.19	44.00	66.00	52.70
DSI-XL	43.03	60.26	49.47	29.64	46.74	36.12
DSI-XL + BGE-Reranker	34.39	64.26	45.74	37.85	52.57	43.49
DSI-XL + BGE-Reranker-FT	50.02	68.60	57.43	36.49	52.40	42.36
DSI-XL + RankGPT	49.47	73.68	59.25	39.00	52.00	44.75
Self-Retrieval (StableLM)	62.16	79.28	69.45	58.69	78.39	66.72
Self-Retrieval (Llama 2)	**63.44**	79.29	**70.00**	**59.94**	**81.06**	**68.74**

Table 8: Comparison between Self-Retrieval and traditional two-stage retriever-reranker pipelines.

We comprehensively evaluate Self-Retrieval against various two-stage retriever-reranker pipelines. Specifically, we construct these pipelines using state-of-the-art retrievers (BGE, GTR, GritLM, and DSI-XL) combined with three different reranking approaches: BGE reranker, fine-tuned BGE reranker, and RankGPT. As shown in Table 8, Self-Retrieval achieves superior performance compared to most retriever-reranker combinations, demonstrating the effectiveness of our end-to-end approach over traditional pipeline methods.

E Efficiency Analysis

We conduct efficiency analysis on NQ dataset using an NVIDIA A100-80G GPU. Results in Table 9 illustrate that, while Self-Retrieval requires slightly higher computational resources than DSI, it provides notable performance benefits. Notably, Self-Retrieval with a beam size of 10 achieves significantly higher H@5 scores compared to DSI-XL with a beam size of 100, enabling a flexible trade-off between retrieval quality and computational efficiency. When compared to SEAL, which also employs natural language decoding, Self-Retrieval demonstrates more efficient memory usage (30MB vs 444MB) by utilizing a lightweight trie structure instead of SEAL's resource-intensive FM-Index post-processing mechanism. Furthermore, the efficiency of Self-Retrieval stands to benefit from ongoing developments in optimization techniques (e.g., quantization and attention acceleration) and hardware advancements.

Model Name	Memory	Beam Size	Latency (s)	Hits@5
SEAL	444MB	10	1.18	61.91
		100	5.92	59.57
DSI-XL	0	10	0.23	60.21
		100	0.45	60.21
Self-Retrieval	30MB	10	1.44	76.17
		100	6.06	81.49

Table 9: Efficiency analysis.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have clearly explained the shortcomings of previous works in the abstract and introduction (Section 1), as well as how we can make improvements. We have validated these claims in the experiment section (Section 4).

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have specifically written a chapter to illustrate the limitations (Section Limitations) of our method and the factors that may affect its performance.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: Our work mainly focuses on the application of large language models and does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide a comprehensive description of the training and inference details for our proposed method in Section 3.4. In the experimental setup section (Section 4.1), we elaborate on the datasets used and the model training process. Additionally, we have made the complete training code available in an open-source repository, ensuring that other researchers can replicate our results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility.

In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We have submitted the code and instructions in an open-source github repository, and can obtain the experimental results in the paper by running our code.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so No is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide a detailed description of the training and testing details in Section 4.1, including data partitioning, training hyperparameters, optimizer, and testing methodology.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We have marked the significance analysis with an asterisk in the main experimental table (Table 1).

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We mentioned in Section 4.1 that the experiments were conducted using 8 Nvidia A100 80GB GPUs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We obey relevant rules in NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: We talk about positive impacts in introduction and method section, and possible negative impacts in limitation part.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Our experiments use publicly available base models and standard training datasets.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We properly cite all datasets and models used in our experiments..

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: All datasets are from existing public resources, and no new data assets are created.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No crowd-sourcing or human annotation is required in our experiments.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our research does not involve human participants or human subjects research, therefore no IRB approval or risk assessment was required.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.