RETRIEVAL AS REASONING: LEARNING TO SELECT AND GENERATE WITH LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) has become a practical solution for addressing hallucination in large language models (LLMs) by conditioning responses on retrieved documents. However, existing RAG systems face two major limitations: (1) retrieval objectives are often misaligned with the downstream generation task, leading to irrelevant documents harmful to the generation; (2) concatenating many retrieved documents into long prompts strains model capacity and introduces positional biases that degrade performance. To overcome these issues, we propose a unified framework where the LLM itself learns to perform document selection and answer generation in an end-to-end manner. Inspired by human reasoning, our model organizes documents via hierarchical semantic IDs and selects relevant content through a self-reflection mechanism composed of query-specific attention and an additional feed-forward MLP layer. This architecture enables the model to promote helpful documents directly during generation, eliminating the need for separate retrievers or rerankers. Through joint training, the model learns to select the most informative 2-3 documents. We conduct experiments to validate the effectiveness of our design.

1 INTRODUCTION

Large language models (LLMs) have demonstrated remarkable capabilities across a wide range of natural language processing tasks, including open-ended conversation (Wang et al., 2024a; Liu et al., 2024; Xi et al., 2025), problem solving (Cobbe et al., 2021; Wei et al., 2022; Lewkowycz et al., 2022), code generation (Chen et al., 2021; Austin et al., 2021; Li et al., 2022). Their success is largely attributed to the scale of their architectures and the massive datasets used during pretraining, which allow them to encode vast amounts of linguistic and factual knowledge. However, despite these strengths, LLMs are inherently limited by the static nature of their pretraining corpus. When presented with complex, ambiguous, or unfamiliar queries, especially those requiring recent or specialized knowledge, they often generate inaccurate or fabricated responses, a behavior commonly referred to as hallucination. This issue becomes even more pronounced in domains where information evolves rapidly or factual accuracy is essential.

To address the hallucination problem in large language models (LLMs), Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) has emerged as a practical solution. Rather than relying solely on static knowledge stored in model parameters, RAG enables LLMs to retrieve and condition on external documents at inference time, grounding their responses in up-to-date and domain-specific evidence. Despite its effectiveness, it have several intrinsic drawback.

(1) The performance of retrieval-augmented generation (RAG) relies critically on the quality of the retrieved documents, yet the retriever's objective is often misaligned with the goal of generation. Most retrievers are trained to maximize semantic similarity between a query and candidate documents, typically through embedding-based scoring (Karpukhin et al., 2020). However, such similarity does not guarantee that the document contains factual or contextually useful information for answering the query. One may expect these semantically similar yet uninformative documents to be harmless. Surprisingly, Cuconasu et al. (2024) show that such documents can actively degrade performance, sometimes performing worse than inserting random documents into the prompt. For instance, when answering "Who won the first Nobel Prize in Physics", a misleading document about Einstein may be more harmful than a random one (Jin et al., 2024; Wang et al., 2024b). These find-

054 055 056 057 058 059 060 061 062 063 064 065 066 067 068 069 070 071 072 073 074 075 076 077 078 079 080 081 082 083 084 085 086 087 088 089 090 091 092 093 094 095 096 097 098 099 100 101 102 103 104 105 106 107

ings motivate the exploration of retriever metrics that are more directly aligned with the generation objective.

(2) RAG systems depend on the long-context processing ability of large language models, as multiple retrieved documents are often concatenated into a single input. This not only increases computational overhead but also weakens the model's ability to leverage the retrieved information. As shown by Liu et al. (2023), even models explicitly trained for long-context exhibit performance degradation when relevant information appears in the middle of the input, rather than at the beginning or end. Such positional sensitivity undermines the effectiveness of retrieval and necessitates document reranking. However, as with retrieval, designing reranking objectives that are well aligned with the generation task remains challenging. Moreover, introducing additional components such as rerankers complicates the training pipeline and can lead to training instability.

To address these limitations, we adopt an intuitive strategy: allow the language model itself to decide which documents are most useful for answering a given question. Rather than depending on a retriever with manually crafted heuristics or embedding similarity, we design the model to learn document selection directly from data in a hierarchical manner. This idea is inspired by how humans naturally approach complex questions: they first organize available information into conceptual or topical categories, and then selectively search within those categories for the most pertinent details. By mimicking this behavior, we encourage the model to develop a coarse-to-fine understanding of the corpus, leveraging hierarchical semantic ids, a concept borrowed for generative retrieval (Wang et al., 2022c), to discriminate between document clusters and select those most likely to support accurate generation.

To support this process, we introduce a lightweight self-reflection mechanism that plays a central role in enabling the model to perform effective document selection. This mechanism consists of additional query-specific attention heads and an independent MLP layer, designed to help the model leverage its intrinsic knowledge and internal representations when selecting candidate documents. Built upon this structure, we train the model in an end-to-end manner, allowing it to jointly learn both document selection and answer generation. For each query, the model first identifies a set of candidate documents, each annotated with hierarchical IDs that reflect their semantic or topical structure. It then selectively incorporates the most relevant candidates into the generation process, learning to associate specific hierarchical patterns with successful answer outcomes. Through this training paradigm, the model effectively aligns document selection with downstream generation quality, without relying on external retrievers or manual scoring heuristics.

Unlike traditional RAG pipelines, which decouple retrieval, reranking, and generation into distinct modules, our framework unifies these components within a single model. The language model simultaneously acts as retriever, reranker, and generator, leveraging the same set of internal parameters across all stages (See Figure 1). This design eliminates the need for an explicit reranking step. Through end-to-end fine-tuning, the model learns to promote the most helpful documents to the top of its input sequence, effectively aligning document selection with the downstream generation objective. Meanwhile, documents that are unhelpful or distracting are implicitly filtered out during training, as their lack of contribution to generation quality provides a negative learning signal. This tightly coupled optimization enables the model to perform competitively even when selecting only the top 2-3 documents.

Our contributions can be summarized as follows:

- We propose an end-to-end framework that enables a large language model to jointly perform document selection and answer generation without relying on external retrievers or rerankers. By leveraging hierarchical semantic identifiers inspired by generative retrieval, the model learns a coarse-to-fine understanding of the document corpus and selects evidence that directly supports the generation task.
- To enable large language models to retrieve documents, a capability not acquired during pretraining, we introduce a lightweight self-reflection module composed of query-specific attention heads and an auxiliary MLP layer. This component allows the model to internalize relevance judgments that are synchronized with generation utility, effectively eliminating the misalignment between the retrieval and generation modules.
- Through unified training, the model learns to surface the most helpful documents and discard unhelpful ones, achieving strong performance while conditioning on only the top 2-3 selected inputs.

Experimental results show that our method outperforms traditional RAG methods, demonstrating both efficiency and effectiveness.

2 Related work

Information Retrieval (IR). Information retrieval techniques aim to efficiently obtain, process, and interpret information from large-scale data. Traditional approaches, known as sparse retrieval, enable fast document search through inverted indexing, where each term is mapped to a list of documents containing that term. Relevance is then determined using term-matching metrics such as TF–IDF (Ramos et al.), query likelihood (Lafferty & Zhai, 2001), and BM25 (Robertson et al., 2009). With the development of pre-trained language models (Devlin et al., 2019; Liu et al., 2019), several works have leveraged Transformer-based encoders to generate dense vector representations for both queries and documents, with similarity typically measured using inner product or cosine similarity (Karpukhin et al., 2020; Xiong et al., 2020; Wang et al., 2022b;a). In contrast, generative retrieval (De Cao et al., 2020; Tay et al., 2022; Wang et al., 2022c; Zhou et al., 2022) represents a different paradigm: instead of relying on similarity matching in the embedding space, it takes the query as input and directly generates document identifiers (DocIDs) corresponding to relevant documents. Specifically, each document in the corpus is assigned a unique identifier, and the retrieval model employs constrained beam search to ensure that the generated DocIDs correspond to valid documents within the corpus. More recent works have focused on the retriever model training (Zhou et al., 2023; 2024), the construction of semantic identifiers (Sun et al., 2023; Yang et al., 2023; Askari et al., 2024; Valluri et al., 2024), and continual learning on dynamic corpora (Mehta et al., 2022; Kishore et al., 2023; Guo et al., 2024).

Retrieval Augmented Generation (RAG). Early efforts to integrate retrieval mechanisms for improving text generation quality can be traced back to Chen et al. (2017); Dinan et al. (2018); Weston et al. (2018). In particular, retrieval systems play a crucial role in open-domain question answering, where a two-stage framework is commonly adopted: a context retriever first selects a small subset of passages, some of which may contain the answer to the question, and a generator then identifies the correct answer from these passages (Chen et al., 2017). Subsequent research has focused on improving retrieval quality by employing dense representing vectors (Karpukhin et al., 2020), or combining the masked language model (Devlin et al., 2019) with the retrieval system (Lee et al., 2019; Guu et al., 2020). This line of work was later formalized under the term Retrieval-Augmented Generation (RAG) by Lewis et al. (2020), which generalizes the framework to all sequence-to-sequence models. After large language models with billions of parameters emerged and demonstrated their superior performance in language generation, further studies explored how RAG could be leveraged to strengthen these models (Izacard & Grave, 2020; Borgeaud et al., 2022; Jiang et al., 2022). Numerous studies have proposed methods that focused on different aspects of retrieval-augmented generation, including training of retrievers or generators (Weijia et al., 2023; Izacard et al., 2023; Lin et al., 2023; Li et al., 2024), instruction fine-tuning (Wang et al., 2023), leveraging in-context abilities (Huang et al., 2023; Trivedi et al., 2022; Wang et al., 2024c), adaptive document selection (Jiang et al., 2023; Asai et al., 2024; Yan et al., 2024; Su et al., 2024; Baek et al., 2024; Jeong et al., 2024; Wang et al., 2024b), passage ranking (Yu et al., 2024), context compressing (Xu et al., 2024a), and parametric knowledge injection (Su et al., 2025). In addition, several studies have investigated how to retrieve relational knowledge relevant to a given query from a pre-constructed graph database (Edge et al., 2024; Hu et al., 2024; Mavromatis & Karypis, 2024; Peng et al., 2024).

3 Preliminaries

In this section, we present the framework for reasoning over a document corpus. We consider the standard QA task, where we take in a query $x \in \mathcal{X}$, and an LLM, denoted by $p_\theta(\cdot|x)$, which outputs a conditional probability distribution over the answer space $\mathcal{Y}$ and generates an answer a by sampling from this distribution. Let the QA dataset be denoted by $\mathcal{D}_{\text{QA}} = \{(x_i, a_i)\}_{i=1}^{M}$, we consider the following training loss of log-likelihood, where each (x_i, a_i) is a query-action pair. The standard training objective is to maximize the log-likelihood of the ground-truth answers under the model,

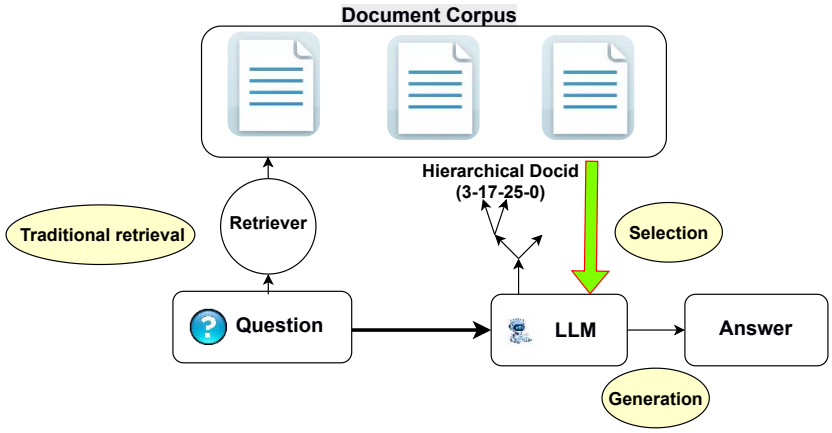

Figure 1: Comparison between the traditional retrieval process and our proposed pipeline. In the traditional setup, the retriever and generator are independent modules: the retriever first ranks documents, and the generator conditions on the top-k results. In contrast, our method enables the LLM to generate hierarchical docids, which directly identify the most relevant document in a semantically structured space.

leading to the loss function

$$L_{\text{QA}}(\phi) = -\sum_{i=1}^{M}\left[\sum_{j=1}^{m}\log p_\theta\big(a_i^j \mid x, a_i^{[1:j-1]}\big)\right], \tag{3.1}$$

where a_i^j is the j-th token of a_i. Although fine-tuning the LLM by minimizing this loss function is straightforward, it may fail to yield improvements when the QA task requires knowledge absent from the pretrained model. In such cases, the model parameters, constrained to remain close to their pretrained values, cannot adequately capture the missing information, leading to persistently high loss and consequently little or no improvement in the quality of the generated answers.

To mitigate this problem, we assume access to an external corpus $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$. We further assume that the knowledge required to answer the question can be found in the document corpus. In the following, we will first review the RALM pipeline, which serves as a baseline framework for incorporating external knowledge into the generation process.

Retrieval augmented generation (RAG). In RAG, a retriever model $g_\phi(\cdot|x)$, parametrized by ϕ, is employed to select a subset of documents $\mathcal{D}^* \subseteq \mathcal{D}$ given the query x. For a detailed survey of retrieval approaches, we refer the reader to Section 2. The retrieved documents $\mathcal{D}^*$ is then provided, together with the query x, as input to the generator p_θ, which produces an answer by sampling from the conditional distribution

$$\widetilde{a} \sim p_\theta(\cdot|x, \mathcal{D}^*).$$

When $\mathcal{D}^*$ contains the supporting evidence, the distribution is expected to generate answers of higher quality compared to the distribution conditioned solely on x.

However, retrieving documents that are genuinely helpful for generation remains challenging. Since most retrieval methods are optimized for semantic similarity rather than generation task, they may return documents that are topically related yet uninformative for answering the query, with their actual contribution to generation becoming apparent only after being processed by the generator. This limitation is exacerbated when fine-tuning the language model, as minimizing the retrieval-augmented QA loss

$$L_{\text{QA}}^{\text{RAG}}(\theta) = -\sum_i \log p_\theta(a_i|x_i, \mathcal{D}^*)$$

depends critically on the quality of the retrieved documents. When performance fails to improve, it becomes unclear whether the bottleneck arises from the quality of the retrieved documents or from

the generator's ability to effectively utilize them during answer generation. While some studies have proposed joint training strategies that simultaneously optimize retriever parameter ϕ and generator parameter θ, the two components maintain fully independent hidden representations, with no parameter sharing or representational alignment. Consequently, even when optimized together, the training process fails to capture or exploit the inherent relatedness between retrieval and generation, thereby limiting the potential gains from joint learning.

4 METHODOLOGY

4.1 HIERARCHICAL SEMANTIC IDS

We apply the idea first proposed in (Wang et al., 2022c) to label the document with hierarchical docids. Specifically, this method assigns each document a structured semantic identifier (docID) that reflects its position within a tree of semantic clusters. Specifically, documents are first embedded into vector representations using a pretrained encoder such as BERT. These embeddings are then clustered using the k-means algorithm. If a cluster contains more than a predefined threshold c of documents, k-means is applied recursively to produce finer-grained clusters. This process continues until every leaf node contains at most c documents.

This hierarchical id exhibits two notable properties that are important for our design. First, documents with longer common prefixes in their semantic identifiers tend to be semantically similar. This means that the identifier structure encodes coarse-to-fine semantic relationships: documents grouped under the same high-level cluster share the initial segments of their identifiers, while finer distinctions emerge in later segments. As a result, the model can leverage this structure to better locate and discriminate between relevant documents based on shared semantics.

Second, the semantic identifiers can be generated in an autoregressive manner. Specifically, given a document identifier $i_d = [i_1, i_2, \ldots, i_m]$, we can apply an autoregressive retriever model $g_\phi(\cdot|x)$ to generate its docids, to predict the sequence of indices one step at a time, conditioned on the query x and the previously predicted indices. To train the model, we minimize a retrieval loss similar to the supervised fine-tuning loss used in question answering (3.1), by minimizing the following retrieval loss:

$$L_{\text{retriever}}(\phi) = -\mathbb{E}\left[\sum_{j=1}^{m} \log g_\phi\big(i_j \mid x, i_{[1:j-1]}\big)\right]. \tag{4.1}$$

The structural similarity between Equation (3.1) and Equation (4.1) motivates our design to unify retrieval and generation into a single language model, streamlining both stages under a shared architecture and training objective.

4.2 UNIFYING RETRIEVAL AND GENERATION WITH A SINGLE LLM

Our approach draws on the parallel between generative retrieval and autoregressive generation in large language models (LLMs). We begin with a standard decoder-only Transformer architecture, where hidden representations are iteratively updated using self-attention and feedforward layers across multiple layers. Each layer processes hidden states $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_T] \in \mathbb{R}^{T\times d}$ through a self-attention mechanism, followed by a gated feedforward network, allowing the model to capture contextual dependencies and non-linear interactions.

In the standard self-attention mechanism, each token in the sequence is projected into query (Q), key (K), and value (V) vectors using learned linear projections, i.e.,

$$\mathbf{Q} = \mathbf{H}\mathbf{W}_Q, \mathbf{K} = \mathbf{H}\mathbf{W}_K, \mathbf{V} = \mathbf{H}\mathbf{W}_V,$$

where $\mathbf{H}_Q, \mathbf{H}_K, \mathbf{H}_V$ are the learned projection parameters. These projections allow the model to compute similarity scores between queries and keys, which are then used to weight the values and produce context-aware representations. This self-attention mechanism, defined as:

$$\text{Attn}(\mathbf{H}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V},$$

serves as the core operation for computing contextualized representations by dynamically weighting token-to-token interactions, thereby enabling the model to capture both local and global dependencies across the sequence.

When extending language models to perform document retrieval, the conventional attention heads, originally optimized for natural language generation, may lack the inductive bias and representational capacity necessary to distinguish useful documents from irrelevant ones. This limitation arises because generation-focused queries are trained to capture linguistic fluency and token dependencies, rather than evidence relevance. Nonetheless, retrieval and generation typically share a common input prefix, the question. The semantic understanding of the question is encoded into key and value representations, forming a latent memory stored in the attention cache and reused across all heads. Although these key-value (KV) pairs are designed to support next-token prediction for question answering, they also contain rich semantic information that can be repurposed for document selection.

To make use of the shared query understanding stored in the attention, we introduce a dedicated retrieval pathway within the transformer architecture by adding a separate set of query projection heads specifically designed for document selection. The retrieval-specific projection, denoted by $\mathbf{W}'_Q$, produce a new set of queries $\mathbf{Q}' = \mathbf{H}\mathbf{W}'_Q$ from the hidden representations $\mathbf{H}$. Unlike the standard query projections used for generation—which prioritize syntactic fluency—these retrieval queries are optimized to evaluate semantic relevance between the query and candidate documents. Importantly, $\mathbf{Q}'$ interacts with the same key–value (KV) pairs, $\mathbf{K} = \mathbf{H}\mathbf{W}_K$ and $\mathbf{V} = \mathbf{H}\mathbf{W}_V$, as in the main attention stream. These KV pairs encode the shared semantic information extracted from the input prefix (the query), serving as a latent memory available to all heads. The retrieval attention is computed as:

$$\text{Attn}'(\mathbf{H}) = \text{softmax}\left(\frac{\mathbf{Q}'\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}.$$

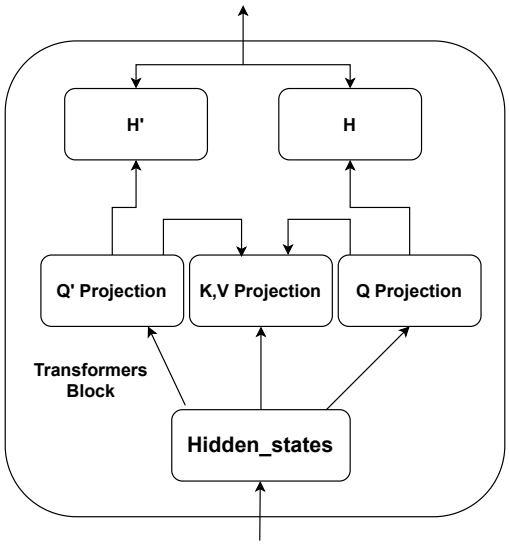

Figure 2: Transformer block design for unified retrieval and generation. Left stream for retrieval. Right stream for the original generation.

To further process the retrieval-specific attention output, we introduce an additional MLP layer that mirrors the standard feedforward network used in Transformer blocks. Rather than feeding this into the same MLP used for generation, we apply an additional retrieval-specific MLP, defined as:

$$\mathbf{H}'_{\text{out}} = \text{MLP}'(\text{Attn}'(\mathbf{H})).$$

For comparison, the standard generative path proceeds as:

$$\mathbf{H}_{\text{out}} = \text{MLP}(\text{Attn}(\mathbf{H})).$$

This design decouples the computation paths for generation and retrieval, allowing the retrieval-specific MLP to specialize in evaluating document relevance without interfering with generation fluency. After processing, the retrieval-enhanced output is integrated back into the main hidden state stream via a residual connection:

$$\mathbf{H}_{\text{out}} = \mathbf{H}_{\text{out}} + \mathbf{H}'_{\text{out}}. \tag{4.2}$$

Finally, we introduce a decoder head that projects the final hidden states into a vocabulary space tailored for document identifiers. This projection produces logits over the docid token space, enabling the model to autoregressively generate semantic document identifiers. With this addition, our framework equips the original language model with retrieval capabilities, allowing it to perform document selection through docid generation. Finally, we train the new added parameters by optimizing

$$L_{\text{retriever}}(\phi) = -\mathbb{E}\left[\sum_{j=1}^{m} \log p^1_{\theta,\phi}\big(i_j \mid x, i_{[1:j-1]}\big)\right], \tag{4.3}$$

Here, θ denotes the parameters of the original language model, while ϕ represents the newly introduced parameters for the retrieval component. We distinguish between two output modes depending on whether retrieval is active. In the **first** output mode, denoted as $p^1_{\theta,\phi}$, the model generates document identifiers from the docid vocabulary space. During the pretraining of ϕ, we freeze θ, ensuring that the retrieval module learns independently without degrading the generation ability of the base language model. When the residual connection in (4.2) is blocked, the system operates solely as a generator, behaving identically to the original language model. However, when (4.2) is enabled, the system transitions into the **second** output mode, denoted as $p^2_{\theta,\phi}$, where document retrieval and answer generation are combined. In this case, the retrieved documents can be incorporated into the generation process, enabling the model to produce answers that are both contextually grounded and faithful to the retrieved evidence.

4.3 Joint process with cross attention

In the previous section, we have described our unified framework for retrieval and generation: the model performs retrieval when the residual path in Equation (4.2) is active, and generation when this path is blocked. However, even under this unified setting, the two components—retrieval and generation—operate in parallel, without direct interaction. This limits the ability of the generation process to dynamically leverage evidence surfaced during retrieval. To more tightly couple the two, we introduce a cross-attention layer that explicitly bridges their hidden states. To be more specific, we consider

$$\mathbf{H}_{\text{out}} = \mathbf{H}_{\text{out}} + \mathbf{H}'_{\text{out}} + \text{CrossAttention}(\mathbf{H}_{\text{out}}, \mathbf{H}'_{\text{out}}), \tag{4.4}$$

where the cross attention function is defined as:

$$\text{CrossAttention}(\mathbf{H}_{\text{out}}, \mathbf{H}'_{\text{out}}) = \text{softmax}\left(\frac{\mathbf{Q}'_{\text{out}}{\mathbf{K}_{\text{out}}}^{\top}}{\sqrt{d_k}}\right)\mathbf{V}_{\text{out}},$$

with $\mathbf{Q}'_{\text{out}} = \mathbf{H}'_{\text{out}}\mathbf{W}^Q_{\text{out}}$, $\mathbf{K}_{\text{out}} = \mathbf{H}_{\text{out}}\mathbf{W}^K_{\text{out}}$, $\mathbf{V}_{\text{out}} = \mathbf{H}_{\text{out}}\mathbf{W}^V_{\text{out}}$. This design provides a direct communication channel between the retrieval-enhanced representations and the generation stream. Unlike independent processing, the retrieval pathway contributes relevance-aware signals that guide generation, and generation in turn reinforces which aspects of retrieval are most useful. This mutual interaction forms a tight bridge between the two processes, enabling the model to more effectively ground its responses in selected documents without losing fluency.

4.4 End-to-end Joint Training

With all the structural designs described above, we are now able to conduct end-to-end joint training of the complete retrieval–generation system. In a complete single training step, the model first performs a forward pass through the retriever to select the document identifiers most relevant to the query. By looking up the corresponding document contents, we then concatenate them with the query and perform a second forward pass through the generator to produce the final answer. This two-stage design unifies retrieval and generation in a differentiable pipeline, enabling shared optimization of both components.

Different from prior work, our framework naturally supports two distinct training modes.

Mode 1: When the ground-truth document id $i_d = [i_1, i_2, \ldots, i_m]$ is available, we can minimize the loss incurred by the first forward process, which directly supervises the retriever and generator jointly from the retriever perspective:

$$L^1_{\text{joint}}(\theta, \phi) = -\mathbb{E}\left[\sum_{j=1}^{m} \log p^1_{\theta,\phi}\big(a^j_i \mid x, a^{[1:j-1]}_i\big)\right] \tag{4.5}$$

Mode 2: On the other hand, when explicit document evidence is not available in the training data, we rely on the retrieval component to propose relevant candidates. In this case, we first sample document paragraphs using $p^1_{\theta,\phi}$ and then evaluate the model directly through the end-to-end QA generation process. The loss is thus computed with respect to the final answer generation, conditioned on both the query x and the retrieved documents $\mathcal{D}^*$:

$$L^2_{\text{joint}}(\theta, \phi) = -\mathbb{E}\left[\sum_{j=1}^{m} \log p^2_{\theta,\phi}\big(a^j_i \mid x, \mathcal{D}^*, a^{[1:j-1]}_i\big)\right]. \tag{4.6}$$

This formulation ensures that the parameters ϕ, θ are trained not only to approximate ground-truth docids, but also to optimize for downstream QA performance. In other words, the retriever is rewarded for selecting documents that lead to better answers under the joint distribution $p^2_{\theta,\phi}$.

5 EXPERIMENTS

5.1 EXPERIMENT SETUP

In our experiment, we evaluate our model in commonly used open-QA datasets.

Natural Questions (NQ). The NQ dataset (Kwiatkowski et al., 2019), is built from real, anonymized, and aggregated queries issued to the Google search engine, paired with corresponding Wikipedia pages. Each example contains a natural user query along with a human-annotated answer span, which may be either long or short. In our experiments, we leverage the query–document correspondence data for retrieval warm-up training. For the QA task, we adopt the open-domain version (NQ-Open) introduced by Lee et al. (2019), where only questions with short-form answers are retained, and models must retrieve the supporting evidence from the full Wikipedia corpus.

TriviaQA. The TriviaQA dataset (Joshi et al., 2017) is a reading comprehension dataset. The questions are authored by Trivia enthusiasts, forming natural question–answer pairs. On average, each question is associated with six supporting evidence documents, which are collected retrospectively from both Wikipedia and the Web. A single query in TriviaQA may correspond to multiple reference documents and multiple valid answers.

For the large language model backbone, we select two competitive open-source models for the generation task: Llama-3.1-8B-Instruct (Dubey et al., 2024) and Qwen3-4B-Instruct-2507 (Yang et al., 2025). We evaluate these models under three different settings:

- No RAG: The model answers questions without retrieval augmentation.
- Vanilla RAG: Use dense passage retriever (DPR) (Karpukhin et al., 2020) for retrieval.
- Our method: Unify retrieval and generation and do joint training.

In our method, we divide the training pipeline into two different stages.

5.2 TRAINING RETRIEVAL

We begin by conducting a warm-up training stage for the retriever, where we minimize the objective in (4.5). To represent document identifiers, we reserve a set of special tokens and initialize their embeddings by copying from existing token embeddings. In addition, we introduce a retrieval marker token, <retrieve_token>, which signals the start of docid generation. This warm-up phase ensures that the retriever learns to associate queries with their corresponding documents based on the training data format, thereby guaranteeing that the retrieved documents are meaningful and relevant. We close the cross-attention in this stage.

5.3 TRAINING GENERATOR

During generator training, we augment the base model with a LoRA structure (Hu et al., 2022), enabling parameter-efficient adaptation. In this stage, we jointly optimize the newly introduced parameters for the retriever, the cross-attention module, and the LoRA components. To preserve retrieval capability and prevent degradation, we train on a balanced mixture of data from both mode 1 (retrieval-focused) and mode 2 (generation-focused). Consequently, the overall training objective becomes a weighted combination of the two losses, (4.5) and (4.6). This design is feasible because the retriever and generator share parameters, allowing both components to reinforce each other while maintaining consistency across tasks. Moreover, we introduce a generation marker token, <generates_token>, which signals the start of answer generation.

Table 1: Evaluation of our method on open-QA datasets

Model	Method	NQ	TriviaQA
Llama3.1-8B-Instruct	No RAG	28.8	62.0
	Vanilla RAG	47.7	64.1
	Our method	**51.3**	**71.9**
Qwen3-4B-Instruct	No RAG	29.3	**57.5**
	Vanilla RAG	40.8	47.4
	Our method	**43.2**	47.9

5.4 Experiment Results

We present our experimental results in Table 1. Overall, the results demonstrate that our method can be regarded as a more effective retrieval-injection strategy compared with the baselines. The performance improvement is more pronounced on NQ than on TriviaQA. One possible reason is that TriviaQA often contains multiple valid answers per query, whereas our training setup only uses the first annotated answer as the label to reduce computational cost.

6 Discussion

The central idea of this paper is to unify retrieval and generation within a single model, mirroring the natural way humans consult documents: we recall potential sources, select the most relevant ones, and integrate them directly into reasoning. We believe that this joint perspective is also theoretically meaningful from an information-theoretic standpoint. In particular, Xu et al. (2024b) characterize large language models as performing latent variable inference. Given a prefix $x, a_{[1:i-1]}$, the probability of generating the next token a_i can be described as

$$p\big(a_i|x, a_{[i-1]}\big) = \int_{\mathcal{Z}} p(a_i|x, a_{[1:i-1]}, z) \cdot p(z|x, a_{[1:i-1]}) \mathrm{d}z,$$

where $\mathcal{Z}$ is the space of high dimensional concept variable. Given a set of evidence documents $\mathcal{D}^*$, the probability distribution shifts accordingly, altering both the conditional likelihood of tokens and the posterior over latent concepts:

$$p\big(a_i|x, \mathcal{D}^*, a_{[i-1]}\big) = \int_{\mathcal{Z}} \underbrace{p(a_i|x, \mathcal{D}^*, a_{[1:i-1]}, z)}_{I_1} \cdot \underbrace{p(z|x, \mathcal{D}^*, a_{[1:i-1]})}_{I_2} \mathrm{d}z.$$

Following Xu et al. (2024b), using the Bayesian formula, we can represent I_2 as

$$I_2 \propto \underbrace{p(\mathcal{D}^*, a_{[1:i-1]}|x, z)}_{I_3} \cdot p(z|x).$$

Therefore, with some further analysis, Xu et al. (2024b) explained the benefit and detriment of retrieval augmented generation as distribution completion and distribution contradiction. We extend this view by analyzing how end-to-end training can optimize these distributions with data. In particular, we assume that $p(z|x)$ remains fixed, as it is primarily determined during large-scale pretraining and reflects the intrinsic concept distribution given the input. Consequently, the optimization in downstream tasks focuses on the shifted terms. Specifically, I_1 captures the probability of generating the correct answer conditioned on both the evidence and the latent concept, while I_2 governs the ability to retrieve appropriate documents by shaping the posterior over concepts. This aligns with our intuition that it is preferable to let the language model select the supporting documents by itself as the retrieval process becomes an internal component of the model's inference.

REPRODUCIBILITY STATEMENT

In Section 4, we provide a detailed description of the model architecture used in this paper. Additional implementation and experimental details can be found in Appendix A.

ETHICS STATEMENT

In this paper, we employ large language models (LLMs) to address standard open-domain question answering (QA) tasks. All documents used are drawn from widely adopted benchmark datasets, primarily consisting of Wikipedia articles and other reputable web sources. During the training process, the LLM does not produce any harmful content, including discriminatory, biased, or unfair outputs. As a result, this work does not raise any ethical concerns related to data usage or model behavior.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. 2024.

Arian Askari, Chuan Meng, Mohammad Aliannejadi, Zhaochun Ren, Evangelos Kanoulas, and Suzan Verberne. Generative retrieval with few-shot indexing. *arXiv preprint arXiv:2408.02152*, 2024.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Ingeol Baek, Hwan Chang, Byeongjeong Kim, Jimin Lee, and Hwanhee Lee. Probing-rag: Self-probing to guide language models in selective document retrieval. *arXiv preprint arXiv:2410.13339*, 2024.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pp. 2206–2240. PMLR, 2022.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*, 2017.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. The power of noise: Redefining retrieval for rag systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 719–729, 2024.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. Autoregressive entity retrieval. *arXiv preprint arXiv:2010.00904*, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*, 2018.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pp. arXiv–2407, 2024.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.

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. *ACM Transactions on Information Systems*, 2024.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pp. 3929–3938. PMLR, 2020.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Yuntong Hu, Zhihan Lei, Zheng Zhang, Bo Pan, Chen Ling, and Liang Zhao. Grag: Graph retrieval-augmented generation. *arXiv preprint arXiv:2405.16506*, 2024.

Jie Huang, Wei Ping, Peng Xu, Mohammad Shoeybi, Kevin Chen-Chuan Chang, and Bryan Catanzaro. Raven: In-context learning with retrieval-augmented encoder-decoder language models. *arXiv preprint arXiv:2308.07922*, 2023.

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

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.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C Park. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. *arXiv preprint arXiv:2403.14403*, 2024.

Zhengbao Jiang, Luyu Gao, Jun Araki, Haibo Ding, Zhiruo Wang, Jamie Callan, and Graham Neubig. Retrieval as attention: End-to-end learning of retrieval and reading within a single transformer. *arXiv preprint arXiv:2212.02027*, 2022.

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7969–7992, 2023.

Zhuoran Jin, Pengfei Cao, Yubo Chen, Kang Liu, Xiaojian Jiang, Jiexin Xu, Qiuxia Li, and Jun Zhao. Tug-of-war between knowledge: Exploring and resolving knowledge conflicts in retrieval-augmented language models. *arXiv preprint arXiv:2402.14409*, 2024.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *EMNLP (1)*, pp. 6769–6781, 2020.

Varsha Kishore, Chao Wan, Justin Lovelace, Yoav Artzi, and Kilian Q Weinberger. Incdsi: Incrementally updatable document retrieval. In *International conference on machine learning*, pp. 17122–17134. PMLR, 2023.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

John Lafferty and Chengxiang Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 111–119, 2001.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300*, 2019.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Xiaoxi Li, Yujia Zhou, and Zhicheng Dou. Unigen: A unified generative framework for retrieval and question answering with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 8688–8696, 2024.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022.

Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Richard James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, et al. Ra-dit: Retrieval-augmented dual instruction tuning. In *The Twelfth International Conference on Learning Representations*, 2023.

Na Liu, Liangyu Chen, Xiaoyu Tian, Wei Zou, Kaijiang Chen, and Ming Cui. From llm to conversational agent: A memory enhanced architecture with fine-tuning of large language models. *arXiv preprint arXiv:2401.02777*, 2024.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*, 2023.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Costas Mavromatis and George Karypis. Gnn-rag: Graph neural retrieval for large language model reasoning. *arXiv preprint arXiv:2405.20139*, 2024.

Sanket Vaibhav Mehta, Jai Gupta, Yi Tay, Mostafa Dehghani, Vinh Q Tran, Jinfeng Rao, Marc Najork, Emma Strubell, and Donald Metzler. Dsi++: Updating transformer memory with new documents. *arXiv preprint arXiv:2212.09744*, 2022.

Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. Graph retrieval-augmented generation: A survey. *arXiv preprint arXiv:2408.08921*, 2024.

Juan Ramos et al. Using tf-idf to determine word relevance in document queries.

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

Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. Dragin: dynamic retrieval augmented generation based on the information needs of large language models. *arXiv preprint arXiv:2403.10081*, 2024.

Weihang Su, Yichen Tang, Qingyao Ai, Junxi Yan, Changyue Wang, Hongning Wang, Ziyi Ye, Yujia Zhou, and Yiqun Liu. Parametric retrieval augmented generation. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1240–1250, 2025.

Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten Rijke, and Zhaochun Ren. Learning to tokenize for generative retrieval. *Advances in Neural Information Processing Systems*, 36:46345–46361, 2023.

Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. Transformer memory as a differentiable search index. *Advances in Neural Information Processing Systems*, 35:21831–21843, 2022.

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

Ravisri Valluri, Akash Kumar Mohankumar, Kushal Dave, Amit Singh, Jian Jiao, Manik Varma, and Gaurav Sinha. Scaling the vocabulary of non-autoregressive models for efficient generative retrieval. *arXiv preprint arXiv:2406.06739*, 2024.

Boxin Wang, Wei Ping, Lawrence McAfee, Peng Xu, Bo Li, Mohammad Shoeybi, and Bryan Catanzaro. Instructretro: Instruction tuning post retrieval-augmented pretraining. *arXiv preprint arXiv:2310.07713*, 2023.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345, 2024a.

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. *arXiv preprint arXiv:2207.02578*, 2022a.

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

Yuhao Wang, Ruiyang Ren, Junyi Li, Wayne Xin Zhao, Jing Liu, and Ji-Rong Wen. Rear: A relevance-aware retrieval-augmented framework for open-domain question answering. *arXiv preprint arXiv:2402.17497*, 2024b.

Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, et al. A neural corpus indexer for document retrieval. *Advances in Neural Information Processing Systems*, 35:25600–25614, 2022c.

Zihao Wang, Anji Liu, Haowei Lin, Jiaqi Li, Xiaojian Ma, and Yitao Liang. Rat: Retrieval augmented thoughts elicit context-aware reasoning in long-horizon generation. *arXiv preprint arXiv:2403.05313*, 2024c.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Shi Weijia, Min Sewon, Yasunaga Michihiro, Seo Minjoon, James Rich, Lewis Mike, and Yih Wentau. Replug: Retrieval-augmented black-box language models. *arXiv preprint ArXiv:2301.12652*, 2023.

Jason Weston, Emily Dinan, and Alexander H Miller. Retrieve and refine: Improved sequence generation models for dialogue. *arXiv preprint arXiv:1808.04776*, 2018.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101, 2025.

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

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

Shicheng Xu, Liang Pang, Huawei Shen, and Xueqi Cheng. A theory for token-level harmonization in retrieval-augmented generation. *arXiv preprint arXiv:2406.00944*, 2024b.

Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. Corrective retrieval augmented generation. 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Tianchi Yang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, and Qi Zhang. Auto search indexer for end-to-end document retrieval. *arXiv preprint arXiv:2310.12455*, 2023.

Yue Yu, Wei Ping, Zihan Liu, Boxin Wang, Jiaxuan You, Chao Zhang, Mohammad Shoeybi, and Bryan Catanzaro. Rankrag: Unifying context ranking with retrieval-augmented generation in llms. *Advances in Neural Information Processing Systems*, 37:121156–121184, 2024.

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.

Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. Enhancing generative retrieval with reinforcement learning from relevance feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12481–12490, 2023.

Yujia Zhou, Jing Yao, Zhicheng Dou, Yiteng Tu, Ledell Wu, Tat-Seng Chua, and Ji-Rong Wen. Roger: Ranking-oriented generative retrieval. *ACM Transactions on Information Systems*, 42(6): 1–25, 2024.

A Experiment Details

For the doc-id generation, we directly follow the construction of Wang et al. (2022c). In the model architecture, we incorporate four additional query heads, denoted as Q', in the attention block of each layer.

For the retrieval training phase, the model is trained on 8 NVIDIA A100 GPUs, each with 80 GB of memory, and the process takes approximately 4 hours. The learning rate is set to 2×10^{-4}. In contrast, the generation training phase is more computationally intensive, requiring around 8 hours on the same hardware configuration. A smaller learning rate of 1×10^{-5} is used to ensure stable fine-tuning of the generator module.

To assess the performance of our system, we evaluate whether any of the gold answers is found within the generated output. Since in our methods, we do not apply complicated prompts, but a single token for the task identification, we utilize a basic input template that explicitly separates the query and retrieved context for fairness. The format is as follows:"Q: {query}\n\nContext:\n{context}A:".

B THE USE OF LARGE LANGUAGE MODELS (LLM)

We leverage large language models (LLMs) to structure our ideas and assist in writing logically organized and coherent paragraphs.